Federated Reinforcement Learning-Empowered Task Offloading for Large Models in Vehicular Edge Computing

Huaming Wu, *Senior Member, IEEE*, Anqi Gu and Yonghui Liang

***Abstract*—Vehicular Edge Computing (VEC) has garnered substantial attention owing to its capacity to provide ample computational resources for computation-intensive tasks. However, how to flexibly allocate computing tasks within vehicles and efficiently manage the resources consumed by tasks has emerged as a challenge. To tackle this issue, this research advances the proposition of employing an auxiliary vehicle (AV) for task offloading and introduces a novel Auxiliary Vehicle Algorithm (AVA). AVA integrates both federated learning and multi-agent reinforcement learning to fully utilize computing resources in the vehicular environment, and simultaneously achieves task delay reduction, energy consumption minimization, and task completion rate augmentation. Moreover, we establish a federated learning framework to judiciously determine the proportion of resource allocation of AV through the implementation of inventive mechanisms. Experiment results validate that our approach not only leads to the improvement of key system performance indicators, but also ensures the comprehensive exploitation of the computing resources of mobile vehicles.**

***Index Terms*—Task offloading, Federated reinforcement learning, Edge computing, Vehicular Networks**

I. Introduction

THE rapid advancement of wireless communication technology, along with the emergence and progression of cloud computing, edge computing, and other technologies, has accelerated the development of the Internet of Vehicles (IoV), consequently fostering the advancement of autonomous driving and related technologies [1], [2]. During autonomous driving, a large amount of data is generated, such as road condition information, traffic flow, etc. The vehicle needs to process the above information and make corresponding decisions to control the vehicle to ensure a normal running process. How to process the above data in real-time is crucial for ensuring the safety of vehicle running. However, the escalating user demand and the high mobility of vehicles raise challenges to the development of autonomous driving technology, as they need to deal with large volumes of data within limited computing resources. Limited local resources of vehicles result in large delays and energy consumption when processing tasks involving large volumes of data [3].

Meeting the latency requirements for processing massive tasks of vehicles with limited computing resources has emerged as a critical challenge [4]. To this end, research has been conducted on task offloading for the Internet of Vehicles, that is, how to transfer large-scale tasks to edge or cloud computing to alleviate local computing pressure. Initially, cloud computing was introduced to help process local data to mitigate this issue. However, the proliferation of vehicle terminals, the expansion of task data, and the considerable distance between the cloud and the local terminals have led to several challenges, including heightened latency and network congestion [5], [6]. To address the limitations of cloud computing in IoV applications, Vehicular Edge Computing (VEC) has attracted great attention because it is much closer to vehicles and offers advantages in alleviating network congestion and latency [7]–[10].

There are studies on different performance indicators, such as latency, energy consumption, etc., in the decision-making of Iov tasks. When optimizing performance indicators, the high mobility of vehicles is also a critical factor to consider, as high-speed movement of vehicles can lead to unstable communication. The consideration of task dependencies and energy consumption of edge servers, along with the utilization of a table-based algorithm for making offloading decisions, has demonstrated its limited suitability for scenarios characterized by high dynamism and significant dimensionality [11]. To address task offloading challenges in complex and dynamic IoV scenarios, Yu *et al.* [12] proposed a task offloading approach that integrates an enhanced fuzzy C-means algorithm with Deep Q-Network (DQN). Clustering vehicles has been shown to reduce communication overhead and improve communication reliability [13]. Yang *et al.* [14] employed a vehicle clustering strategy to designate cluster head vehicles, aiming to ensure the stability of data transmission while optimizing energy consumption and delay. However, this approach failed to fully utilize the available computing resources of the vehicles. Raza *et al.* [15] proposed task offloading onto neighboring vehicles to mitigate time costs. However, it lacked specific vehicle selection strategies and did not consider energy consumption. TOERT [16] aims to eliminate redundancy in tasks while enhancing resource utilization. Nevertheless, it is worth noting that this approach solely concentrates on optimizing task completion rates and does not take into account metrics related to energy consumption. Feng *et al.* [17] focused on reverse offloading tasks to vehicles to reduce time-related costs incurred by servers. However, it ignored the consideration of energy consumption, and it was based on a greedy algorithm, which made it less adaptable to the intricacies and dynamism inherent in complex environments.

H. Wu, A. Gu and Y. Liang are with the Center for Applied Mathematics, Tianjin University, Tianjin 300072, China. E-mail: {whming, guanqi9_, liangyonghui}@tju.edu.cn.

(Corresponding author: Huaming Wu)

Due to cost and other reasons, the distance between edge servers is often large, and the communication range of edge servers is limited, which leads to queuing when facing a large number of simultaneously generated tasks, resulting in additional delay. To solve such problems, attention has been paid to idle vehicle resources. VEC servers can allocate their idle resources to nearby vehicles [18]. However, the rational selection of allocated vehicles and the proportion of allocated resources have not been fully discussed. This dramatically affects the efficiency of utilizing vehicle resources. In complex edge computing scenarios, federated reinforcement learning can be employed for task offloading to optimize returns [19]–[22]. Li *et al.* [23] explored the incentive mechanism of federated learning, which dynamically adjusts participants' weights based on statistical features to mitigate the impact of malicious users. This mechanism not only mitigates potential harm but also encourages active participation. Inspired by this, we apply this mechanism to the IoV, where vehicles assigned tasks are treated as participants, and the allocation proportions of resources as weights, effectively addressing the challenge of proportional allocation.

Vehicles, due to their high maneuverability, generate significant volumes of data during the driving process. The timely processing and feedback of this data are crucial determinants of decision-making accuracy within the vehicle networking environment. Large models enhance decision-making accuracy and timeliness in connected vehicle contexts by analyzing vast amounts of data and real-time information. In order to process large-scale data in time, we introduce large models. The term "large language model", or "large model" refers to machine learning models characterized by a substantial number of parameters and extensive training data [24]–[26]. Compared to traditional small models, large models, owing to their greater parameter count and more intricate structures, exhibit stronger learning, generalization, and computing capabilities, rendering them suitable for applications such as autonomous driving and road condition prediction in vehicular networking scenarios. Using large models in vehicular networking contexts yields more robust and accurate performance in complex driving environments [27].

In the decision-making stage of the task, traditional methods rely on predefined models and rules to achieve optimal solutions, lacking adaptability and primarily serving static problems. In contrast, multi-agent reinforcement learning defines a state space, selects appropriate actions, and sets rewards for executing actions, thereby continuously interacting with the connected vehicle environment to learn and make real-time task-offloading decisions in dynamic and complex environments. The incorporation of deep learning further enhances the capabilities of multi-agent reinforcement learning models, improving decision accuracy and enabling the handling of complex problems and long-term reward considerations that traditional methods cannot achieve. In comparison to single-agent reinforcement learning, multi-agent systems facilitate collaborative problem-solving, enhancing efficiency and generalization abilities. Therefore, in this paper, we employ multi-agent reinforcement learning as the underlying algorithm that makes the final task-offloading decisions.

For the above issues, this article is the first to study how to scientifically use the idle computing resources of surrounding vehicles to achieve low delay, low energy consumption, and high task-offloading completion rate when edge computing power is tight. To address the gaps in the above research, we present an innovative approach that integrates AVs. Our primary aim is to efficiently utilize spare computing resources of vehicles to alleviate the pressure of shortage of edge computing resources and optimize delay, energy consumption, and completion rate. The primary contributions of this paper are threefold:

- This paper introduces a federated learning incentive mechanism to solve the problem of offloading waiting caused by edge-computing resource constraints, by effectively utilizing nearby vehicle spare resources. Compared with single edge-computing task offloading schemes, this mechanism improves communication stability and releases the problem of long waiting time caused by the lack of edge computing resources.
- In order to solve the problems of data explosion and communication instability caused by high-speed vehicle movement, this paper combines deep reinforcement learning with federated learning and introduces large models to adapt to highly dynamic environments and process large amounts of data in a short time, ensuring accurate and timely vehicle decision-making.
- This paper proposes a total cost as a system evaluation indicator that integrates delay, energy consumption, and completion rate under the maximum allowable energy, delay threshold, and limitations on computing resources. This enables the optimization of overall performance based on user preferences while considering multiple indicators, making the evaluation system more comprehensive.

