frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 16 October 2023
DOI 10.3389/fphy.2023.1292702

OPEN ACCESS

EDITED BY
Jianrong Wang,
Shanxi University, China

REVIEWED BY
Xiaolong Kong,
NASA Jet Propulsion Laboratory, United States
Liangyuan Wang,
Huazhong University of Science and Technology, China
Yunyun Yang,
Taiyuan University of Technology, China

*CORRESPONDENCE
Haitao Xu,
alex_xuht@hotmail.com

RECEIVED 12 September 2023
ACCEPTED 29 September 2023
PUBLISHED 16 October 2023

CITATION
Wang L, Zhou W, Xu H, Li L, Cai L and Zhou X (2023), Research on task offloading optimization strategies for vehicular networks based on game theory and deep reinforcement learning. *Front. Phys.* 11:1292702.
doi: 10.3389/fphy.2023.1292702

Research on task offloading optimization strategies for vehicular networks based on game theory and deep reinforcement learning

Lei Wang[1], Wenjiang Zhou[2], Haitao Xu[1,3]*, Liang Li[1], Lei Cai[4] and Xianwei Zhou[1]

[1]School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing, China, [2]China Academy of Information and Communications Technology, Beijing, China, [3]Shunde Innovation School, University of Science and Technology Beijing, Foshan, China, [4]The North China Institute of Computing Technology, Beijing, China

With the continuous development of the 6G mobile network, computing-intensive and delay-sensitive onboard applications generate task data traffic more frequently. Particularly, when multiple intelligent agents are involved in tasks, limited computational resources cannot meet the new Quality of Service (QoS) requirements. To provide a satisfactory task offloading strategy, combining Multi-Access Edge Computing (MEC) with artificial intelligence has become a potential solution. In this context, we have proposed a task offloading decision mechanism (TODM) based on cooperative game and deep reinforcement learning (DRL). A joint optimization problem is presented to minimize both the overall task processing delay (OTPD) and overall task energy consumption (OTEC). The approach considers task vehicles (TaVs) and service vehicles (SeVs) as participants in a cooperative game, jointly devising offloading strategies to achieve resource optimization. Additionally, a proximate policy optimization (PPO) algorithm is designed to ensure robustness. Simulation experiments confirm the convergence of the proposed algorithm. Compared with benchmark algorithms, the presented scheme effectively reduces delay and energy consumption while ensuring task completion.

KEYWORDS

multi-access edge computing, cooperative game, task offloading, proximate policy optimization, deep reinforcement learning

1 Introduction

With the advent of the Internet of Things (IoT), many sensing devices have been deployed in networks. The data generated by these devices and related large-scale mobile applications are growing explosively [1]. In the context of IoT, the Internet of Vehicles (IoV) is a study hotspot. It uses IoV technology to provide services for vehicles through onboard processors [2,3]. However, task data also increase with a significant increase in the number of vehicles. The emergence of various computing-intensive tasks poses a significant challenge to the onboard computing capability of the vehicle itself Zhou et al. [4]. Multi-Access Edge Computing (MEC) is considered a feasible method to tackle this issue. MEC has significant advantages in addressing compute-intensive tasks in the IoV system. By moving

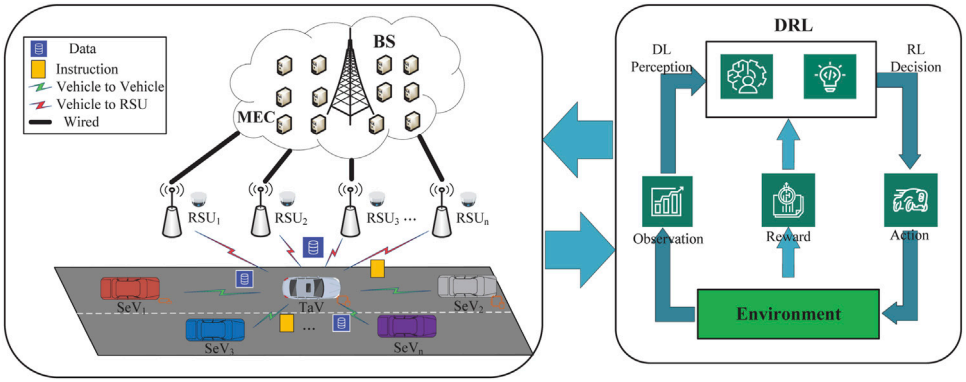

FIGURE 1
Intelligent architecture network of IoV.

TABLE 1 Parameter setting.

Parameter	Value
The transmission path loss index α	3.4
The transmission power of noise $\sigma_n^2/\sigma_{N+1}^2$	$10^{-13}w$
The wired transmission power from RSUs to MEC P'_{N+1}	2w
The height of RSUs H_{N+1}	20m
The distance from RSUs to the road center D_{N+1}	6m
The strength of vehicle coordinate transformation M_{tra}	1000 *cycles/bit*

computational and data processing functions to the network edge, it reduces task processing latency, enabling faster real-time decision-making, which is crucial for areas such as autonomous driving, traffic optimization, and intelligent traffic management. Moreover, MEC alleviates the burden on TaV, optimizes network load, and reduces energy consumption. MEC is considered a prevalent computing paradigm that has been widely studied to promote data processing efficiency, which can perform computation services closer to the data sources Porambage et al. [5].

Specifically, in the IoV system, tasks are offloaded to the service nodes (SNs) with computing power, and tasks are processed cooperatively to improve efficiency. The premise of task offloading is that jobs can be split into multiple subtasks and offloaded to SNs. Parked or moving vehicles, as idle resources, can provide specific computing and storage resources for task processing Sookhak et al. [6]. In addition to offloading tasks to the service vehicles (SeVs), task vehicles (TaVs) can also offload tasks to the MEC servers. The MEC servers coexist with the base station (BS) and connect the roadside units (RSUs) to provide services Xiao and Zhu [7]. In recent years, the issue of task offloading in the IoV system has received extensive research Zhou et al. [8], and task processing delay and energy consumption are essential indicators. It is challenging to minimize overall delay and energy consumption while completing the task Li et al. [9]. When the amount of task data is large, the task transmission delay is high, increasing the total task delay and energy consumption. To solve this problem, integrated radar sensing and communication is a feasible solution. The integrated radar sensing and communication technology aims to reduce the task processing latency and energy consumption in the IoV, improving the efficiency and performance of task processing in IoV. By collecting data through radar sensing and sharing, instead of traditional data transmission, it reduces node waiting energy consumption and enhances the response speed of the IoV system. Its advantage lies in optimizing the overall performance of the IoV system, including perception of traffic data, improvement of communication quality, and increased accuracy of vehicle positioning, thereby enhancing the efficiency and safety of the entire IoV system. Game theory and optimization techniques provide technical support for it. In this study, a game theoretic approach was utilized to construct a game model, analyze the cooperative relationship between TaVs and SNs, and define the utility function for task offloading. This facilitated the development of an optimal task offloading decision strategy, encompassing task allocation and resource coordination. Optimization techniques were employed to achieve an optimal allocation of resources, including computing, storage, and communication resources, maximizing system utility while minimizing task processing delay and energy consumption.

Some scholars have conducted some studies on this issue. For example, in [10], a relatively practical IoV scenario was considered, and a matching game method was used to model the task allocation. The simulation results show that the input data transmission delay accounts for 73% of the total task processing time. In [11], the task is assigned to the MEC and the SeVs for processing. The results show that when the task size is 80 Mb, the input data transfer delay accounts for 50%. The delay in uploading data can significantly affect delay-sensitive applications. Therefore, several cars have an integrated radar system to sense the surrounding environmental data for local processing or assist connected vehicles in processing task data to ensure safe driving [12]. Furthermore, RSUs use radar to sense environmental data and use ecological data as input to reduce the transmission delay [13]. In summary, instruction transfer and environmental data sensing provide new possibilities for task offloading. For the issue of transmission delay, consider perceived environmental data and calculation instructions to reduce transmission delay [14].

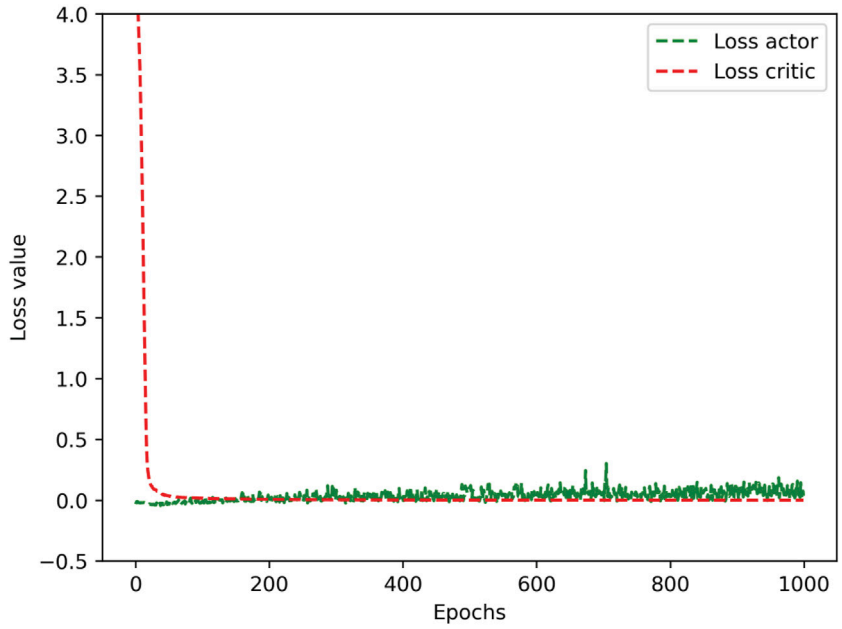

FIGURE 2
Training curve of PPOTR.

Energy is currently a major concern worldwide, and the increase in the number of IoV equipment will lead to increased energy demand and higher energy costs. Therefore, reducing energy consumption has become one of the issues that the IoV system needs to resolve [15]-[16]. To tackle this issue, Cesarano et al. designed a greedy heuristic algorithm to reduce the energy consumption of the task [17]. Some scholars applied minimizing of energy consumption and execution delay as the objective functions and reasonably selected the task offloading strategies [18]. In [19], the IoV system data transmission scheme adopts the deep Q-network (DQN) method to reduce transmission costs. Altogether, energy consumption is a key factor influencing the task offloading strategy. The focus of future study will be the proper selection of task offloading strategies to ensure delay and energy consumption.

For the aforementioned issues, many studies have adopted heuristic algorithms to solve them. For example, the author considers the reliability of task offloading in the IoV scenarios and uses heuristic algorithms to optimize the reliability [20]. Aiming at the issue of poor robustness of traditional heuristic algorithms for continuous state and action space in the IoV scenario [21], an offloading strategy-based method was studied to learn the optimal mapping from constant input state to discrete output and deal with continuous state space and action space scenarios. Although the aforementioned algorithm can solve the issue of the task offloading strategy in the IoV, the algorithm used has poor robustness in ensuring the reliability of data transmission [22].

DRL algorithms have significant advantages over heuristic algorithms. First, DRL algorithms can automatically learn and optimize decision strategies through large-scale data, eliminating the need for manual design of complex rules. Second, DRL algorithms can handle high-dimensional and complex state and action spaces, making them suitable for solving complex real-world problems. Additionally, DRL algorithms have the ability to generalize learned knowledge to unseen environments, enabling more intelligent and flexible decision-making. Given the more significant potential and application value of policy-based deep reinforcement learning (DRL) [23], this paper discusses task offloading based on DRL. For sensitive applications with environmental data as input, we proposed a task offloading decision mechanism (TOMD) based on cooperative game and DRL. This paper is based on cooperative game theory, considered the overall task processing delay (OTPD) and overall task energy consumption (OTEC), and constructed a joint optimization issue. We transformed the joint optimization issue into a DRL issue and used the PPO algorithm to solve the issue. The main contributions are summarized as follows:

1. Considering dynamic wireless edge computing networks, a framework for joint task offloading is designed. On this basis, according to the wireless transmission requirements of SNs, combined with the game theory and communication function, a cooperative game and DRL-based TODM is proposed. The joint optimization issue is derived to minimize the delay and energy consumption.
2. DRL is more robust than the heuristic algorithm as it can make real-time online decisions. Therefore, combined with DRL, the designed joint optimization issues transformed into reinforcement learning (RL) issues. This paper develops an algorithm based on PPO to solve the aforementioned issues and theoretically analyze the algorithm's complexity.
3. Finally, we designed a simulation experiment to evaluate the algorithm's performance. The results show that the algorithm

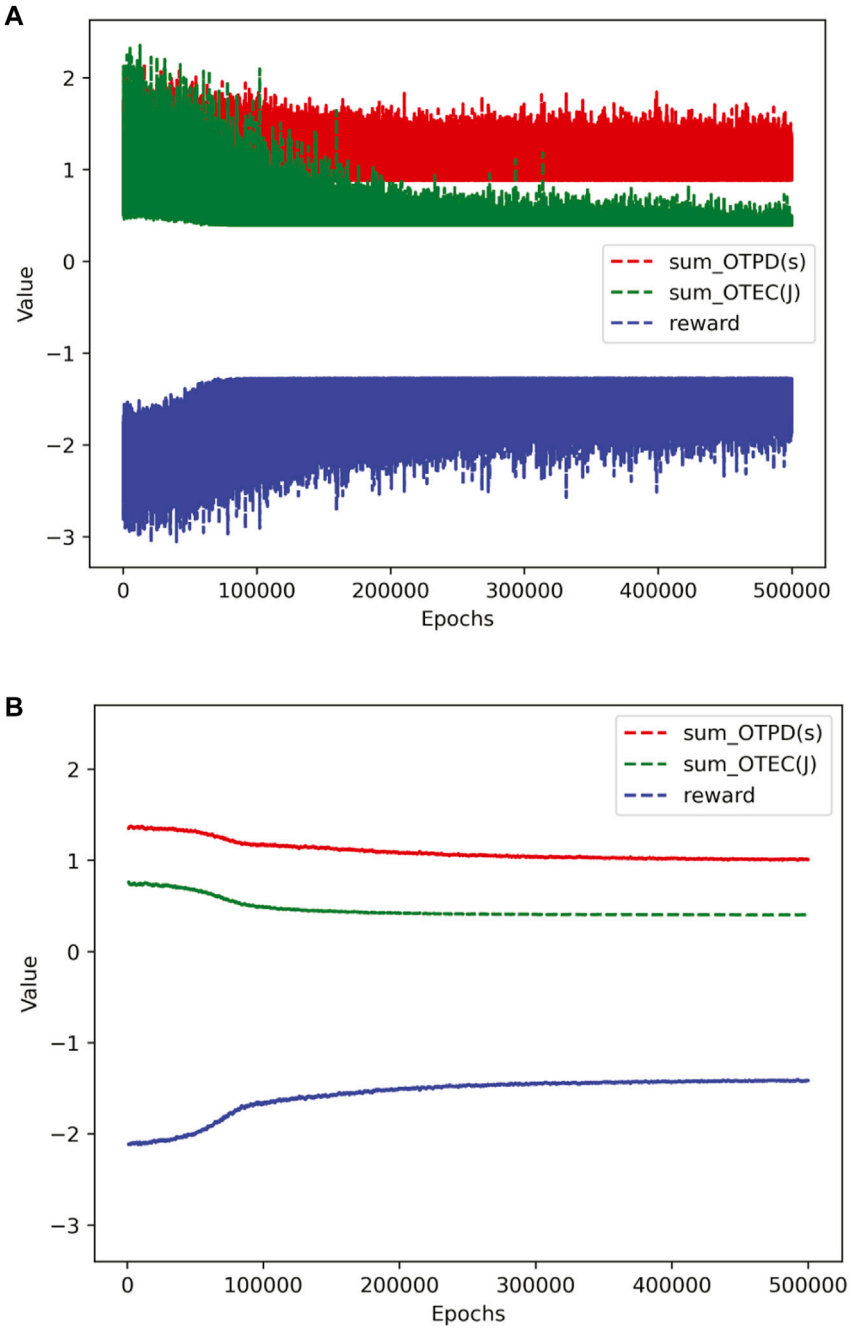

FIGURE 3
Convergence curve of PPOTR. **(A)** Unsmoothed convergence curve of PPOTR. **(B)** Smoothed convergence curve of PPOTR.

converges better than the soft actor-critic (SAC) algorithm, which can achieve the goal of a reasonable choice of the task offloading strategy. The proposed algorithm can reduce the task delay and energy consumption cost while improving the performance of the IoV system.

The remainder of this paper is arranged as follows: Section 2 presents relevant work. Section 3 presents the system model in detail, expounds on the task offloading mechanism TODM, and gives the issue formulation. Section 4 proposes a task offloading algorithm based on DRL to solve the aforementioned issues. Section 5 proposes a simulation for evaluating the solution. Finally, Section 6 summarizes this paper.

2 Related works

This section summarizes the current study of the IoV, including the connected study of task offloading, radar sensing and communication, game theory, and DRL of the IoV edge intelligent system.

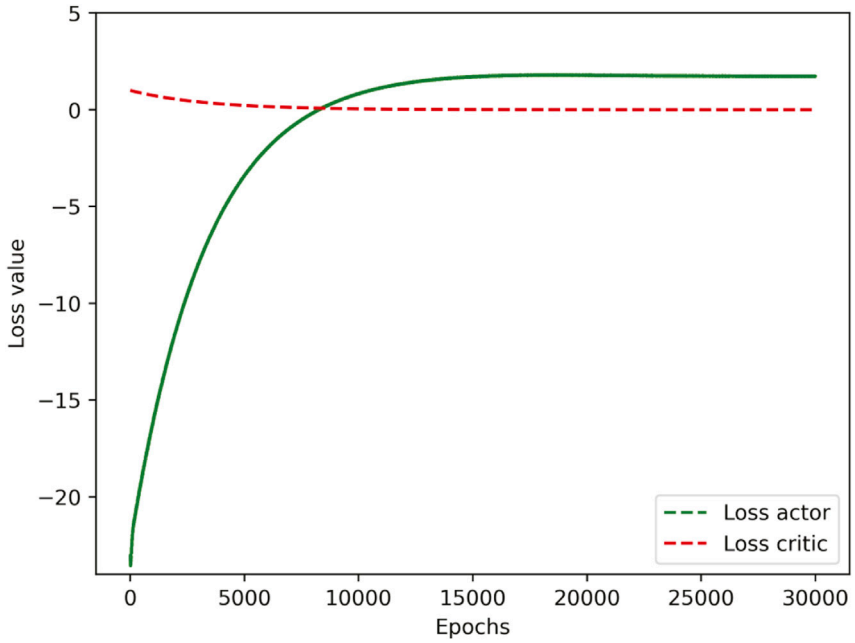

FIGURE 4
Training curve of SAC.

2.1 Task offloading of IoV

With the advent of the 6G era, mission data volume has experienced a blowout growth. With the intellectual development of the IoV intelligent system applications, the requirements for task data computation have also improved. Because the cloud is relatively far from users, traditional cloud computing has relatively high latency, which has become the focus of the task offloading strategy [24]. Researchers considered MEC as an effective technique to address the delay issue. Because the MEC servers are closer to users than the cloud, they can reduce the delay in task processing and enhance the user experience [25].

In light of MEC characteristics, it will be widely used in the future IoV system. In [26], the architecture of the vehicle network was defined according to the properties of the MEC, which can enhance the scalability of the network. In [27], an SDN-enabled network architecture assisted by the MEC was proposed to provide low-latency and high-reliability communication. In [28], the optimal task offloading issue in MEC was studied, which was transformed into two subproblems, task offloading and resource allocation, to minimize the delay.[29], considers an edge server and describes the computing and physical resource problems as optimization issues. In [30], a new offloading method was proposed to minimize transmission delay while improving resource utilization. In [31], a task offloading scheme fuzzy-task-offloading-and-resource-allocation (F-TORA) based on Takagi–Sugeno fuzzy neural network (T-S FNN) and game theory is designed.[32] proposes a UAV-assisted offloading strategy, which has been experimentally verified to reduce the delay by 30%.

2.2 Radar sensing and communication in the IoV

The integrated radar and communication design has great potential in cost-constrained scenarios. For example, by combining radar and communication functions, an IoV system can be designed to solve the issues of high latency and energy consumption. Some scholars have proposed a path estimation method to realize longitudinal and lateral vehicles followed only by radar and vehicle-to-vehicle (V2V) [33]. This paper introduces an intelligent real-time dual-functional radar–communication (iRDRC) system for autonomous vehicles (AVs) [34]. Obstacle detection is a very important part of the realization of intelligent vehicles. To avoid the problem that metal objects seriously block the millimeter wave, an active obstacle detection method based on a millimeter-wave radar base station is proposed [35]. The radar and communication integrated system (RCIS) can overcome the time-consuming problems of data format transfer and complex data fusion across multiple sensors in autonomous driving vehicles (ADVs) [36]. In summary, the integrated radar and communication design is a promising direction for future autonomous driving technology development.