The remainder of this paper is organized as follows. Section II provides an overview of federated learning and the incentive mechanisms adopted in this study, and provides a review of the literature on DRL and large model algorithms. In Section III, we present the system model and define the task offloading model. Section IV offers a detailed exposition of the AVA algorithm. Simulation results, comparative experiments, and the evaluation of our algorithm are presented in Section V. Finally, Section VI concludes the paper.

II. Related Work

This section will discuss the research related to federated learning and incentive mechanisms, DRL, and large models, and elucidate the rationale behind certain innovative aspects highlighted in the paper.

A. Federated Learning-based Approaches

Federated learning is a machine learning algorithm that consists of two main components: the client and the server. The model training takes place on the client side, and then the model parameters from the clients are uploaded to the central server for aggregation [32]. Because the models on the client side do not share parameters during training, federated learning

TABLE I
COMPARISON OF THE PROPOSED SCHEME WITH PRIOR STUDIES

Perspective	Innovation Aspects	This Work	[28] 2020	[29] 2018	[30] 2022	[11] 2022	[14] 2022	[13] 2022	[15] 2020	[16] 2022	[19] 2023	[31] 2023
Problem background	Partial offloading	✓	✗	✗	✗	✗	✗	✗	✓	✓	✗	✓
	Auxiliary vehicle	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
	SCBS sever	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗
	Resource allocation	✓	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗
Algorithm	DRL	✓	✓	✓	✗	✗	✓	✗	✗	✗	✓	✗
	Joint optimization	✓	✓	✗	✗	✓	✓	✗	✓	✗	✗	✗
	Federated learning	✓	✗	✗	✓	✗	✗	✗	✗	✗	✓	✓
	Incentive mechanism	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
	Task priority	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Metrics	Completion rate	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗
	Latency	✓	✗	✓	✗	✓	✓	✗	✓	✗	✗	✓
	Energy	✓	✓	✓	✗	✓	✓	✓	✓	✗	✗	✗

is typically used to address privacy protection issues while also reducing the computing burden on the server side [33].

In federated learning, incentive mechanisms encourage clients to participate and share their models. This is because when clients participate in federated learning, they expend their resources, and the incentive mechanism provides model-based compensation to the clients to promote their involvement. In [34], the incentive mechanism of federated learning is used in the edge-cloud collaborative scenario of blockchain to compensate participants to balance system overhead and model performance. In mobile edge computing, federated learning can also be used to enhance privacy and decentralization, and [35] discussed several significant issues that need to be addressed, including the incentive mechanism. Therefore, it can be inferred that when a task is offloaded to a vehicle in transit, the computing resources the vehicle provides represent a cost to its resources. In order to allocate resources reasonably, we can incorporate the incentive mechanism of federated learning here to fairly evaluate the contribution of the vehicle and provide appropriate rewards to encourage the vehicle to contribute more resources for task offloading.

B. DRL-based Approaches

DRL combines deep learning and reinforcement learning to help learn optimal strategies through environment interaction [36]. Its application in automotive networks is widespread due to the complex and dynamic nature of vehicular environments, enabling intelligent agents to make adaptive decisions regarding task offloading and resource allocation [37]. Task offloading is an important issue in in-vehicle networks and is frequently tackled using DRL techniques for decision-making, including multi-agent DRL methods [38].

In [39], DRL is applied within the incentive mechanism of federated learning in Intelligent Cyber-Physical Systems to offer long-term incentives for model participants operating in dynamic environments. The multi-agent game is formulated as a Markov decision process to devise allocation strategies efficiently. This inspired employing DRL as the underlying algorithm for federated learning, enabling real-time dynamic decisions for task offloading in vehicular networking.

In recent years, OpenAI's GPT family has attracted widespread attention, such as ChatGPT [40]. This is a large language model with many parameters and data that provides strong learning capabilities and the ability to solve complex problems [26]. In the highly dynamic and complex environment of connected vehicles, ordinary models may struggle to ensure the accuracy of decisions, potentially impacting critical issues such as vehicle safety [41]. Therefore, large models can be introduced into connected vehicles to ensure the accuracy of decisions. However, the accuracy of large models comes at the cost of high computing resources, which is difficult for ordinary vehicles to satisfy locally [42]. Hence, we can alleviate this burden by leveraging task offloading to nearby edge or cloud computing resources [43], which offer significantly enhanced computing capabilities. The task offloading process can employ appropriate decision algorithms to assign large model tasks for processing either in the cloud or at the edge, aiming to mitigate the high delay and low-quality issues arising from local computing resource constraints [44].

Numerous delay-sensitive applications demand substantial computational resources, such as autonomous driving and vehicle queuing [45]. It is necessary to offload and deploy large model application tasks within the IoV onto nearby edge servers for computing processing. However, due to the significant computing requirements of large model applications, deploying them extensively on edge servers is not feasible due to the high configuration costs. Therefore, vehicles equipped with high computing capabilities emerge as alternative offloading targets, leading to the issue of vehicle selection and resource allocation. This paper will elaborate on the specific implementation of such offloading schemes.

C. DRL under Federated Learning Framework

There has been growing interest in combining federated learning frameworks with reinforcement learning decision-making to address various challenges in distributed systems.

For instance, Yu *et al.* [46] introduced the I-UDEC framework, which enables heterogeneous resource allocation and hybrid computing offloading. They proposed a DRL approach to optimize delay, considering varying levels of delay sensitivity. Additionally, federated learning was employed to ensure the security of private data. Wu *et al.* [47] introduced an asynchronous federated learning scheme to address local model failure due to high vehicle mobility, which can lead to inaccuracies in global models. They also developed a collaborative caching scheme using an adversarial DQN algorithm to minimize content transmission delay. In [48], DRL is employed for resource allocation to reduce the total delay and

energy consumption of federated learning. However, the use of federated learning here is only for privacy security. Wang *et al.* [49] applied federated learning to protect data privacy in the IoV, and used DRL for wireless network selection to improve learning performance. Al-Maslamani *et al.* [50] integrated DRL as a reputation model within the edge server of federated learning. This integration aimed to enhance the accuracy of the global model while also strengthening data privacy measures.

Compared to the contributions of the studies above, this paper offers several advantages. Firstly, while previous articles investigate the offloading of tasks from mobile devices, they do not extend their theories to address task offloading in the context of IoV. In contrast, our paper leverages federated learning and reinforcement learning to tackle task-offloading challenges within IoV scenarios. Secondly, whereas previous studies primarily aim to optimize delay, our paper takes a broader approach by introducing a comprehensive metric capable of concurrently measuring delay, energy consumption, and completion rate. Lastly, in terms of algorithms, previous studies primarily utilize federated learning frameworks for privacy protection. In contrast, our study focuses on leveraging federated learning incentive mechanisms to optimize the resource allocation ratio of auxiliary vehicles to enhance the contribution ratio of vehicles to other vehicle offloading tasks and provide additional resources for vehicle networking task offloading. Overall, our paper presents innovative applications by integrating federated learning and DRL within the context of connected vehicles.

III. System Model and Problem Formulation

A. Network Model

Fig. 1 illustrates the network model, which comprises a macro cell base station (MCBS), multiple small cell base stations (SCBSs), and a fleet of vehicles denoted as i ($i \in N$) [16]. The SCBSs and vehicles are situated within the communication range of the central MCBS, and the SCBSs have a coverage radius of 200 meters. The SCBSs are connected to the MCBSs through wired cables. Each vehicle is equipped with a single antenna and has communication and computing capabilities. Vehicles communicate with each other using radio technology, whereas the SCBSs are connected to the MCBSs through wired cables.