2.3 Game theory in ToV

Game theory provides a framework for analyzing strategic interactions among rational decision-makers, while optimization techniques are designed to seek the most favorable outcomes. Some scholars have proposed a dependable content distribution framework that combines big data-based vehicle trajectory prediction with coalition game-based resource allocation in cooperative vehicular networks [37]. This paper proposes an

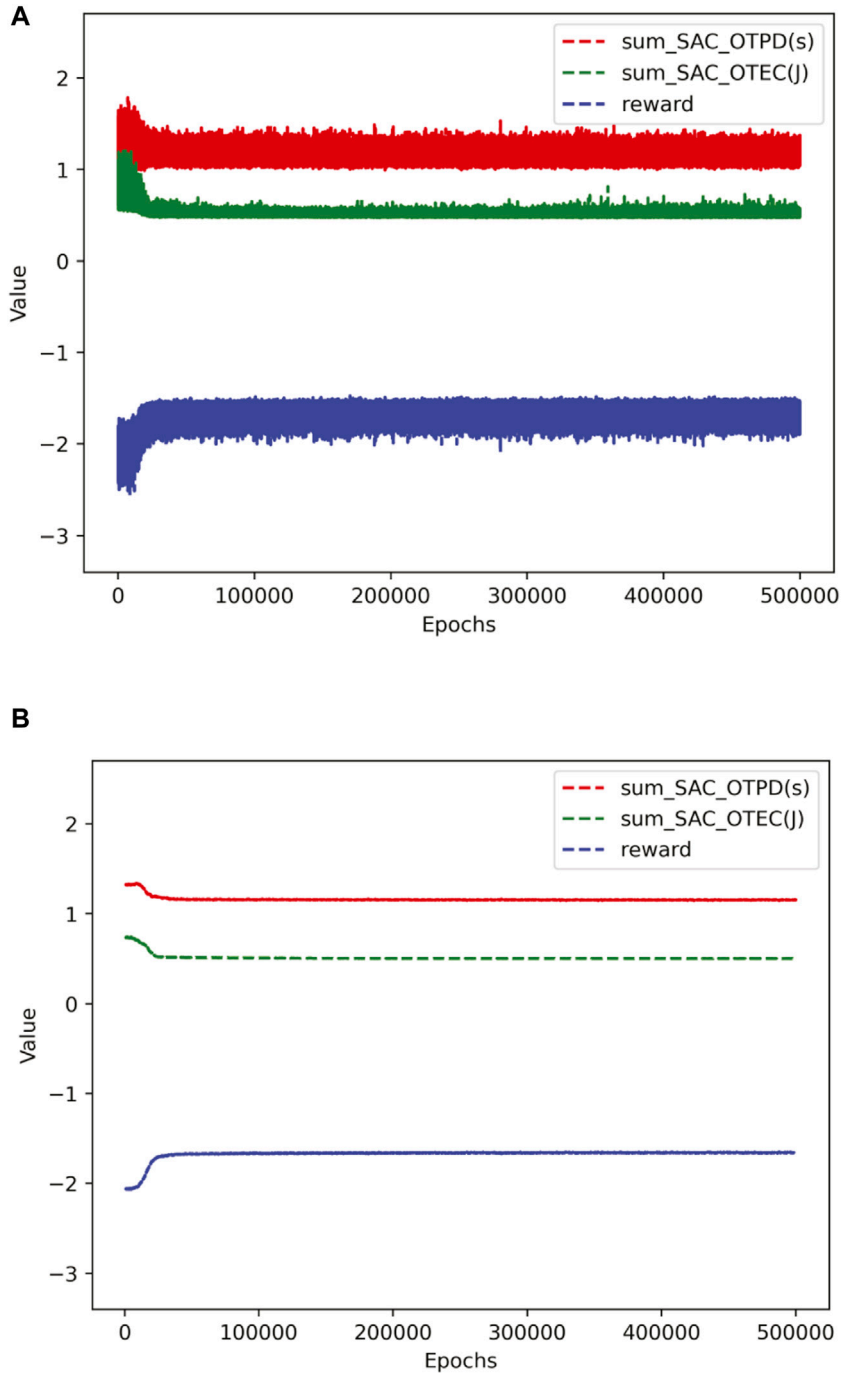

FIGURE 5
Convergence curve of SAC. **(A)** Unsmoothed convergence curve of SAC. **(B)** Smoothed convergence curve of SAC.

energy-efficient matching mechanism for resource allocation in device-to-device (D2D)-enabled cellular networks, which employs a game theoretical approach to formulate the interaction among end users and adopts the Gale–Shapley algorithm to achieve stable D2D matching [38]. Some scholars have proposed a novel game theoretical approach to encourage edge nodes to cooperatively provide caching services and reduce energy consumption [39]. In [40], the author has developed a two-player Stackelberg game-based opportunistic computation offloading scheme, which can significantly shorten task completion delay. In conclusion, game theory holds significant and extensive application prospects within the realm of the IoV.

2.4 DRL methods for IoV

Regarding resource optimization for the IoV, DRL has strong sensing and decision-making capabilities compared to traditional

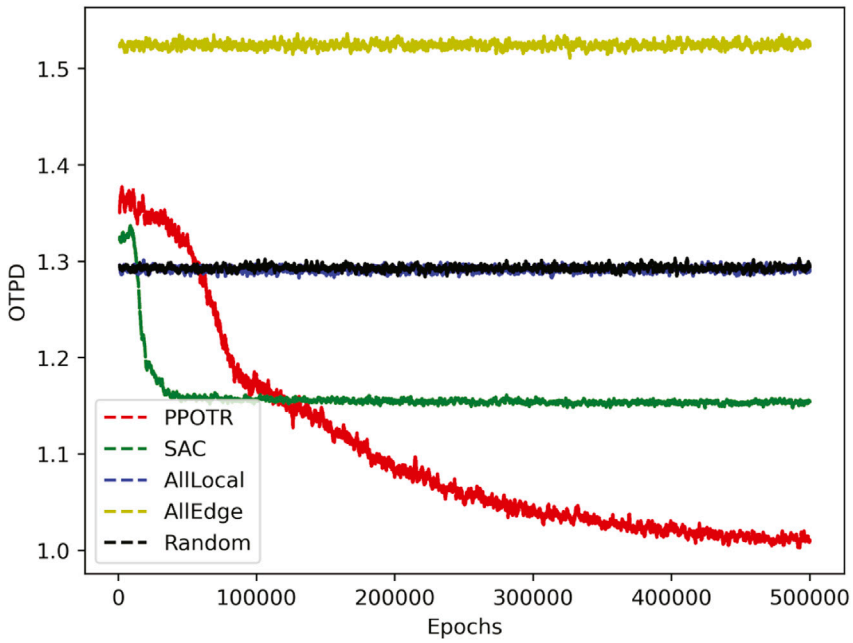

FIGURE 6
Total delay under different policies.

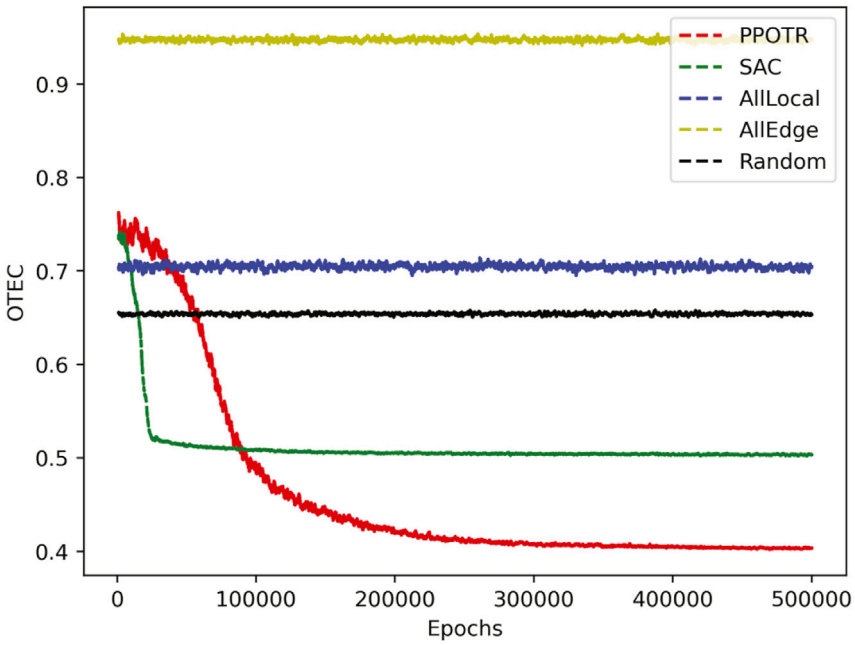

FIGURE 7
Total energy consumption under different policies.

heuristic algorithms and can analyze the long-term impact of current resource allocation on the system. Many scholars have applied DRL techniques to the study of the IoV. For example, in [41], DRL technology was used to transfer vehicle tasks to the edge server when facing the challenge of task delay. In [42], a UAV was placed in the vehicle network to assist resource allocation, and the deep deterministic policy gradient (DDPG) method was used to reduce the task delay. In [43], an online computation offloading strategy based on DQN was proposed, which takes the discrete channel gain as input to minimize energy consumption and delay

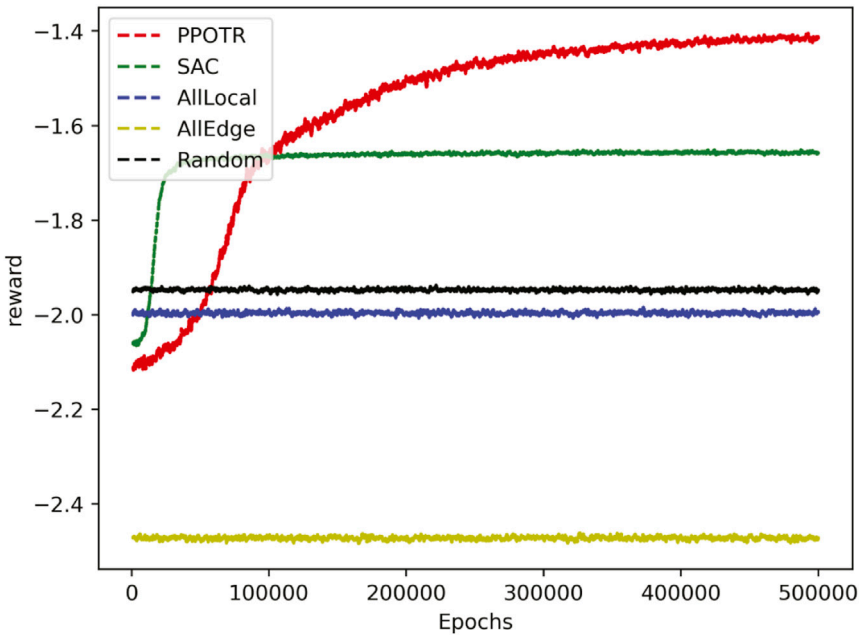

FIGURE 8
Reward under different policies.

and realize computation offloading and resource allocation. In [44], a hybrid scheduling mechanism to reduce computation was proposed for vehicle-to-vehicle communication in a specific area. In [45], the author proposed a priority-sensitive task offloading and resource allocation scheme in an IoV network to validate the feasibility of distributed reinforcement learning for task offloading in future IoV networks. In [46], the author proposed a multi-agent deep reinforcement learning (MA-DRL) algorithm for optimizing the task offloading decision strategy, while improving the offloading rate of the tasks and ensuring that a higher number of offloaded tasks are completed.

Given the preponderance of DRL techniques in the IoV system, two metrics are considered: delay and energy consumption. This paper aims to select the optimal task offloading strategy to save delay and resource costs. Consequently, we propose a framework for task offloading that uses a DRL-based algorithm to achieve optimal solutions in the network.

3 System description and problem formulation

In this section, Section 3.1 presents an edge computing network of the IoV. Section 3.2 presents an optimization issue.

3.1 System model

3.1.1 TODM mechanism based on the cooperative game

In light of the issue that the large amount of task data in the IoV leads to significant overall task delay and energy consumption, we build an intelligent system for the IoV by using the sensing capabilities of SNs. To achieve a practical and distributed solution, we realize that the task assignment problem in MEC architectures can also be formulated as a cooperative game. The cooperative game is applicable to the case of multi-node cooperation, where multiple agents work together to formulate resource allocation strategies to minimize overall delay and energy consumption. First, this paper defines the participants of the game, i.e., TaV, SeVs, and MEC. Second, it defines the strategies for task offloading, decomposing tasks into multiple subtasks assigned to different SNs, with the delay and energy consumption for nodes completing the task as the criteria for cooperative cost allocation. Finally, cooperative constraints are introduced to construct a cooperative game theory model.

The intelligent architecture network of the IoV is featured in Figure 1. A BS and MEC servers are deployed at the same location to improve MEC computing power and save costs. For RSUs reasonably deployed along the road, each RSU is equipped with storage resources and radars for real-time sensing of ambient data. The storage resources of RSUs support the storage of all sensed task data and are periodically cleared to maintain usability. RSUs are linked to the MEC via wired links. Each car is equipped with computing, storage resources, and radars. The TaV is linked to the SeVs and RSUs via wireless transmission. The communication between nodes adopts frequency division multiplexing (FDM) access technology, and the upload and feedback process adopts the time division duplexing (TDD) mode. This paper assumes that BS covers the entire IoV system, including all RSUs and vehicles. The coverage of RSUs is tangential to each other, and the TaV is always within the range of the nearest RSUs when processing the task. The task can be divided into several subtasks. Each subtask is independent and can be processed in parallel [47]. Considering the

impact of the delay and energy consumption on the offloading strategy, the MEC sends the offloading decision to the TaV and then offloads the task. In real-life scenarios, two-way roads are more practical. However, the study is still in its infancy. This paper only considers one-way lanes and ignores the car service in the opposite direction to the TaV in our model.

The delay and energy consumption are critical technical indicators in the TODM design, and this paper aims to minimize the OTPD and OTEC. The OTPD includes task description delay, offloading decision delay, offloading decision transmission delay, task upload delay, task processing delay, and task feedback delay. In this paper, each variable is represented by 64 bits, i.e., a double float. The task description size is a few kilobits, and the delay can be disregarded. The offloading decision comprises task allocation, transmission bandwidth, transmission power, and transmission policy. Compared with the amount of the task input data, the size of the offloading decision is small, so the offloading decision delay and transmission delay are disregarded. The amount of data after the task is completed is smaller than the amount of data input, and the task feedback delay can be disregarded. Thus, the OTPD consists of the task upload and computation delays. Similarly, the OTEC consists of the task upload energy consumption and the task computation energy consumption. It should be noted that if SNs perform other tasks, there will be waiting delays, and energy consumption is possible. In this paper, it is assumed that only one task needs to be processed, and the waiting delay and energy consumption are neglected. Multi-tasking will be considered in future study.

Due to the different perspectives of sensing environmental data, the TaV coordinate in the calculation instruction is used for coordinate transformation (CdT) preprocessing to eliminate differences [48]. The TaV has two ways to transmit the task: conventional data transmission (DaT) and instruction transmission (InT) with cooperative environment awareness. Different transfer methods offer new options for task offloading. The delay and energy consumption constraints affect the task upload mode, further affecting the task offloading strategy. Therefore, the transmission strategy can be chosen adaptively based on the objective function, traffic size, propagation capability, transmission delay, energy consumption, *etc.* Compared to the traditional offloading mechanism, TODM can potentially reduce energy consumption and transmission delay caused by the inputs. However, this mechanism incurs an additional cost to the overall IoV system, which is ignored in this paper.

3.1.2 Task model

The task of TaV is computationally intensive and delay-sensitive. The total task data are denoted by S_{DaT} and can be arbitrarily divided into infinitely many subtasks. The task ratio is denoted as $x_n\,(x_n \in [0,1]), n \in \mathcal{N} \coloneqq \{0,1,2,\ldots,n,N+1\}$. H is used to denote the task, and the h-th subtask is denoted as h, $h \in \mathcal{N} \coloneqq \{0,1,2,\ldots,n,N+1\}$, where $h \in H$. Some subtasks select local computations, while others select DaT or InT for SNs according to the task ratio. Task offloading to SNs can satisfy the delay and energy consumption constraints. V_n, $n \in \mathcal{N} \coloneqq \{0,1,2,\ldots,n,N+1\}$, is used to denote the SNs; the wireless bandwidth ratio is denoted as $b_n\,(b_n \in [0,1])$, $n \in \mathcal{N} \coloneqq \{0,1,2,\ldots,n,N+1\}$; and the transmission power is denoted as $P_n\,(P_n \in [0.5w, 1.5w])$, $n \in \mathcal{N} \coloneqq \{0,1,2,\ldots,n,N+1\}$. TaV is denoted as V_0, and the SeVs are denoted as V_n. The computational resources of both TaV and SeVs satisfy all computing tasks. RSUs and MEC are connected by wires, denoted by $V_N + 1$. The choice of the aforementioned three variables ensures the optimal task offloading strategy.

3.1.3 OTPD and OTEC of the TaV

When a subtask is selected to perform a local computation on V_0, the OTPD of the subtask is the local computation. The OTEC of the subtasks is the energy consumption computed locally and uploaded without energy consumption. The delay is denoted as T_0^{comput}, and the energy consumption is denoted as E_0^{comput}. The T_0^{comput} and E_0^{comput} are given as follows [49]:

$$T_0^{comput} = t_0^{comput}(x_0) = \frac{S_{DaT} x_0 M}{F_0}, \tag{1}$$

$$E_0^{comput} = e_0^{comput}(x_0) = K_0 (f_0)^2 C_0, \tag{2}$$

$$C_0 = S_{DaT} x_0 M. \tag{3}$$

Here, M (in cycles/bit) is the task calculation strength, which refers to the computing resources required to input 1 bit of data. F_0 represents the CPU cycles of V_0. K_0 is the effective switching capacitance related to the chip structure in the car. f_0 is the computing capacity of the car itself. C_0 represents the number of CPU revolutions required for processing the subtasks h_0.

3.1.4 OTPD and OTEC of the SeVs

When a subtask is offloaded to the SeVs for processing, data or calculation instructions are transmitted wirelessly to the SeVs. For DaT, it is essential to consider the upload delay. For InT, the transmission delay is not considered, but it is essential to consider the CdT delay.

Uploading delay model: Based on comparing the delays between the two upload modes, the mode with the smaller delay is selected as the upload mode. The DaT and InT upload methods are considered, and an energy consumption model is built. The energy consumption corresponding to different upload methods is calculated. T_1^{upload} is used to denote the uploading delay for V_0 transmitting the task to V_n. The uploading rate from V_0 to V_n is given by

$$R^{DaT}_{V_0 \to V_n}(t) = B_{ToT} b_n \log_2\left(1 + \frac{P_n \left|\tilde{h}_n\right|^2 d_n^{-\alpha}(t)}{\sigma_n^2}\right). \tag{4}$$

Here, $R^{DaT}_{V_0 \to V_n}(t)$ denotes the upload rate in time t. B_{ToT} denotes the total bandwidth of the wireless transmission. b_n denotes the transmission bandwidth ratio. P_n denotes the transmission power. σ_n^2 denotes the noise power. $\tilde{h}_n$ denotes the channel fading coefficient from V_0 to V_n. $d_n(t)$ denotes the distance from V_0 to V_n in time t. $d_n^{-\alpha}(t)$ denotes the path loss from V_0 to V_n. α denotes the path loss index.

During the task data upload, the car's motion causes changes in $d_n(t)$. We assume that the coordinate of V_0 is 0, and $S_{DaT} x_n$ is G_n, where $G_n \neq 0$. The moving speeds of the cars are v_0 and v_n. The formula for calculating $d_n(t)$ is given by

$$d_n(t) = \sqrt{|G_n + (v_n - v_0)t|^2}, \quad G_n \neq 0, \quad n \in \mathcal{N}. \tag{5}$$

The cars are running on the expressway, and the maximum difference between their relative speeds does not exceed 30 km/h [50]. Take 10 ms as an example; $(v_n - v_0)t = 0.008\ m$. The relative position changes are relatively small and do not affect the optimization results. This paper ignores the change in position. The calculation formula of $d_n(t)$ is given by

$$d_n(t) = |G_n| \quad G_n \neq 0 \quad n \in \mathcal{N}. \tag{6}$$

For DaT, $S_{DaT}x_n$ represents the amount of the task data allocated to V_n. $B_{ToT}b_n$ represents the transmission bandwidth from V_0 to V_n. The upload delay T_{1a}^{upload} is given by

$$\begin{aligned} T_{1a}^{upload} &= t_{1a}^{DaT}(x_n, b_n, P_n) = \frac{S_{DaT}x_n}{R_{V_0 \to V_n}^{DaT}} \\ &= \frac{S_{DaT}x_n}{B_{ToT}b_n \log_2\left(1 + \frac{P_n\left|\tilde{h}_n\right|^2 |G_n|^{-\alpha}}{\sigma_n^2}\right)}. \end{aligned} \tag{7}$$

For InT, this paper needs to consider the delay of CdT. Assume V_n stores the environmental data sensed by the radar and performs CdT immediately after receiving the calculation instruction. The delay of CdT depends on the amount of sensed data and the strength of the CdT calculation. T_{1b}^{upload} is used to denote the upload delay; T_{1b}^{upload} is given by

$$T_{1b}^{upload} = t_{1b}^{tra}(x_n) = \frac{S_{DaT}x_n M_{tra}}{F_n}, \tag{8}$$

where M_{tra} represents the computation intensity of CdT. F_n represents the CPU cycles of V_n.

Considering the TODM, DaT or InT with a lower delay is chosen as the upload method to minimize the task upload delay. The calculation formula of upload delay T_1^{upload} from V_0 to V_n is given by

$$T_1^{upload}(x_n, b_n, P_n) = \min\left\{T_{1a}^{upload}, T_{1b}^{upload}\right\}. \tag{9}$$

Uploading energy consumption model: Given the selected upload mode, the upload energy consumption model is built, and the upload energy consumption E_{1a}^{upload} and E_{1b}^{upload} are calculated. E_1^{upload} is used to denote the upload energy consumption; E_1^{upload} is given by [51]

$$E_1^{upload} = \begin{cases} E_{1a}^{upload} = e_{1a}^{upload}(x_n, b_n, P_n) = P_n T_{1a}^{upload} \\ \text{or} \\ E_{1b}^{upload} = e_{1b}^{upload}(x_n) = K_n\left(f_n\right)^2 C_{n1} \end{cases}, \tag{10}$$

$$C_{n1} = S_{DaT}x_n M_{tra}, \tag{11}$$

where K_n is the effective switching capacitor related to chip structure in cars. f_n is the calculation capacity of the car itself. C_{n1} represents the number of CPU revolutions required for processing the subtasks h_n.

Computing delay model: After the task is uploaded, the SeVs V_n start the parallel computation of the subtasks and obtain the computation delay. T_1^{comput} is used to denote the computing delay; T_1^{comput} is given by

$$T_1^{comput} = t_1^{comput}(x_n) = \frac{S_{DaT}x_n M}{F_n}. \tag{12}$$

Computing energy consumption model: The computing energy consumption model is designed according to the assigned task. C_{n2} is used to denote the number of CPU revolutions required for processing the subtasks h_n. E_1^{comput} is used to denote the computing energy consumption; E_1^{comput} is given by

$$E_1^{comput} = e_1^{comput}(x_n) = K_n\left(f_n\right)^2 C_{n2}, \tag{13}$$

$$C_{n2} = S_{DaT}x_n M. \tag{14}$$

3.1.5 OTPD and OTEC of MEC

When a subtask is offloaded to the MEC servers for processing, the upload delay includes both wireless and wired transmission delays. The upload energy consumption includes both wireless transmission energy consumption and wired transmission energy consumption.