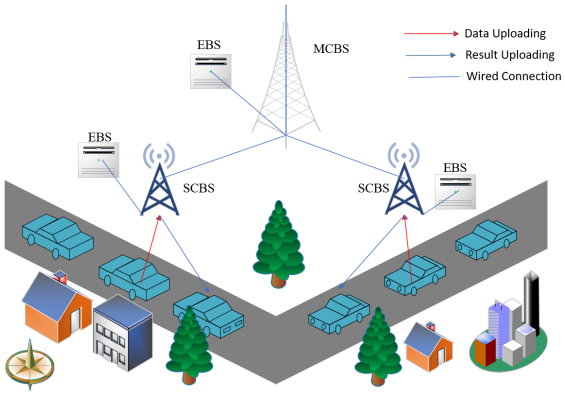

Fig. 1. System model

We assume that each vehicle is assigned a task, which can be divided into M subtasks. The amount of CPU cycles required to complete subtask j ($j \in M$) of vehicle i is represented as $B_{i,j}$. The priority of subtasks is $pref(n_j)$. The predecessor subtasks need higher priorities relative to their successor subtasks. The CPU frequency associated with vehicle i and the SCBS is denoted as f_{vi} and f_S, respectively. However, offloading all tasks to the SCBS would inevitably lead to network congestion and a surge in time delay. Consequently, this paper focuses on reducing system costs by leveraging the inherent resources of each vehicle. The decision variable is defined as a three-dimensional matrix Y of dimensions $N \times M \times 3$. Each element $Y_{i,j,k} \in [0,1]$ denotes the offloading decision for the j-th subtask of vehicle i, where $i \in N$, $j \in M$, and $k \in [1,2,3]$. When $k = 1$, the j-th subtask for vehicle i is processed locally, while $k = 2$ or $k = 3$ indicates offloading to the SCBS or Auxiliary Vehicle (AV), respectively. The symbols and their detailed meanings can be found in Table II.

TABLE II
Notations and Definitions

Notation	Definition
N	Number of vehicles
M	Number of subtasks
$T_{i,j}$	The j^{th} subtask of the i^{th} vehicle
$pre_{i,j}$	Priority of subtasks
v_i	The velocity of vehicle i
$l_i(t)$	Coordinates of the position of the i^{th} vehicle
AV	Auxiliary vehicle
D_i	The average distance of v_i from other vehicles
p^S	The transmit power of SCBS
$Y_{i,j,k}$	The offloading decision for subtask $T_{i,j}$
f_{v_i}	Computing power of v_i
P_i	Transmitting power of vehicle i
$B_{i,j}$	The number of CPU required to execute the task
$Data_{i,j}^{up}$	Amount of uploaded data
$Data_{i,j}^{down}$	Amount of downloaded data
$R_{v_i,A}^{V2V}$	Transmitting power between vehicle and AV
$W_{v_i,A}^{trans}$	Transmitting energy consumption between V2V
$T_{i,j}^{wait}$	Task $T_{i,j}$'s waiting time for all predecessor tasks
r_i	Task completion rate of vehicle i
T_{f_i}	An indicator on whether i^{th} task has been completed
S_i	The distance traveled by vehicles covered by SCBS
B_{V2I}	Bandwidth of task uploaded to SCBS
$t_{i,S}^{V2Ir}$	Time within SCBS's communication range of vehicle i
$d_{v_i,A}^{rest}(t)$	The remaining distance of vehicle i from the AV range

We can calculate the distance of vehicles operating within the coverage area of SCBS as follows [15]:

$$S_i = 2\sqrt{r^2 - e^2}, \tag{1}$$

where r is the radius of the SCBS and e is the vertical distance from the SCBS to the road surface.

The duration for which the vehicle remains within the coverage area of the SCBS is given by:

$$t_{i,S}^{V2Ir} = \frac{S_i}{v_i}, \tag{2}$$

where v_i represents the velocity of vehicle i.

B. Auxiliary Vehicle

We introduce an Auxiliary Vehicle (AV) into the problem scenario to assist SCBS in computing offloading tasks for other vehicles. Without loss of generality, the AV may also equip tasks that need to be computed. Therefore, we introduce a variable R to represent the proportion of the AV's resources used to compute offloading tasks for other vehicles.

To effectively conduct computations, it is essential to select a suitable vehicle as an AV based on the criterion CS_i. Considering both the distance and computing capacity of the vehicles, this can be formulated as $\boldsymbol{\mathcal{P}_1}$:

$$(\boldsymbol{\mathcal{P}_1}) \ \max: \quad CS_i = \alpha D_i + \beta f_{vi} \tag{3}$$

$$\text{s.t.}: \quad \alpha + \beta = 1, \tag{4}$$

where α and β represent the weighting factors for distance and computational capabilities, respectively, determined based on user requirements.

The average distance between vehicle i and other vehicles is given by:

$$D_i = \frac{1}{N-1} \sum_{j=1, j\neq i}^{N} \|l_i(t) - l_j(t)\|, \tag{5}$$

where the variables l_i and l_j represent the coordinates of vehicle i and vehicle j, respectively. The distance between the two vehicles is obtained through the Euclidean distance formula. Then, the total distance between vehicle i and all other vehicles is divided by the total number of vehicles minus one to obtain the average distance between vehicle i and other vehicles.

We quantify the computing resources utilized by the AV for the computation of tasks assigned to other vehicles as $f_{AV} = R \times f_A$, where f_A represents the computing capacity of the AV.

C. Communication Model

The communication model encompasses two distinct modes: vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication [51]. We employ Dedicated Short Range Communication (DSRC) for V2V communication and LTE-Advanced (LTE-A) for V2I communication. Orthogonal frequency is typically utilized for V2V communication, while we assume a Rayleigh fading channel for the channel model. The Rayleigh fading channel model is frequently employed to characterize multipath fading in wireless communication. This model, rooted in the Rayleigh distribution, posits that signals traverse multiple reflective paths before reaching the receiver, leading to fluctuations in signal strength across time and space. In the context of VEC environments characterized by vehicle mobility and dynamic communication, the Rayleigh fading channel model remains applicable for describing signal multipath effects.

According to [52], we can determine the path loss of communication of V2V and V2A as follows, respectively:

$$los_i^{V2V} = 10^{\frac{-63.3+17.7\lg(d_{i,A}(t))}{10}}, \tag{6}$$

$$los_i^{V2I} = 10^{\frac{-63.3+17.7\lg(d_{i,S}(t))}{10}}, \tag{7}$$

where $0 \leq d_{i,A} \leq C_{limit}$ and $d_{i,S}(t)$ represent the distance between vehicle i and vehicle A, and between vehicle i and the SCBS at time t, respectively. These distances can be expressed as follows:

$$d_{i,A}(t) = \|l_i(t) - l_A(t)\|, \tag{8}$$

$$d_{i,S}(t) = \|l_i(t) - l_S(t)\|, \tag{9}$$

where $l_i(t)$, $l_A(t)$ and $l_s(t)$ denote the positions of the vehicle i, vehicle A, and the SCBS at time t, respectively. C_{limit} represents the maximum communication range between vehicles.

The uploading transmission rates from vehicle i to the AV and the SCBS are given by [17]:

$$R_{v_i,A}^{V2V} = B_{V2V} \log_2 \left(1 + \frac{P_i los_i^{V2V} \|h^2\|}{N_0}\right), \tag{10}$$

$$R_{v_i,S}^{V2I} = B_{V2I} \log_2 \left(1 + \frac{P_i los_i^{V2I} \|h^2\|}{N_0}\right), \tag{11}$$

where B_{V2V} denotes the bandwidth between vehicles, and B_{V2I} denotes the bandwidth between the vehicle and SCBS. P_i is the transmitting power of the vehicle's onboard device. N_0 represents the power of white Gaussian noise. The channel fading coefficient h follows a Rayleigh distribution, and the path loss is characterized by d^σ, where σ denotes the path loss exponent.