Uploading delay model: The V_0 transmits the subtasks' data to RSUs via wireless transmission. RSUs transmit the subtasks' data to the MEC servers via wired transmission. The uploading rate from V_0 to V_{N+1} is given by

$$R_{V_0 \to V_{N+1}}^{DaT}(t) = B_{ToT}b_{N+1}\log_2\left(1 + \frac{P_{N+1}\left|\tilde{h}_{N+1}\right|^2 d_{N+1}^{-\alpha}(t)}{\sigma_{N+1}^2}\right), \tag{15}$$

where $R_{V_0 \to V_{N+1}}^{DaT}(t)$ denotes the upload rate in time t. b_{N+1} denotes the transmission bandwidth ratio. P_{N+1} denotes the transmission power. σ_{N+1}^2 denotes the noise power. $\tilde{h}_{N+1}$ denotes the channel fading coefficient from V_0 to V_{N+1}. $d_{N+1}(t)$ denotes the distance from V_0 to V_{N+1} in time t. $d_{N+1}^{-\alpha}(t)$ denotes the path loss from V_0 to V_{N+1}. During the upload of the task data, the movement of cars causes changes in $d_{N+1}(t)$. We assume that the coordinate of V_0 is 0, and V_{N+1} is G_{N+1}, where $G_{N+1} \neq 0$. The moving speed of the car is v_0; the calculation formula of $d_{N+1}(t)$ is given by

$$d_{N+1}(t) = \sqrt{|G_{N+1} - v_0 t|^2 + D_{N+1}^2 + H_{N+1}^2}, \quad G_{N+1} \neq 0, \quad N \in \mathcal{N}, \tag{16}$$

where D_{N+1} is the distance from RSUs to the centerline. H_{N+1} represents the height of RSUs. Take t = 20 ms as an example; when the speed of the car is 120 km/h, $v_0 t = 0.67$ m. The change in position is ignored compared with tens of meters. The calculation formula of $d_{N+1}(t)$ is given by

$$d_{N+1}(t) = \sqrt{|G_{N+1}|^2 + D_{N+1}^2 + H_{N+1}^2}, \quad G_{N+1} \neq 0, \quad N \in \mathcal{N}. \tag{17}$$

For DaT, $S_{DaT}x_{N+1}$ represents the amount of the task data allocated to V_{N+1}. $B_{ToT}b_{N+1}$ represents the transmission bandwidth from V_0 to V_{N+1}. The upload delay T_{2a}^{upload} is given by

$$\begin{aligned} T_{2a}^{upload} &= t_{2a}^{DaT}(x_{N+1}, b_{N+1}, P_{N+1}) = \frac{S_{DaT}x_{N+1}}{R_{V_0 \to V_{N+1}}^{DaT}} \\ &= \frac{S_{DaT}x_{N+1}}{B_{ToT}b_{N+1}\log_2\left(1 + \frac{P_{N+1}\left|\tilde{h}_{N+1}\right|^2 d_{N+1}^{-\alpha}(t)}{\sigma_{N+1}^2}\right)}. \end{aligned} \tag{18}$$

After the task data are uploaded to RSUs, RSUs will transmit the data to MEC via wired transmission. T_{2R}^{upload} is used to denote the wired upload delay. R_{wired} is used to denote the wired transmission speed. The wired upload delay T_{2R}^{upload} is given by

$$T_{2R}^{upload}(x_{N+1}) = t_{2R}^{upload}(x_{N+1}) = \frac{S_{DaT}x_{N+1}}{R_{wired}}. \quad (19)$$

Thus, let T_{2aR}^{upload} be the total upload delay, which is equal to the sum of T_{2a}^{upload} and T_{2R}^{upload}. T_{2aR}^{upload} is given by

$$T_{2aR}^{upload} = T_{2a}^{upload} + T_{2R}^{upload}. \quad (20)$$

For InT, this paper assumes CdT is carried out immediately after the MEC servers receive the calculation instruction. Let T_{2b}^{upload} be the CdT delay. The calculation formula of T_{2b}^{upload} is given by

$$T_{2b}^{upload} = t_{2b}^{Rtra}(x_{N+1}) = \frac{S_{DaT}x_{N+1}M_{tra}}{F_{N+1}}, \quad (21)$$

where F_{N+1} denotes the CPU cycles of the MEC. T_{2bR}^{upload} denotes the total upload delay, which is equal to the sum of T_{2b}^{upload} and T_{2R}^{upload}. The T_{2bR}^{upload} is given by

$$T_{2bR}^{upload} = T_{2b}^{upload} + T_{2R}^{upload}. \quad (22)$$

Similarly, DaT or InT with a lower delay is chosen as the upload method. The formula for the upload delay from V_0 to V_{N+1} is given by

$$\begin{aligned} T_2^{upload}(x_{N+1}, b_{N+1}, P_{N+1}) &= \min\left\{T_{2aR}^{upload}, T_{2bR}^{upload}\right\} \\ &= \min\left\{T_{2a}^{upload}, T_{2b}^{upload}\right\} + T_{2R}^{upload}. \end{aligned} \quad (23)$$

Uploading energy consumption model: In light of the selected upload mode, build the upload energy consumption model and calculate the upload energy consumption E_{2a}^{upload} and E_{2b}^{upload}. Each transmission mode shall transmit data from RSUs to the MEC via wired mode, using E_{2R}^{upload} to denote the energy consumption of wired transmission. The E_{2R}^{upload} is given by

$$E_{2R}^{upload} = e_{2R}^{upload}(x_{N+1}) = P'_{N+1}T_{2R}^{upload}, \quad (24)$$

where P'_{N+1} denotes the wired transmission power. E_2^{upload} denotes the total upload energy consumption. E_2^{upload} is given by

$$E_2^{upload} = \begin{cases} E_{2aR}^{upload} = E_{2a}^{upload} + E_{2R}^{upload} \\ or \\ E_{2bR}^{upload} = E_{2b}^{upload} + E_{2R}^{upload} \end{cases}, \quad (25)$$

$$\begin{aligned} E_2^{upload} &= \begin{cases} e_{2a}^{upload}(x_{N+1}, b_{N+1}, P_{N+1}) + e_{2R}^{upload}(x_{N+1}) \\ \text{or} \\ e_{2b}^{upload}(x_{N+1}) + e_{2R}^{upload}(x_{N+1}) \end{cases} \\ &= \begin{cases} P_{N+1}T_{2a}^{upload} + P'_{N+1}T_{2R}^{upload} \\ \text{or} \\ K_{N+1}\left(f_{N+1}\right)^2 C_{N+1}^1 + P'_{N+1}T_{2R}^{upload} \end{cases}, \end{aligned} \quad (26)$$

$$C_{N+1}^1 = S_{DaT}x_{N+1}M_{tra}, \quad (27)$$

where K_{N+1} is the effective switching capacitor related to chip structure in the MEC. f_{N+1} is the calculation capacity of the server itself. C_{N+1}^1 represents the number of CPU revolutions required for processing the subtasks h_{N+1}.

Computing delay model: After the task is uploaded, the MEC servers start the parallel computation of the subtasks and obtain the computation delay. $T_2{}^{comput}$ is used to denote the computing delay; $T_2{}^{comput}$ is given by

$$T_2^{comput} = t_2^{comput}(x_{N+1}) = \frac{S_{DaT}x_{N+1}M}{F_{N+1}}. \quad (28)$$

Computing energy consumption model: The computing energy consumption model is created according to the assigned task. Let C_{N+1}^2 be the number of CPU revolutions required for processing the subtasks h_{N+1}. Let E_2^{comput} be the computing energy consumption; E_2^{comput} is given by

$$E_2^{comput} = e_2^{comput}(x_{N+1}) = K_{N+1}\left(f_{N+1}\right)^2 C_{N+1}^2, \quad (29)$$

$$C_{N+1}^2 = S_{DaT}x_{N+1}M. \quad (30)$$

3.2 Problem formulation

This paper aims to solve the issue of joint task offloading based on the edge computing network of IoV, that is, to minimize the task delay and energy consumption under the constraints of limited system resources. The payoff function is the weighted sum of task processing, energy consumption, and delay. The additional weight balances the effect of energy consumption and delay on the payoff function. T^{total} is used to denote the total delay, which is given by

$$T^{total} = T_0^{comput} + T_1^{upload} + T_1^{comput} + T_2^{upload} + T_2^{comput}. \quad (31)$$

E^{total} is used to denote the total energy consumption, which is given by

$$E^{total} = E_0^{comput} + E_1^{upload} + E_1^{comput} + E_2^{upload} + E_2^{comput}. \quad (32)$$

The payoff function S_R^{total} is expressed as

$$S_R^{total}(x, b, P) = \zeta T^{total}(x, b, P) + (1-\zeta)E^{total}(x, b, P). \quad (33)$$

The payoff function is transformed into the total objective function of the joint optimization issue. The optimization problem can be described as minimizing the delay and energy consumption under task allocation, transmission bandwidth allocation, and transmit power control constraints. Thus, the optimization issue can be formulated as

$$\begin{aligned} (P1): \quad & \underset{x,b,P}{minimize} \quad S_R^{total}(x, b, P) := \min_{n=0,1,\ldots,N+1}\left\{S_R^{total}(x_n, b_n, P_n)\right\} \\ & s.t. \quad C1: \sum_{n=0}^{N+1} x_n = 1 \\ & \qquad C2: \sum_{n=1}^{n+1} b_n \le 1 \\ & \qquad C3: 0 \le x_n \le 1, \quad n = 1, 2, \ldots\ldots, N+1 \\ & \qquad C4: 0 \le b_n \le 1, \quad n = 1, 2, \ldots\ldots, N+1 \\ & \qquad C5: 0.5 \le P_n \le 1.5, \quad n = 1, 2, \ldots\ldots, N+1 \\ & \qquad C6: 0 \le \zeta \le 1 \end{aligned} \quad (34)$$

In problem P1, constraint C1 represents the task allocation ratio, and the sum of the ratio is 1. C2 denotes the allocation ratio of wireless bandwidth. The sum of the wireless bandwidth allocation ratios is less than 1. C3 and C4 represent the value range of the task allocation ratio and wireless bandwidth ratio, respectively. C5 limits the transmit power of the uplink transmission rate. C6 represents the weight value.

4 DRL-based algorithm for task offloading

Section 4.1 presents DRL techniques and the Markov decision process (MDP). Section 4.2 proposes the conversion of the

optimization issues in the model into DRL issues. Section 4.3 proposes a PPO-based approach to address the task offloading issue.

4.1 DRL-based framework

4.1.1 DRL techniques

Deep learning (DL) has strong perception ability but lacks specific decision-making abilities; RL has decision-making abilities but does not address the solving of perception issues. DRL integrates DL's perception ability and RL's decision-making ability, which solves the perceptual decision issue of complex systems. DRL is an end-to-end sensing and control method with strong generality. Its learning process can be described as follows: (i) at each moment, the agent interacts with the environment to get a high-dimensional observation and specific state characteristics. (ii) The current state is mapped to the corresponding action through the strategy, and the value function of each action is evaluated. (iii) The environment gives feedback to the action to obtain the next observation object. The optimal policy is obtained by successive cycles of the aforementioned procedure.

4.1.2 Markov decision process

Almost all issues can be formulated as MDP in the formal description of RL environments. MDP refers to the decision-maker who periodically or continuously observes the stochastic dynamic system with Markov properties and makes decisions. It includes the environmental state, action, reward, state transition probability matrix, and discount factor. The process is given a state. The agent obtains the new state by performing actions based on the state transition probability matrix. Each strategy is rewarded for its implementation.

4.2 Problem transformation

The IoV scenario has continuous state and action space, which will increase the issue's complexity. So, it is a challenge to find the best task offloading strategy. Traditional optimization algorithms require significant iterations to achieve an approximate solution when solving such issues, which does not meet the requirements of time-varying systems. However, DRL algorithms can meet real-time decision-making requirements. Therefore, this paper adopted the DRL algorithm to solve the aforementioned issues. Get the optimal task offloading strategy through continuous interaction with the IoV environment.

Problem P1 is a complex issue with continuous real variables, which have strong coupling. Task allocation, transmission power, and transmission bandwidth are all continuous real variables. Therefore, P1 is a non-convex combined issue that cannot be solved directly through mathematical calculation. In light of this, this paper turns the optimization issue into a DRL issue and proposes adopting the DRL algorithm to solve the global optimization issue. Thus, the optimization issue (34) is established as follows:

$$(P2): \quad \underset{x,b,P}{minimize} \quad \tilde{S}_R^{total} \triangleq \mathbb{E}\left[\lim_{|T|\to\infty} \frac{1}{|T|} \sum_{t\in T} S_R^{total}\right], \qquad (35)$$
$$s.t. \quad C1 - C6$$

where $\mathbb{E}(\cdot)$ represents the mathematical expectation.

In the IoV system, the cars are moving, and the vehicle status, edge server status, wireless transmission channel status, and RSU status are changing. The system needs to make different decisions to minimize delay and energy consumption and meet the reasonable allocation of resources. The transmission bandwidth and computing resources allocated by the IoV system to cars and RSUs are continuous values. Traditional DQN is mainly for discrete space. The DDPG is mainly for constant action space. The SAC and PPO can be applied to discrete and continuous spaces. Therefore, this paper designs a PPO-based method to find the optimal task offloading strategy. Next, the paper delves into the environmental state, action space, and reward function of Markov games.

4.2.1 Environment state

The environment state $S(t)$ reflects the impact of the channel condition information and agent behavior on the environment [52]. The state information includes the state of the cars, BS, and RSUs. The state of the car consists of the vehicle coordinates, transmission bandwidth, transmission power, and task allocation. The state of BS and RSUs includes the task size, transmission power, and transmission bandwidth. Each agent observes that the environment state is

$$\mathcal{S}(t) = \{U_n(t), N_n(t), R_n(t)\}, \qquad (36)$$

where $U_n(t)$, $N_n(t)$, and $R_n(t)$ denote the status of vehicles, BS, and RSUs, respectively.

4.2.2 Action space

Although the computational complexity of DRL is relatively low in large-scale network scenarios, the spatial dimension changes as the number of agents increases. The high-dimensional space will make the system calculation difficult and affect the best decision. The algorithm's performance will suffer from dimension disaster due to the high-dimensional action and state space [53]. The agent takes actions according to the currently observed state to avoid the high computational complexity, that is, jointly optimize the task allocation, transmission bandwidth allocation, uplink power control, and offloading decision. Hence, the action is

$$a(t) = \{x_n(t), b_n(t), P_n(t)\}. \qquad (37)$$

- x_n represents the task allocation policy.
- b_n represents the uplink transmission bandwidth.
- $P_n(t)$ represents the uplink transmission power.

The agent selects the offloading decision based on the present state. If the agent sets local computing for the task, the computing resources must meet the requirements. However, if the agent selects to calculate the task on the SeVs or MEC, the transmission bandwidth and computing resources must meet the needs.

4.2.3 Rewards

This paper should strictly follow constraints C1–C6 in the design of state space, action space, and reward function to optimize the task offloading strategies. The sum of the reward functions of nodes in all states is constant, and there is a competitive and cooperative relationship between nodes. Therefore, the paper sets the reward value as the opposite of the objective function. In optimization issue

P2, the agent maximizes the interests through action selection to affect the system's state. The reward function is

$$R_t = -S_R^{total}. \tag{38}$$

4.3 PPO-based algorithm framework

This paper proposes a PPO-based task offloading and resource allocation (PPOTR) algorithm to obtain stable performance in the actual changing network. The agent chooses the action to interact with the environment according to the policy, thus affecting the environment state and updating the environment parameters. Next, according to the new policy, the agent chooses actions to interact with the environment. Let $r_t(\theta)$ denote the action probability ratio of new and old strategies.

$$r_t(\theta) = \frac{\pi_\theta(a_t, s_t)}{\pi'_\theta(a_t, s_t)}. \tag{39}$$

When $r_t(\theta) > 1$, it indicates that the current strategy is more inclined to select the sampling action. Otherwise, it is not. The PPO algorithm improves the original policy gradient (PG) algorithm, and the formula of the new objective function is given by

$$L(\theta) = E_t\left[\frac{\pi_\theta(a_t, s_t)}{\pi'_\theta(a_t, s_t)} \cdot A_t\right]. \tag{40}$$

4.3.1 Training algorithm

PPO uses a new objective function to control the change in the strategy in each iteration, which is uncommon in other algorithms. The objective function is

$$L^{clip}(\theta) = E_t\left[\min\left(r_t(\theta)\right)A_t,\ clip\left(r_t(\theta), 1-\varepsilon, 1+\varepsilon\right)A_t\right)], \tag{41}$$

where θ denotes the policy parameters. E_t denotes the empirical expectation of the time step. r_t denotes the probability ratio under the new and old strategies. A_t represents the estimated advantage. ε denotes the hyperparameter. The value is usually 0.1 or 0.2. *clip* $(r_t(\theta), 1-\varepsilon, 1+\varepsilon)$ is given by

$$\text{clip}\left(r_t(\theta), 1-\varepsilon, 1+\varepsilon\right) = \begin{cases} 1-\varepsilon, & if\, r_t(\theta) \le 1-\varepsilon \\ 1+\varepsilon, & if\, r_t(\theta) \ge 1+\varepsilon \\ r_t(\theta), & otherwise \end{cases}. \tag{42}$$

4.3.2 Replay buffer

The static data in the DL differ from the data in the DRL, which is obtained according to machine learning. At each time step, the agent observes the current environment state and saves the state, action, reward, and prediction data $\text{data}_t = (s_t, a_t, r_t, s_{t+1})$ of the following environment state to the replay buffer [54]. In particular, in our model, the data of the training network will be aggregated after 1,000 time steps. We can see the data changes in the training process and avoid the correlation in the observation state sequence to reduce the update variance. Moreover, the data of each experiment can be used continuously in other weight updates to improve the efficiency of data use.

4.3.3 Algorithm steps

In the aforementioned architecture, the agent is the car, MEC is the policy decision center, and the SeVs and RSUs are the intermediaries of perception information. The algorithm's input is the environment state information, and the output is the optimal offloading policy and target value. Algorithm 1 presents the pseudo-code.

Input: initial policy parameters and initial value function parameters $\phi_\emptyset$

1: **for** k = 0, 1, 2 ... do **do**

2: Collect a set of trajectories $D_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.

3: Compute rewards-to-go R_t

4: Compute advantage estimates, A_t (using any method of advantage estimation) based on the current value function V_{θ_k}.

5: Update the policy by maximizing the PPO-clip objective, normally via stochastic gradient ascent with Adam.
$\theta_{k+1} = \arg\max_\theta \frac{1}{|D_k|T} \sum_{\tau \in D_k} \sum_{T}^{t=\emptyset} \text{m} \cdot (\frac{\pi_\theta(a_t|s_t)}{\pi'_{\theta_k}(a_t|s_t)} A^{\pi'_{\theta_k}}(s_t, a_t), g(, A^{\pi'_{\theta_k}}(s_t, a_t)))$

6: Fit value function by regression on the mean-squared error, normally via some gradient descent algorithm.
$\phi_{k+1} = \arg\min_\phi \frac{1}{|D_k|T} \sum_{T \in D_k} \sum_{T}^{t=\emptyset} (V_\phi(s_t) - R_t)^2$

7: **end for**

Algorithm 1. PPO-based algorithm for task offloading and resource allocation.

The following illustrates the steps of the proposed PPOTR algorithm. First, enter the initial policy parameter θ_0 and the value function parameter ϕ_0. Second, start iteration and collect a set of trajectories $D_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment. Then, in the fourth and fifth steps, calculate the reward value R_t and use the advantage estimation method based on the current value function V_{θ_k} to calculate the advantage estimation A_t. Then, in the sixth and seventh steps, update the strategy θ_{k+1} through PPO-clip objective function $L^{clip}(\theta)$ and the fit value function ϕ_{k+1} through mean square error regression. Finally, the algorithm iteration is ended.

Complexity analysis: The algorithm's main computational costs include the interaction with the environment, the action, and evaluation under the old and new strategies. In the process of interacting with the environment, the agent determines the input state according to the policy. Furthermore, the agent calculates the probability ratio of the action under the new and old policies through the transmission between the action network and the critic network. The time complexity of the training process interacting with the environment is given by [55]

$$C^1_{PPOTR} = \mathcal{O}\left(\mathcal{N}_r \mathcal{N}_e d^{\max} \sum_{x=0}^{X} \Omega^A_x \Omega^A_{x+1}\right). \tag{43}$$

The time complexity of policy updates is given by

$$C^2_{PPOTR} = \mathcal{O}\left(\mathcal{N}_r \mathcal{N}_s \left(\sum_{x=0}^{X} \Omega^A_x \Omega^A_{x+1} + \sum_{y=0}^{Y} \Omega^C_y \Omega^C_{y+1}\right)\right), \tag{44}$$

where x, y are the quantities of full connection layers of the network, respectively. Ω^A_x represents the unit of the x-th actor network, and Ω^C_y represents the artificial neuron of the y-th critic network. Then, the total time complexity of Algorithm 1 is given by

$$C_{PPOTR} = \mathcal{O}\left(\mathcal{N}_r\left(\mathcal{N}_s + \mathcal{N}_e d^{\max}\right)\sum_{x=0}^{X}\Omega_x^A\Omega_{x+1}^A\right) + \mathcal{O}\left(\mathcal{N}_r\mathcal{N}_s\sum_{y=0}^{Y}\Omega_y^C\Omega_{y+1}^C\right). \quad (45)$$

The space complexity of the algorithm [56] is given by

$$C_{PPOTR}^{space} = \mathcal{O}\left(\mathcal{N}_r\left(\mathcal{N}_s + \mathcal{N}_e d^{\max}\right)\sum_{x=0}^{X}\Omega_x^A\Omega_{x+1}^A\right) + \mathcal{O}\left(\mathcal{N}_r\mathcal{N}_s\sum_{y=0}^{Y}\Omega_y^C\Omega_{y+1}^C\right) + \mathcal{O}(\mathcal{N}), \quad (46)$$

where $\mathcal{N}$ is the space complexity of the experience replay buffer in the algorithm.

5 Simulation results

In this section, a series of simulation experiments to verify the performance of the proposed algorithm have been proposed. The simulation results of different algorithms under the same network settings are given to compare the characteristics of different algorithms. This paper analyzes the convergence curves of delay, energy consumption, and objective function under the offloading strategy and shows that the algorithm is reasonable. This paper adopts four benchmark schemes from the perspective of convergence, and the effectiveness and efficiency of the algorithm are verified through the analysis of energy consumption and delay. The four schemes are as follows:

- Actor critical (AC) algorithm based on SAC [57]: Under the same environmental settings, this paper uses the AC algorithm with a soft update mechanism to solve the issue.
- Local computing policies for all the tasks (AllLocal): All the tasks are performed locally, and the local computing resources meet the requirements of task calculation. Calculate the corresponding delay, energy consumption, and objective function value.
- All-edge server-only execution policy (AllEdge): Offload all the tasks to the edge server, and the edge computing resources and transmission bandwidth meet the task's requirements.
- Random offloading policy (Random): Offload the task randomly and allocate resources randomly.