Similarly, the downloading transmission rates of AV and SCBS to the vehicle i are given by:

$$R_{A,v_i}^{V2V} = B_{V2V} \log_2 \left(1 + \frac{P_A los_i^{V2V} \|h^2\|}{N_0}\right), \tag{12}$$

$$R_{S,v_i}^{V2I} = B_{V2I} \log_2 \left(1 + \frac{P_S los_i^{V2I} \|h^2\|}{N_0}\right), \tag{13}$$

where P_S and P_A represent the transmitting power of the SCBS and the AV, respectively. The AV's CPU frequency and the CPU frequency allocated to the other vehicles by the AV are denoted by f_i and f_A (cycles per second), respectively.

The average uploading transmission rates of the vehicle i to AV and SCBS are given by [15].

$$\overline{R_{vi,A}^{V2V}} = \frac{\int_0^{t_{i,A}^{VVr}} R_{vi,A}^{V2V}(t)\, dt}{t_{i,A}^{V2Ir}}, \tag{14}$$

$$\overline{R_{vi,S}^{V2I}} = \frac{\int_0^{t_{i,S}^{VVr}} R_{vi,S}^{V2I}(t)\, dt}{t_{i,S}^{VIr}}. \tag{15}$$

Similarly, the average downloading transmission rates for V2V and V2I communication channels are provided by [15].

$$\overline{R_{vi,A}^{V2V}} = \frac{\int_0^{t_{i,A}^{VVr}} R_{vi,A}^{V2V}(t)\, dt}{t_{i,A}^{VVr}}, \tag{16}$$

$$\overline{R_{V2I}^{vi,S}} = \frac{\int_0^{t_{i,S}^{VIr}} R_{V2I}^{vi,S}(t)\, dt}{t_{i,S}^{VIr}}. \tag{17}$$

D. Task Offloading Decision

As to the task offloading decision, we can divide it into three parts according to different processors.

1) Local Computing: $y_{i,j,1} = 1$ means the j^{th} subtask of the i^{th} vehicle is computed locally. The delay and energy consumption associated with the local computation of the j^{th} subtask for the i^{th} vehicle are expressed as:

$$T_{i,j}^{l} = \frac{B_{i,j}}{f_{v_i}}, \tag{18}$$

$$E_{i,j}^{l} = W_C^{l} T_{i,j}^{l} = \mu f_{v_i} \nu^2 \frac{B_{i,j}}{f_{v_i}}. \tag{19}$$

where W_C represents the energy consumption of the CPU, ν represents the effective capacitance coefficient for each CPU cycle, and v corresponds to the working voltage. The energy consumption mentioned here specifically refers to the local computing energy of vehicles, which can be calculated by $W_C = \mu f \nu^2$.

2) AV Computing: $y_{i,j,3} = 1$ indicates that the j^{th} subtask of the i^{th} vehicle is offloaded to the AV for computation, and the processing results are subsequently sent to the corresponding vehicles.

The delay associated with uploading the subtask data to the AV and downloading it back to the vehicle is defined as follows [15]:

$$T_{i,j}^{up\,to\,A} = \frac{Data_{i,j}^{up}}{R_{v_i,A}^{V2V}}, \tag{20}$$

$$T_{i,j}^{down\,to\,v_i} = \frac{Data_{i,j}^{down}}{R_{A,v_i}^{V2V}}. \tag{21}$$

where $Data_{i,j}^{up}$ and $Data_{i,j}^{down}$ represent the sizes of the uploading and downloading data volumes for the j^{th} subtask of the i^{th} vehicle, respectively.

The time required for processing tasks on the AV is given by:

$$T_{i,j}^{Aexe} = \frac{B_{i,j}}{f_A}. \tag{22}$$

The total delay of the j^{th} subtask of the i^{th} task with AV computing comprises the task-uploading time, task-downloading time, and processing time.

$$T_{i,j}^{TA} = T_{i,j}^{up\,to\,A} + T_{i,j}^{Aexe} + T_{i,j}^{down\,to\,v_i}. \tag{23}$$

Similarly, the total energy consumption comprises the energy used for uploading tasks, the energy used for processing tasks by the AV, and the energy spent on result retrieval.

$$E_{i,j}^{TA} = E_{v_i,A}^{trans} + E_{i,j}^{Aexe} + E_{i,j}^{down\,to\,v_i}, \tag{24}$$

where the required energy to transmit the task from vehicle i v_i to AV is given by;

$$E_{v_i,A}^{trans} = W_{v_i,A}^{trans} T_{i,j}^{up\,to\,A}. \tag{25}$$

The required energy that takes for the task to be executed on AV is:

$$E_{i,j}^{Aexe} = W_A^{exe} T_{i,j}^{Aexe} = \mu_A f_A \nu_A^2 T_{i,j}^{Aexe}. \tag{26}$$

The required energy for downloading the task to the original vehicle is:

$$E_{i,j}^{down\,to\,v_i} = W_{A,v_i}^{trans} T_{i,j}^{down\,to\,v_i} = \mu_A f_A \nu_A^2 T_{i,j}^{down\,to\,v_i}. \tag{27}$$

3) SCBS Computing: $y_{i,j,2} = 1$ means the SCBS helps to compute the j^{th} subtask of the i^{th} vehicle and send the results back.

The total delay of the i^{th} task with SCBS computing comprises several components, including the delay of uploading the task data to the SCBS and receiving the results, the processing time of the task on the SCBS, and the executing time of all predecessor tasks for the j^{th} subtask of the i^{th} vehicle. Where the required delay to transmit the task from vehicle i to SCBS is

$$T_{i,j}^{up\,to\,V2I} = \frac{Data_{i,j}^{up}}{R_{V2I}}. \tag{28}$$

The required energy that takes for the task to be executed on AV is

$$T_{i,j}^{Sexe} = \frac{B_{i,j}}{f_s}. \tag{29}$$

The required waiting time for the task is

$$T_{i,j}^{wait} = \max_{k \in pred(T_{i,j})} T_{i,k}^{Sexe}, T_{i,j}^{down\,to\,v_i} = \frac{Data_{i,j}^{down}}{R_{S,v_i}^{V2I}}, \tag{30}$$

where $T_{i,k}^{Sexe}$ represents the task processing time of the j^{th} subtask of the i^{th} vehicle on the SCBS.

$$T_{i,j}^{TS} = T_{i,j}^{up\,to\,V2I} + T_{i,j}^{Sexe} + \max_{k \in pred(T_{i,j})} T_{i,k}^{Sexe} + T_{i,j}^{wait}. \tag{31}$$

The total energy consumption consists of the energy used for uploading tasks, the energy expended during task execution, and the energy required for downloading the results.

The energy required to transmit the task from vehicle i to SCBS is given by:

$$E_{v_i,S}^{trans} = W_{v_i,S}^{trans} T_{i,j}^{up\,to\,V2I}. \tag{32}$$

The energy required for the task to be executed on the SCBS is given by:

$$E_{i,j}^{Sexe} = W_S^{exe} T_{i,j}^{Sexe} = \mu_S f_S \nu_S^2 T_{i,j}^{Sexe}. \tag{33}$$

The energy required for downloading the subtask data to the vehicle from the SCBS is given by:

$$\begin{aligned} E_{i,j}^{Strans} &= W_S^{trans} T_{i,j}^{up\,to\,V2I} \\ &= \mu_S f_S \nu_S^2 T_{i,j}^{up\,to\,V2I}, \end{aligned} \tag{34}$$

where the transmitting energy consumption of V2I is W_S^{trans}.

Therefore, the total energy consumption is given by:

$$E_{i,j}^{TS} = E_{v_i,S}^{trans} + E_{i,j}^{Sexe} + E_{i,j}^{Strans}. \tag{35}$$

E. Task Completion Rate

To ensure the successful completion of a task, we introduce the concept of task completion rate. Task completion is contingent upon the successful execution of the last subtask. If the previous subtask fails, it implies that the result fails to be transmitted to the vehicle before it exits the communication range of the SCBS or AV. The total number of completed tasks is denoted as

$$T_f = \Sigma_{i=1}^{N} T_{fi}, \tag{36}$$

where T_{fi} indicates whether the i^{th} task has been completed.