5.1 Simulation setup

This paper evaluates the performance of the proposed algorithm through multiple simulation experiments. We assume that four RSUs are set at the roadside, the coverage diameter of each RSU is 160 m, and the coverage is tangential to each other. For the sake of driving safety, we assume that there is one TaV and ten SeVs within the coverage of RSUs. The TaV and SeVs are always within the coverage of RSUs. Assume the input data S_{DaT} size is 25 Mb (one frame with a resolution of 1920*1080, 12 bits per pixel), and the total transmission bandwidth B_{ToT} is 100 MHz [58].

The calculation strength is 2,640 cycles/bit [59]. The CPU frequency of each car is randomly selected within the range of $0.3 \times 10^{12} \sim 0.6 \times 10^{12}$ cycles/s, and the CPU frequency of the MEC servers is randomly selected within the range of $1\ *10^{12} \sim 2*10^{12}$ cycles/s [60]. The effective switching capacitor of the vehicle and MEC is 10^{-27}. The transmission rate R_{wired} for RSU wired transmission to the MEC is 100 Gb/s [50]. The calculation capacity of cars and MEC is set to 1.4 Gr/s and 2.8 Gr/s, respectively Song et al. [51].

This paper considers the effect of small-scale fading on transmission performance; the channel fading coefficients $|\tilde{h}_n|^2$ and $|\tilde{h}_{N+1}|^2$ are 1 [61]. The simulation experiment is completed in the environment of Pytorch 1.11.0 using Windows 10 system and Python 3.10 software. Other system parameters used in simulations are shown in Table 1.

5.2 Results

The learning curve of the PPOTR algorithm is shown in Figure 2, including the training losses of the action and criticism network. In the simulation experiment, if the value function of training loss does not tend to 0 for the action network, it proves that the whole action space has many places not explored, and there are still differences between the new and old action space. For the critic network, if the value function of training loss does not tend to 0, it proves that the critic network cannot perfectly predict the value of the state space. The simulations show that starting from the 50th training set, the value function fluctuates in a small range, and the gradient of the loss value decreases gradually. This indicates that the algorithm begins to converge and can quickly learn the optimal strategy.

Figures 3A, B show the convergence curves of the delay, energy consumption, and objective function of the PPOTR algorithm to solve the aforementioned issues. Figure 3A shows the unsmoothed curve of the training process, illustrating the total delay sum_OTPD in seconds, the total energy consumption sum_OTEC tin joules, and the reward value. Figure 3B is the convergence curve after smoothing in Figure 3A; the smooth curve is obtained by averaging the data under each training step with the previous 999 data. The simulations show that, although the curve fluctuates, the whole process tends to be flat and the algorithm converges.

The learning curve of the SAC algorithm is shown in Figure 4, including the training loss of the action and critic networks. It can be seen from the simulation results that the convergence speed of the loss function of the SAC algorithm is fast. Therefore, the algorithm can quickly learn the optimal strategy.

Figure 5A, B show the convergence curves of the delay, energy consumption, and objective function of the SAC algorithm to solve the aforementioned issues. Figure 5A shows the unsmoothed curve of the training process, illustrating the total delay sum_SAC_OTPD in seconds, the total energy consumption sum_SAC_OTEC in joules, and the reward value. Figure 5B is the convergence curve after smoothing in Figure 5A. The simulations show that the SAC algorithm converges quickly and can solve the aforementioned issues.

The aforementioned two algorithms can solve the issue in this paper. The SAC algorithm has a fast convergence rate because it scales the state characteristics before inputting data parameters into the model. There is no difference in orders of magnitude between

variables, which is conducive to optimizing the initial model. However, the convergence effect of the PPOTR algorithm is better because the algorithm is trained based on dynamic fitting data parameters. In this paper, the simulation is set to train once every 2,048 steps, so the convergence speed of the PPOTR algorithm is slow, but the convergence effect is good.

5.3 Performance comparison

In this part, this paper compares the PPOTR algorithm with the four benchmark algorithms in terms of delay, energy consumption, and reward value to verify the proposed algorithm's performance.

As shown in Figure 6, it represents the total delay of different policies under the same task data. Each scheme will converge to the optimal value with increased training times. Under the same computing task, the SAC algorithm converges faster, but the PPOTR algorithm converges better. When the task volume increases to 25 Mb, the proposed algorithm saves approximately 17.33%, 32.74%, 56.63%, and 32.63%, respectively, compared with the SAC algorithm, local computing, edge execution, and random computing of the time cost. This shows that the algorithm proposed in this paper can achieve better performance in terms of task processing delay.

Figure 7 illustrates the total energy consumption corresponding to different policies under the same task data. When the task data volume is 25 Mb, the energy consumption cost of edge execution calculation is about 2.3 times that of the PPOTR algorithm. The simulations show that the PPOTR algorithm saves approximately 25.79%, 77.53%, and 63.31%, respectively, compared with SAC, AllLocal, and Random of the energy consumption. The proposed algorithm achieves the lowest energy consumption cost.

Finally, this paper normalizes the delay and energy consumption and converts the objective function value into the reward value in DRL, as shown in Figure 8. The reward value is composed of delay and energy consumption, with these variables being highly coupled and interactive. Under the constraint conditions, the reward value is minimized to obtain the best task-unloading strategy. The reward values of the proposed algorithm in this paper were improved by 15%, 28%, 30%, and 44% compared to four baseline algorithms. Numerical comparative analysis provides strong evidence for the reliability of the algorithm and approach proposed in this paper. Compared with the four benchmark algorithms, the algorithm proposed in this paper is superior in terms of delay, energy consumption, and reward value. Therefore, this scheme can guarantee to minimize the energy consumption cost under the tolerable delay.

6 Conclusion

This paper investigates a joint optimization strategy for task offloading in the IoV edge computing network. In the IoV scenario, while considering the timeliness of task data and resource constraints, we constructed a model based on cooperative games and transformed it into a joint optimization issue. This paper models the optimization issue as a Markov game based on intelligent edge, game theory, communication, and DRL. The reward function is devised as the sum of delay and energy consumption. We adopted the PPO-based algorithm to solve the previously mentioned issue. Finally, the performance of the algorithm is verified using the simulation experiments. The numerical results show that, compared with SAC and other baseline schemes, this scheme can achieve stable convergence in the system environment and obtain the optimal reward value. This scheme minimizes the system cost and meets the development needs of the future IoV.

Data availability statement

The raw data supporting the conclusion of this article will be made available by the authors, without undue reservation.

Author contributions

LW: investigation, methodology, validation, writing–original draft, and writing–review and editing. WZ: writing–review and editing. HX: methodology, writing–original draft, and writing–review and editing. LL: writing–original draft. LC: writing–review and editing. XZ: writing–review and editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. Network data security monitoring technology and cross-border control for smart cars (2022YFB3104900). Scientific and Technological Innovation Foundation of Shunde Graduate School, USTB (BK22BF002).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

1. Song C, Zhang M, Zhan Y, Wang D, Guan L, Liu W, et al. Hierarchical edge cloud enabling network slicing for 5g optical fronthaul. *J Opt Commun Networking* (2019) 11: B60–B70. doi:10.1364/JOCN.11.000B60

2. Liu B, Jia D, Wang J, Lu K, Wu L. Cloud-assisted safety message dissemination in vanet–cellular heterogeneous wireless network. *IEEE Syst J* (2017) 11:128–39. doi:10.1109/JSYST.2015.2451156

3. Ma Q, Liu Y-F, Huang J. Time and location aware mobile data pricing. *IEEE Trans Mobile Comput* (2016) 15:2599–613. doi:10.1109/TMC.2015.2503763

4. Zhou Y, Tian L, Liu L, Qi Y. Fog computing enabled future mobile communication networks: A convergence of communication and computing. *IEEE Commun Mag* (2019) 57:20–7. doi:10.1109/MCOM.2019.1800235

5. Porambage P, Okwuibe J, Liyanage M, Ylianttila M, Taleb T. Survey on multi-access edge computing for internet of things realization. *IEEE Commun Surv Tutorials* (2018) 20:2961–91. doi:10.1109/COMST.2018.2849509

6. Sookhak M, Yu FR, He Y, Talebian H, Sohrabi Safa N, Zhao N, et al. Fog vehicular computing: Augmentation of fog computing using vehicular cloud computing. *IEEE Vehicular Techn Mag* (2017) 12:55–64. doi:10.1109/MVT.2017.2667499

7. Xiao Y, Zhu C. Vehicular fog computing: Vision and challenges. In: 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops); March 13 2017; Atlanta, GA, USA (2017). p. 6–9.

8. Zhou C, Wu W, He H, Yang P, Lyu F, Cheng N, et al. Deep reinforcement learning for delay-oriented iot task scheduling in sagin. *IEEE Trans Wireless Commun* (2021) 20: 911–25. doi:10.1109/TWC.2020.3029143

9. Li L, Cheng Q, Xue K, Yang C, Han Z. Downlink transmit power control in ultra-dense uav network based on mean field game and deep reinforcement learning. *IEEE Trans Vehicular Techn* (2020) 69:15594–605. doi:10.1109/TVT.2020.3043851

10. Gu B, Chen Y, Liao H, Zhou Z, Zhang D. A distributed and context-aware task assignment mechanism for collaborative mobile edge computing. *Sensors* (2018) 18: 2423. doi:10.3390/s18082423

11. Zhou J, Wu F, Zhang K, Mao Y, Leng S. Joint optimization of offloading and resource allocation in vehicular networks with mobile edge computing. In: 2018 10th International Conference on Wireless Communications and Signal Processing (WCSP); October 18-20, 2018; Zhejiang, China (2018). p. 1–6.

12. Qi Y, Tian L, Zhou Y, Yuan J. Mobile edge computing-assisted admission control in vehicular networks: The convergence of communication and computation. *IEEE Vehicular Techn Mag* (2019) 14:37–44. doi:10.1109/MVT.2018.2883336

13. Zhang S, Chen J, Lyu F, Cheng N, Shi W, Shen X. Vehicular communication networks in the automated driving era. *IEEE Commun Mag* (2018) 56:26–32. doi:10.1109/MCOM.2018.1701171

14. Zhang K, Mao Y, Leng S, He Y, Zhang Y. Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading. *IEEE Vehicular Techn Mag* (2017) 12:36–44. doi:10.1109/MVT.2017.2668838

15. Cesarano L, Croce A, Martins LDC, Tarchi D, Juan AA. A real-time energy-saving mechanism in internet of vehicles systems. *IEEE Access* (2021) 9:157842–58. doi:10.1109/ACCESS.2021.3130125

16. Lixin L, Yan S, Cheng Q, Dawei W, Wensheng L, Wei C. Optimal trajectory and downlink power control for multi-type uav aerial base stations. *Chin J Aeronautics* (2021) 34:11–23. doi:10.1016/j.cja.2020.12.019

17. Zhao P, Tian H, Qin C, Nie G. Energy-saving offloading by jointly allocating radio and computational resources for mobile edge computing. *IEEE Access* (2017) 5: 11255–68. doi:10.1109/ACCESS.2017.2710056

18. Wang Y, Sheng M, Wang X, Wang L, Li J. Mobile-edge computing: Partial computation offloading using dynamic voltage scaling. *IEEE Trans Commun* (2016) 64: 1–4282. doi:10.1109/TCOMM.2016.2599530

19. Zhang K, Leng S, Peng X, Pan L, Maharjan S, Zhang Y. Artificial intelligence inspired transmission scheduling in cognitive vehicular communications and networks. *IEEE Internet Things J* (2019) 6:1987–97. doi:10.1109/JIOT.2018.2872013

20. Hou X, Ren Z, Wang J, Cheng W, Ren Y, Chen K-C, et al. Reliable computation offloading for edge-computing-enabled software-defined iov. *IEEE Internet Things J* (2020) 7:7097–111. doi:10.1109/JIOT.2020.2982292

21. Yang K, Shen C, Liu T. Deep reinforcement learning based wireless network optimization: A comparative study. In: IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS); 6-9 July 2020; Toronto, Ontario (2020). p. 1248–53.

22. Kong X, Duan G, Hou M, Shen G, Wang H, Yan X, et al. Deep reinforcement learning-based energy-efficient edge computing for internet of vehicles. *IEEE Trans Ind Inform* (2022) 18:6308–16. doi:10.1109/TII.2022.3155162

23. Li L, Cheng Q, Tang X, Bai T, Chen W, Ding Z, et al. Resource allocation for noma-mec systems in ultra-dense networks: A learning aided mean-field game approach. *IEEE Trans Wireless Commun* (2021) 20:1487–500. doi:10.1109/TWC.2020.3033843

24. Xia F, Ahmed AM, Yang LT, Ma J, Rodrigues JJ. Exploiting social relationship to enable efficient replica allocation in ad-hoc social networks. *IEEE Trans Parallel Distributed Syst* (2014) 25:3167–76. doi:10.1109/TPDS.2013.2295805

25. Abbas N, Zhang Y, Taherkordi A, Skeie T. Mobile edge computing: A survey. *IEEE Internet Things J* (2018) 5:450–65. doi:10.1109/JIOT.2017.2750180

26. Liu J, Wan J, Zeng B, Wang Q, Song H, Qiu M. A scalable and quick-response software defined vehicular network assisted by mobile edge computing. *IEEE Commun Mag* (2017) 55:94–100. doi:10.1109/MCOM.2017.1601150

27. Huang M, Liu W, Wang T, Liu A, Zhang S. A cloud–mec collaborative task offloading scheme with service orchestration. *IEEE Internet Things J* (2020) 7:5792–805. doi:10.1109/JIOT.2019.2952767

28. Chen M, Hao Y. Task offloading for mobile edge computing in software defined ultra-dense network. *IEEE J Selected Areas Commun* (2018) 36:587–97. doi:10.1109/JSAC.2018.2815360

29. Wang C, Yu FR, Liang C, Chen Q, Tang L. Joint computation offloading and interference management in wireless cellular networks with mobile edge computing. *IEEE Trans Vehicular Techn* (2017) 66:7432–45. doi:10.1109/TVT.2017.2672701

30. Xu X, Zhang X, Liu X, Jiang J, Qi L, Bhuiyan MZA. Adaptive computation offloading with edge for 5g-envisioned internet of connected vehicles. *IEEE Trans Intell Transportation Syst* (2021) 22:5213–22. doi:10.1109/TITS.2020.2982186

31. Xu X, Jiang Q, Zhang P, Cao X, Khosravi MR, Alex LT, et al. Game theory for distributed iov task offloading with fuzzy neural network in edge computing. *IEEE Trans Fuzzy Syst* (2022) 30:4593–604. doi:10.1109/TFUZZ.2022.3158000

32. Zou Y, Lin L, Zhang L. A task offloading strategy for compute-intensive scenarios in uav-assisted iov. In: 2022 IEEE 5th International Conference on Electronic Information and Communication Technology (ICEICT); 21-23 August 2022; Hefei, China. IEEE (2022). p. 427–31.

33. Wei S, Zou Y, Zhang X, Zhang T, Li X. An integrated longitudinal and lateral vehicle following control system with radar and vehicle-to-vehicle communication. *IEEE Trans Vehicular Techn* (2019) 68:1116–27. doi:10.1109/TVT.2018.2890418

34. Hieu NQ, Hoang DT, Luong NC, Niyato D. irdrc: An intelligent real-time dual-functional radar-communication system for automotive vehicles. *IEEE Wireless Commun Lett* (2020) 9:2140–3. doi:10.1109/LWC.2020.3014972

35. Zhao J, Lou C, Hao H. Intelligent vehicle communication and obstacle detection based on millimetre wave radar base station. In: 2021 International Wireless Communications and Mobile Computing (IWCMC); June 28 2021; Harbin, China. IEEE (2021). p. 1818–22.

36. Zhang Q, Li Z, Gao X, Feng Z. Performance evaluation of radar and communication integrated system for autonomous driving vehicles. In: IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS); 10-13 May 2021; New York City, NY, USA. IEEE (2021). p. 1–2.

37. Zhou Z, Yu H, Xu C, Zhang Y, Mumtaz S, Rodriguez J. Dependable content distribution in d2d-based cooperative vehicular networks: A big data-integrated coalition game approach. *IEEE Trans Intell Transportation Syst* (2018) 19:953–64. doi:10.1109/tits.2017.2771519

38. Zhou Z, Ota K, Dong M, Xu C. Energy-efficient matching for resource allocation in d2d enabled cellular networks. *IEEE Trans Vehicular Techn* (2016) 66:5256–68. doi:10.1109/tvt.2016.2615718

39. Xu Q, Su Z, Zheng Q, Luo M, Dong B. Secure content delivery with edge nodes to save caching resources for mobile users in green cities. *IEEE Trans Ind Inform* (2017) 14: 2550–9. doi:10.1109/tii.2017.2787201

40. Liwang M, Wang J, Gao Z, Du X, Guizani M. Game theory based opportunistic computation offloading in cloud-enabled iov. *Ieee Access* (2019) 7:32551–61. doi:10.1109/ACCESS.2019.2897617

41. Ye H, Li GY, Juang B-HF. Deep reinforcement learning based resource allocation for v2v communications. *IEEE Trans Vehicular Techn* (2019) 68:3163–73. doi:10.1109/TVT.2019.2897134

42. Peng H, Shen XS. Ddpg-based resource management for mec/uav-assisted vehicular networks. In: 2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall); 4-7 October 2020; Victoria, Canada. IEEE (2020). p. 1–6.

43. Hu X, Wong K-K, Yang K. Wireless powered cooperation-assisted mobile edge computing. *IEEE Trans Wireless Commun* (2018) 17:2375–88. doi:10.1109/TWC.2018.2794345

44. Wei C-Y, Huang AC-S, Chen C-Y, Chen J-Y. Qos-aware hybrid scheduling for geographical zone-based resource allocation in cellular vehicle-to-vehicle communications. *IEEE Commun Lett* (2018) 22:610–3. doi:10.1109/LCOMM.2017.2784453

45. Hazarika B, Singh K, Biswas S, Li C-P. Drl-based resource allocation for computation offloading in iov networks. *IEEE Trans Ind Inform* (2022) 18:8027–38. doi:10.1109/TII.2022.3168292

46. Hazarika B, Singh K, Li C-P, Biswas S. Multi-agent drl-based computation offloading in multiple ris-aided iov networks. In: MILCOM 2022-2022 IEEE

Military Communications Conference (MILCOM); November-2 December 2022; Rockville, Maryland. IEEE (2022). p. 1–6.

47. Wang H, Li X, Ji H, Zhang H. Federated offloading scheme to minimize latency in mec-enabled vehicular networks. In: 2018 IEEE Globecom Workshops (GC Wkshps); 9-13 December 2018; Abu Dhabi, UAE (2018). p. 1–6.

48. Liu G, Wang W, Yuan J, Liu X, Feng Q. Elimination of accumulated error of 3d target location based on dual-view reconstruction. In: 2009 Second International Symposium on Electronic Commerce and Security. vol. 2; 22-24 May 2009; Nanchang City, China (2009). p. 121–4.

49. Sun Y, Guo X, Song J, Zhou S, Jiang Z, Liu X, et al. Adaptive learning-based task offloading for vehicular edge computing systems. *IEEE Trans Vehicular Techn* (2019) 68:3061–74. doi:10.1109/TVT.2019.2895593

50. Qi Y, Zhou Y, Liu Y-F, Liu L, Pan Z. Traffic-aware task offloading based on convergence of communication and sensing in vehicular edge computing. *IEEE Internet Things J* (2021) 8:17762–77. doi:10.1109/JIOT.2021.3083065

51. Song Z, Ma R, Xie Y. A collaborative task offloading strategy for mobile edge computing in internet of vehicles. In: 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC). vol. 5; March 15-17, 2024; Chongqing, China (2021). p. 1379–84.

52. Pham Q-V, Mirjalili S, Kumar N, Alazab M, Hwang W-J. Whale optimization algorithm with applications to resource allocation in wireless networks. *IEEE Trans Vehicular Techn* (2020) 69:4285–97. doi:10.1109/TVT.2020.2973294

53. Wang Y, Fang W, Ding Y, Xiong N. Computation offloading optimization for uav-assisted mobile edge computing: a deep deterministic policy gradient approach. *Wireless Networks* (2021) 27:2991–3006. doi:10.1007/s11276-021-02632-z

54. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, et al. Human-level control through deep reinforcement learning. *nature* (2015) 518:529–33. doi:10.1038/nature14236

55. Chen Z, Yin B, Zhu H, Li Y, Tao M, Zhang W. Mobile communications, computing, and caching resources allocation for diverse services via multi-objective proximal policy optimization. *IEEE Trans Commun* (2022) 70:4498–512. doi:10.1109/TCOMM.2022.3173005

56. Qiu C, Hu Y, Chen Y, Zeng B. Deep deterministic policy gradient (ddpg)-based energy harvesting wireless communications. *IEEE Internet Things J* (2019) 6:8577–88. doi:10.1109/JIOT.2019.2921159

57. Haarnoja T, Zhou A, Abbeel P, Levine S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: International conference on machine learning (PMLR); 10-15 July 2018; Stockholm, Sweden (2018). p. 1861–70.

58. Choi Y-S, Shirani-Mehr H. Simultaneous transmission and reception: Algorithm, design and system level performance. *IEEE Trans Wireless Commun* (2013) 12: 5992–6010. doi:10.1109/TWC.2013.101713.121152

59. Kwak J, Kim Y, Lee J, Chong S. Dream: Dynamic resource and task allocation for energy minimization in mobile cloud systems. *IEEE J Selected Areas Commun* (2015) 33: 2510–23. doi:10.1109/JSAC.2015.2478718

60. Ye T, Lin X, Wu J, Li G, Li J. Toward dynamic computation offloading for data processing in vehicular fog based f-ran. In: 2019 IEEE Fourth International Conference on Data Science in Cyberspace (DSC); June 25 2019; Hangzhou, China. IEEE (2019). p. 196–201.

61. Liu Y, Wang S, Huang J, Yang F. A computation offloading algorithm based on game theory for vehicular edge networks. In: 2018 IEEE International Conference on Communications (ICC); May 20-24, 2018; Kansas City, MO, USA (2018). p. 1–6.