If $t_{i,S}^{VIr} - t_{to}^{M-1} \geq T_{i,M}^{toS} \geq 0$ and $t_{i,A}^{V2Ir} - t_{to}^{M-1} \geq T_{i,j}^{toA} \geq 0$, then T_{f_i} will be set to 1; otherwise, it is set to 0. This condition ensures that until vehicle i completes $M-1$ tasks, it remains within the communication range of the AV or SCBS, and the remaining stay time is sufficient for the completion of the task. If either of the two inequality conditions is satisfied, the task is considered completed.

Subsequently, the task completion rate of vehicle i is calculated as follows:

$$r_i = \frac{T_f}{N}, \tag{37}$$

which represents the percentage of successful tasks to the total number of tasks.

F. Problem Formulation

The total delay of the system is calculated as follows:

$$T_{to} = \sum_{i=1}^{N}\sum_{j=1}^{M}(Y_{i,j,1}T_{i,j}^{l} + Y_{i,j,2}T_{i,j}^{TA} + Y_{i,j,3}T_{i,j}^{TS}). \tag{38}$$

The total energy consumption of the system is calculated as follows:

$$E_{to} = \sum_{i=1}^{N}\sum_{j=1}^{M}(Y_{i,j,1}E_{i,j}^{l} + Y_{i,j,2}E_{i,j}^{TA} + Y_{i,j,3}E_{i,j}^{TS}). \tag{39}$$

We aim to minimize the total delay and energy consumption while maximizing task completion. This can be regarded as a multi-objective joint optimization problem, and we convert it into a single-objective optimization problem via weighted summation. The objective function of the problem is defined as the total cost to be optimized, represented as $\boldsymbol{\mathcal{P}_2}$, which consists of the weighted sum of total delay, total energy consumption, and the reciprocal of task completion rate. This way, when optimizing the total cost, it can comprehensively reduce the total delay and total energy consumption, and improve the task completion rate, achieving multi-objective joint optimization. This can be formulated as follows:

$$\begin{aligned}
(\boldsymbol{\mathcal{P}_2}) \ \min: \ & \Lambda_1 \times T_{to} + \Lambda_2 \times E_{to} + \Lambda_3 \times 1/r_i \\
\text{s.t.}: \ & 0 \leq f_i \leq f_i^{max}, i \in N && \text{(40a)} \\
& 0 \leq f_A \leq f_A^{max} && \text{(40b)} \\
& 0 \leq f_s \leq f_s^{max} && \text{(40c)} \\
& \sum_{k=1}^{3} Y_{i,j,k} = 1, i \in N, j \in M && \text{(40d)} \\
& Y_{i,j,k} \in \{0,1\} && \text{(40e)} \\
& \max\{T_i^l, T_i^{V2V}, T_i^S\} \leq t_i^{max} && \text{(40f)} \\
& \max\{E_i^l, E_i^{V2V}, E_i^S\} \leq E_i^{max} && \text{(40g)} \\
& 0 \leq P_i \leq P_i^{max}, i \in N, && \text{(40h)}
\end{aligned}$$

where Λ_1, Λ_2, and Λ_3 represent the weighting factors of delay, energy consumption, and task completion, respectively. These values can be determined based on user preferences. We employ the Analytic Hierarchy Process [53] to qualitatively and quantitatively analyze the determination of weighting factors. The three factors mentioned in the text are used as row and column labels to construct a 3×3 symmetric judgment matrix. Then, users assign scores based on the importance level, utilizing a predefined scale table. Finally, column normalization, row summation, and consistency checks are performed. The final output that passes the consistency check is used as the weighting factor. If the objective is to enhance a specific indicator individually, one can adjust the weight factor of the remaining indicators within the overall cost framework to zero. In $\boldsymbol{\mathcal{P}_2}$, Constraints (a-c) represent the maximum computing resource limit for vehicles and SCBS. Constraints (d-e) pertain to vehicular decision constraints for the j^{th} subtask of the i^{th} vehicle, ensuring that a subtask can only be executed on a single processor. Constraint (f-g) limits the maximum tolerable delay and energy consumption for each mode. Constraint (h) denotes the maximum transmitting power.

$\boldsymbol{\mathcal{P}_2}$ is a challenging Mixed-Integer Nonlinear Programming (MINLP) problem, generally NP-hard. It is challenging to obtain the solutions effectively because of their non-convexity. To address this, we decompose the original problem into two parts and design algorithms to solve each corresponding part.

IV. AVA Algorithm

A. Federated Learning on Server

To effectively motivate the chosen AV to allocate its computing resources, we introduce the incentive mechanism of federated learning. This mechanism operates as a reward or punishment system within the federated learning framework.

In the incentive mechanism of federated learning, reputation value is used to measure the level of contribution and trustworthiness of participants. It is typically derived from the participant's contribution level and is essential in the incentive mechanism. Reputation is employed to penalize malicious participants and reward cooperative ones. In our context, we will employ this incentive mechanism to motivate the AVs to actively contribute their computing resources, thereby alleviating the strain on the SCBS terminal or local computing. We define the reputation in federated learning as:

$$Reputation = \left(G_T - \frac{\sum_{i=1}^{T-1} G_i}{T-1}\right) \times R, \tag{41}$$

where T is the epoch, and G_i represents the total system cost for the i^{th} round. R represents the allocation ratio of resources the current assisting vehicle utilizes to aid other vehicles in task offloading. The greater the proportion of resources contributed by the AV to assist other vehicles in task computation, the more significant its contribution to other vehicles. Meanwhile, we must consider the AV's contribution to optimizing the objective function G. To achieve a more balanced performance, we incorporate the difference between the total cost of the current round and the average cost of the previous $T-1$ rounds into the contribution metric. A larger difference implies a greater contribution in the current round.

Regarding the setting of rewards, we introduce the priority of vehicles and set it as pre_i, indicating the priority of task execution for the $i-th$ vehicle. When the vehicles offload their tasks to the server for execution, they are sorted according to

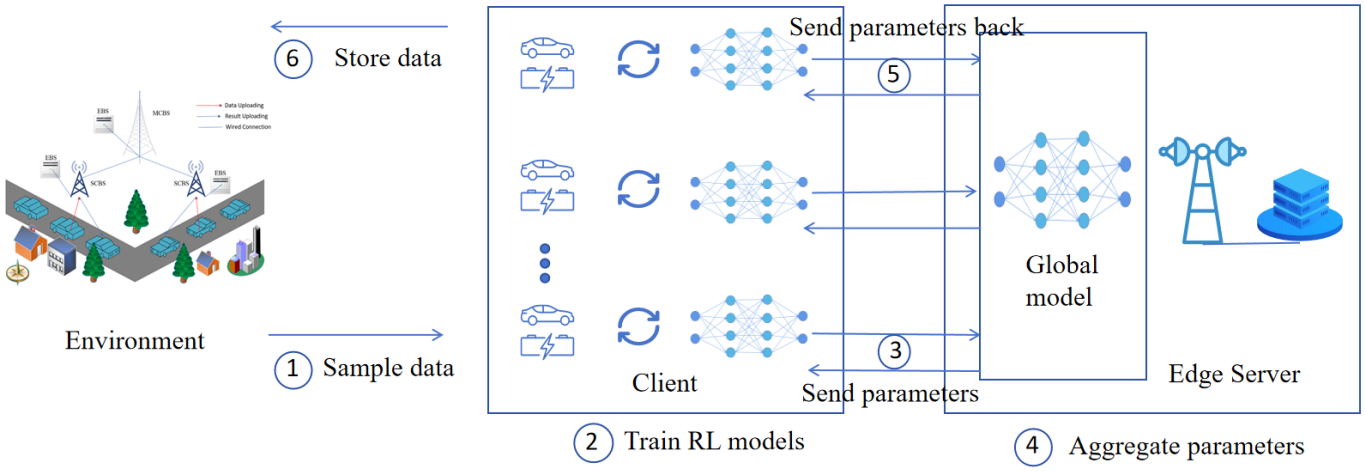

Fig. 2. The architecture of federated reinforcement learning-empowered task offloading in VEC.

priority, and the tasks of vehicles with the higher priority are executed first. We update the priority as follows:

$$pre_i = \begin{cases} pre_i, & pre_i < pre_i - PID \\ pre_i - PID, & i = n_A \\ pre_i + 1, & i = pre_i > pre_i - PID \end{cases} \quad (42)$$

where PID $(0 \leq PID \leq pred(A))$ represents the degree to which the priority is increased, and PID is a natural number. The priority among vehicles is initially randomized. Rewards are then adjusted according to variations in the reputation value of the AV, particularly focusing on increments in priority.

Algorithm 1 FedRep

Initialize $\phi^0, h_1^0, \cdots, h_n^0$.
for $t = 1, 2, \cdots, T$ **do**
  Server receives clients' parameters
  Server sends current representation ϕ^0 to client
  for $k = 1, 2, \cdots, K$ **do**
    Receive $h_i^{t,k}$ from client
    Implementing incentive mechanism
    Aggregate $h_i^{t,k} \leftarrow \sum_{j=1}^{N} h_{i,j}^{t,k}$
    Send $h_i^{t,k}$ to client
  end for
end for

Fig. 2 shows the overall architecture of federated reinforcement learning-empowered task offloading in VEC. The detailed steps are as follows:

- **Step 1:** Each agent selects an action from the environment's experience pool based on selecting strategies.
- **Step 2:** The agents in the client train local models through multi-agent reinforcement learning.
- **Step 3:** The parameters of the local model are passed to the global model.
- **Step 4:** Aggregate parameters of the local model and incentivize the increase or decrease of R.
- **Step 5:** The global model transmits the parameters back to the local model.
- **Step 6:** The data is stored in the experience pool.

The specific joint optimization algorithm is outlined in Algorithm 1. Initially, we initialize the parameters of the server model. Subsequently, clients receive data to train the network parameters of DRL. Following this, all clients upload their parameters to the server for aggregation. The server then employs incentive mechanisms to dynamically adjust resource allocation for AV, aiming to minimize the total cost.

For federated learning, a critical challenge is to involve AVs in the learning process effectively. The introduction of incentive mechanisms facilitates the sharing of benefits between federated and assisted vehicles during the reinforcement learning process, involving the participation of assisted vehicles. However, to ensure the long-term stability of federated learning and enhance the availability of computing resources of AVs to perform offloading tasks from other vehicles, this paper adopts priority as an incentive for AVs. Moreover, it dynamically allocates AVs' computing resources to other vehicles by continuously maximizing reputation values to incentivize participation.

Algorithm 2 MADDPG-based client training

Input: vehicle positions, computing resources of vehicles and SCBS.

Initialize the weights of target, eval networksθ and $\theta^{'}$ with random number, and the replay buffer D.
for episode = $1, \cdots, M$ **do**
  Receive initial state S;
  Initialize a random action;
  for vehicle $i = 1, \cdots, N$ **do**
    Execute actions a_i and obtain new state $s_i^{'}$;
    Obtain the reward R of vehicle i based on Eq. (45);
    Obtain the action A, new state $S^{'}$;
    Store (S, A, $S^{'}$, R) in replay buffer D;
  end for
  for vehicle $i = 1, \cdots, N$ **do**
    Sample a random mini-batch of samplesfrom D;
    Update the eval network by minimizing the loss function;
  end for
  Update the target network parameters of each vehicle i: $\theta_i^{'} \leftarrow \delta\theta_i + (1-\delta)\,\theta_i^{'}$ at every C steps.
end for

B. Federated Learning on Client

We employ the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) on the client to tackle the task offloading problem. We formulate this problem as a Markov Decision Process (MDP).

1) State Space: We define the system's state space as $s(t)$, which encompasses the computing capacity of vehicles and SCBS, as well as the positions of vehicles.

$$s(t) = \{x_1(t), \cdots x_i(t), C_1(t), C_2(t) \cdots C_i(t), C_b(t)\}, \quad (43)$$

where $x_i(t)$ represents the position of vehicle i at time t, while $C_i(t)$ and $C_b(t)$ denote the computing capacity of vehicle i and SCBS, respectively. This enables rational decision-making based on real-time vehicle position and current resources of vehicles and SCBS.

2) Action Space: The action space $a(t)$ indicates whether to retain the task or offload it to the other two servers, expressed as:

$$a(t) = \{a_1(t), a_2(t), \cdots a_i(t), \cdots, a_N(t)\}, \quad (44)$$

where $a_i(t)$ represents the decision of the i^{th} task. In particular, $a_i(t) = 1$, 2 or 3 represents local processing, SCBS processing, and AV processing, respectively. Since the action vector is discrete, DQN can be utilized for action selection.

3) Reward Function: We aim to minimize the system cost (G) while considering real-world constraints, with the reward set as the negative value of G.

$$r_{(t)} = -G. \quad (45)$$

This approach is adopted because the objective is to minimize the total cost, whereas the goal of reinforcement learning is to maximize long-term expected rewards.

The algorithmic process for the MADDPG-based client training is as described in Algorithm 2.

C. Computing Complexity

In Alg. 1, there is a nested loop. The complexity can be expressed as $\mathcal{O}(TK)$. The complexity of Alg. 2 is determined by another nested loop, and it amounts to $\mathcal{O}(MN)$. This loop corresponds to the number of episodes in the interaction between the agent and the environment, and the number of times the vehicles are traversed, respectively.

V. Performance Evaluation

A. Parameter Settings

In the considered VEC network scenario, we assume a total of N vehicles, along with one SCBS. Each vehicle is assigned a task that can be further segmented into M subtasks, and the interdependence of these subtasks is illustrated in Fig. 2. The weight values α and β are both set to 0.5. The channel model adheres to the Rayleigh fading model with a path loss exponent $\sigma = 2$. White noise N_0 is specified as 3×10^{-13}, and the bandwidth for V2V (B_{V2V}) and V2I (B_{V2I}) communication is set to 1 MHz. Key parameter values include $C_{\text{limit}} = 150$ m [54], $f_{\text{vi}} = \left[10^6, 2 \times 10^8\right]$ cycles/s, $P_i = P_t = 1.3$ W [55], and $P_{\text{SCBS}} = 40$ W. The capacitance coefficients $\mu = 10^{-28}$ [56], $\nu = \nu_{\text{vi}} = \nu_{\text{AV}} = 12$ V, $\nu_{\text{SCBS}} = 220$ V, and maximum time delay and energy consumption are specified as $f_{\text{s}}^{\max} = 8 \times 10^8$ c/s [57] and $E_i^{\max} = 3$ J. For the values of other parameters, please refer to Table III.

TABLE III
Simulation Parameters

Parameter	Value	Parameter	Value
α	0.5	β	0.5
σ	2	N_0	3×10^{-13}
B_{V2V}	1 MHz	B_{V2I}	1 MHz
C_{limit}	150 m	f_{vi}	$\left[10^6, 2 \times 10^8\right]$ cycles/s
P_{SCBS}	40 W	μ	10^{-28}
ν	$12V$	ν_{SCBS}	220 V
$E_i^{\max}$	3 J	$Data_{i,j}^{up}$	$[1, 10]$ MB
P_i	1.3 W	$f_s^{\max}$	8×10^8 cycles/s

B. Baselines

The following schemes are employed as baseline algorithms in the experiments to compare with the algorithm proposed in this paper.

- **Full local executing (FL)**: All the tasks are executed locally
- **Full SCBS executing (FS)**: All the tasks are executed on SCBS
- **Energy and delay greedy (EDS)**: Offloading decisions are made using a greedy algorithm, considering energy and delay factors [58].
- **Energy and delay multi-agent reinforcement (EDM)**: Multi-agent reinforcement learning is employed to determine the offloading strategy, taking into account energy and delay considerations [28].

C. Experiment Results

We evaluated the algorithm's performance by measuring the total cost across diverse simulation configurations. This comprehensive metric incorporates factors such as delay, energy consumption, and success rate.

1) Impact of Learning Rate: Fig. 3 illustrates the growth of average rewards at various learning rates ranging from 0.001 to 0.2. As shown in the figure, higher learning rates result in faster convergence of the average reward. However, after 600 epochs of training, the learning rate of 0.1 reaches the optimal value first. It is also noteworthy that the increasing learning rate can lead to finding a local optimal solution instead of the global optimal.

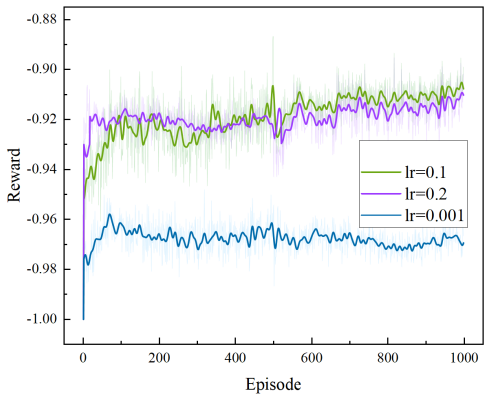

Fig. 3. Average rewards under different learning rates

2) Impact of Transmitting Power: In Fig. 4, the relationship between the total cost and transmitting power is depicted for different task offloading methods. Unlike the FL scheme, the results obtained from the other four methods consistently reveal a reduction in the total cost as transmitting power increases. This is because the increase in transmitting power will reduce delay and improve task completion rate. The FL scheme executes tasks locally, so transmitting power does not affect it. Nevertheless, excessively high transmitting rates can result in elevated energy consumption, subsequently leading to an increase in the total cost. Remarkably, the AVA method consistently maintains the lowest cost across all transmitting power levels employed in the experiment simulation. This means that compared to other offloading schemes that do not include idle vehicle resources, the proposed scheme effectively utilizes idle vehicle resources to alleviate the pressure on the vehicle and edge computing power.

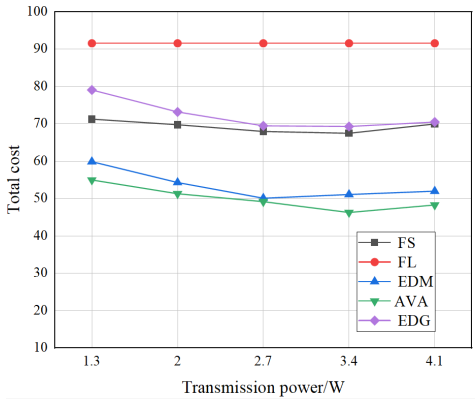

Fig. 4. Comparison of different schemes under different transmitting power scenarios.

3) Impact of Number of Tasks: Fig. 5 demonstrates that the task offloading cost is directly proportional to the number of tasks in all schemes. As the number of tasks increases, the total cost correspondingly rises. This is because as the number of tasks increases, both delay and energy consumption will significantly increase, and the success rate will also decrease. The proposed scheme surpasses the other four schemes due to its generation based on the minimum offloading cost, indicating that the use of idle vehicle resources significantly improves the optimization of task offloading costs. As to FL, due to the local execution of all tasks, with the increase of task quantity and limited local resources, the latency and energy consumption will be much higher than other baseline schemes, and the completion rate will also decrease. Therefore, the overall cost is the highest. This illustrates the importance of task offloading.

4) Impact of Computing Capacity of SCBS: In Fig. 6, it is evident that as the computing capacity of the SCBS increases, the total cost decreases for all baseline schemes except the FL scheme. This is attributed to the fact that full local computation is independent of the SCBS computing capacity and solely relies on the local vehicle's computing capability. Therefore, the trend of FL remains unchanged. Moreover, the trend indicates that as the SCBS computing capacity reaches a high level, the task waiting time becomes negligible, resulting in the lowest total cost for offloading tasks to SCBS. Furthermore,

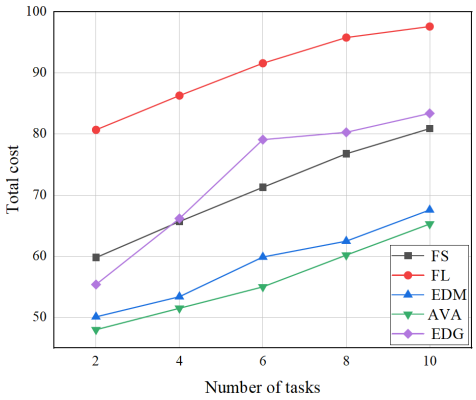

Fig. 5. Comparison of different schemes across different numbers of tasks.

the curves for EDS, EDM, and AVA gradually approach the curve of FS and eventually converge to the FS curve. Notably, the total cost of AVA remains the lowest throughout, demonstrating the superiority of the proposed approach. This is because the proposed solution reduces the waiting time for tasks on the edge server and the time for vehicles to transmit tasks by transferring them to nearby vehicles.

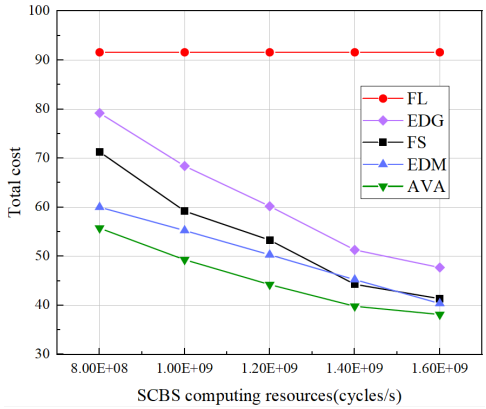

Fig. 6. Comparison of different schemes across different computing capacities of SCBS

5) Impact of Number of Vehicles: In Fig. 7, the comparison results for total cost among the five task offloading schemes are presented across different numbers of vehicles. The total cost consistently rises with the increase in the number of vehicles. This is because an increase in the number of vehicles also leads to a rise in the number of tasks, as mentioned in 3) above. Compared with other schemes, the proposed scheme still maintains the lowest cost, which proves that the proposed scheme has effectively improved overall performance.

VI. Conclusions and Future Work

In this paper, we propose the AVA algorithm to tackle the joint optimization problem in IoV. By leveraging reinforcement learning, agents can dynamically adjust to changes in the real-time environment. The integration of federated learning further enhances and facilitates flexible resource allocation of AV. Simulation experiments conclusively demonstrate that our proposed method effectively allocates unused vehicle resources and reduces overall system costs. Moreover, it adeptly handles complex task offloading problems, even in scenarios involving a large number of states.

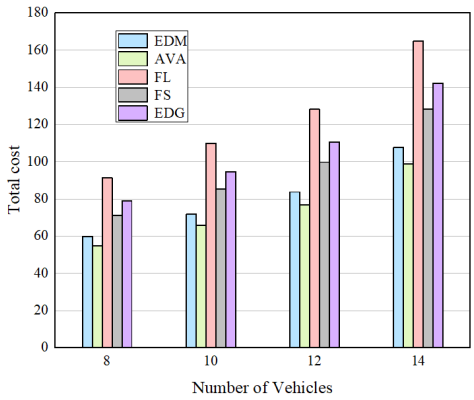

Fig. 7. Comparison of different schemes across different numbers of vehicles

In the future, we aim to enhance the model's performance in complex real-world scenarios. This includes utilizing digital twin simulations to model vehicle trajectories and real-time traffic conditions and incorporating these factors into our model. We also plan to explore increasing the number of auxiliary vehicles to boost available resources and alleviate communication congestion.

ACKNOWLEDGMENT

This work is supported by the Tianjin Science and Technology Planning Project (No. 22ZYYYJC00020) and the National Natural Science Foundation of China (No. 62071327).

REFERENCES

[1] J. Contreras-Castillo, S. Zeadally, and J. A. Guerrero-Ibañez, "Internet of vehicles: Architecture, protocols, and security," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3701–3709, 2018.

[2] A. Kojima and Y. Nose, "Development of an autonomous driving robot car using fpga," in *2018 International Conference on Field-Programmable Technology (FPT)*, 2018, pp. 411–414.

[3] C. Tang, Y. Zhao, and H. Wu, "Lyapunov-guided optimal service placement in vehicular edge computing," *China Communications*, vol. 20, no. 3, pp. 201–217, 2023.

[4] H. Tran-Dang and D.-S. Kim, "Dynamic task offloading approach for task delay reduction in the iot-enabled fog computing systems," in *2022 IEEE 20th International Conference on Industrial Informatics (INDIN)*, 2022, pp. 61–66.

[5] X. Liu and G. Zhang, "Joint optimization offloading and resource allocation in vehicular edge cloud computing networks with delay constraints," in *2020 IEEE International Conference on Progress in Informatics and Computing (PIC)*, 2020, pp. 363–368.

[6] H. Wu, J. Chen, T. N. Nguyen, and H. Tang, "Lyapunov-guided delay-aware energy efficient offloading in iiot-mec systems," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 2, pp. 2117–2128, 2023.

[7] G. Qu, H. Wu, R. Li, and P. Jiao, "Dmro: A deep meta reinforcement learning-based task offloading framework for edge-cloud computing," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3448–3459, 2021.

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

[9] C. Tang, G. Yan, H. Wu, and C. Zhu, "Computation offloading and resource allocation in failure-aware vehicular edge computing," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 1877–1888, 2024.

[10] S. Wang, N. Xin, Z. Luo, and T. Lin, "An efficient computation offloading strategy based on cloud-edge collaboration in vehicular edge computing," in *2022 International Conference on Computing, Communication, Perception and Quantum Technology (CCPQT)*, 2022, pp. 193–197.

[11] X. Lv, H. Du, and Q. Ye, "Tbtoa: A dag-based task offloading scheme for mobile edge computing," in *ICC 2022-IEEE International Conference on Communications*. IEEE, 2022, pp. 4607–4612.

[12] S. Yu, X. Chen, L. Yang, D. Wu, M. Bennis, and J. Zhang, "Intelligent edge: Leveraging deep imitation learning for mobile edge computation offloading," *IEEE Wireless Communications*, vol. 27, no. 1, pp. 92–99, 2020.

[13] M. S. B. F. L. A. Ding, "A cluster-based cooperative computation offloading scheme for c-v2x networks," *Ad Hoc Networks*, vol. 132, p. 102862, 2022.

[14] C. Yang, X. Xu, X. Zhou, and L. Qi, "Deep q network–driven task offloading for efficient multimedia data analysis in edge computing–assisted iov," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 18, no. 2s, pp. 1–24, 2022.

[15] S. Raza, W. Liu, M. Ahmed, M. R. Anwar, M. A. Mirza, Q. Sun, and S. Wang, "An efficient task offloading scheme in vehicular edge computing," *Journal of Cloud Computing*, vol. 9, pp. 1–14, 2020.

[16] R. Zhang, L. Wu, S. Cao, D. Wu, and J. Li, "A vehicular task offloading method with eliminating redundant tasks in 5g hetnets," *IEEE Transactions on Network and Service Management*, vol. 20, no. 1, pp. 456–470, 2022.

[17] W. Feng, S. Yang, Y. Gao, N. Zhang, R. Ning, and S. Lin, "Reverse offloading for latency minimization in vehicular edge computing," in *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[18] Y. Liu, H. Yu, S. Xie, and Y. Zhang, "Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11 158–11 168, 2019.

[19] J. Wang, J. Hu, J. Mills, G. Min, M. Xia, and N. Georgalas, "Federated ensemble model-based reinforcement learning in edge computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 06, pp. 1848–1859, 2023.

[20] C. Nadiger, A. Kumar, and S. Abdelhak, "Federated reinforcement learning for fast personalization," in *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, 2019.

[21] X. He, H. Lu, M. Du, Y. Mao, and K. Wang, "Qoe-based task offloading with deep reinforcement learning in edge-enabled internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 4, pp. 2252–2261, 2021.

[22] X. Wang, C. Wang, X. Li, V. C. Leung, and T. Taleb, "Federated deep reinforcement learning for internet of things with decentralized cooperative edge caching," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9441–9455, 2020.

[23] H. Li and K. Shi, "Incentive mechanism for federated learning participants based on statistical analysis features," in *2023 4th International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE)*, 2023, pp. 158–161.

[24] G. Mani and G. B. Namomsa, "Large language models (llms): Representation matters, low-resource languages and multi-modal architecture," in *2023 IEEE AFRICON*, 2023, pp. 1–6.

[25] K. Chen, Y. Yang, B. Chen, J. A. H. López, G. Mussbacher, and D. Varró, "Automated domain modeling with large language models: A comparative study," in *2023 ACM/IEEE 26th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, 2023, pp. 162–172.

[26] Z. Bimagambetova, D. Rakhymzhanov, A. Jaxylykova, and A. Pak, "Evaluating large language models for sentence augmentation in low-resource languages: A case study on kazakh," in *2023 19th International Asian School-Seminar on Optimization Problems of Complex Systems (OPCS)*, 2023, pp. 14–18.

[27] P. Ulhe and S. Asole, "Empirical analysis of qos and security aware iov routing models from a statistical perspective," in *2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE)*, 2023, pp. 496–501.

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

[29] L. Huang, X. Feng, L. Qian, and Y. Wu, "Deep reinforcement learning-based task offloading and resource allocation for mobile edge computing," in *Machine Learning and Intelligent Communications: Third International Conference, MLICOM 2018, Hangzhou, China, July 6-8, 2018, Proceedings 3*. Springer, 2018, pp. 33–42.

[30] C. Pan, Z. Wang, H. Liao, Z. Zhou, X. Wang, M. Tariq, and S. Al-Otaibi, "Asynchronous federated deep reinforcement learning-based urllc-aware computation offloading in space-assisted vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 7, pp. 7377–7389, 2023.

[31] Y. Bian, Y. Sun, M. Zhai, W. Wu, Z. Wang, and J. Zeng, "Dependency-aware task scheduling and offloading scheme based on graph neural network for mec-assisted network," in *2023 IEEE/CIC International Conference on Communications in China (ICCC Workshops)*, 2023, pp. 1–6.

[32] "Ieee guide for architectural framework and application of federated machine learning," *IEEE Std 3652.1-2020*, pp. 1–69, 2021.

[33] J. Kang, R. Yu, X. Huang, M. Wu, S. Maharjan, S. Xie, and Y. Zhang, "Blockchain for secure and efficient data sharing in vehicular edge computing and networks," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4660–4670, 2019.

[34] X. Wang, Y. Zhao, C. Qiu, Z. Liu, J. Nie, and V. C. M. Leung, "Infedge: A blockchain-based incentive mechanism in hierarchical federated learning for end-edge-cloud communications," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 12, pp. 3325–3342, 2022.

[35] D. C. Nguyen, M. Ding, Q.-V. Pham, P. N. Pathirana, L. B. Le, A. Seneviratne, J. Li, D. Niyato, and H. V. Poor, "Federated learning meets blockchain in edge computing: Opportunities and challenges," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12 806–12 825, 2021.

[36] L. Lyu, Y. Shen, and S. Zhang, "The advance of reinforcement learning and deep reinforcement learning," in *2022 IEEE International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA)*, 2022, pp. 644–648.

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

[38] B. Hazarika, K. Singh, S. Biswas, S. Mumtaz, and C.-P. Li, "Multi-agent drl-based task offloading in multiple ris-aided iov networks," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 1, pp. 1175–1190, 2024.

[39] M. Xu, J. Peng, B. B. Gupta, J. Kang, Z. Xiong, Z. Li, and A. A. A. El-Latif, "Multiagent federated reinforcement learning for secure incentive mechanism in intelligent cyber–physical systems," *IEEE Internet of Things Journal*, vol. 9, no. 22, pp. 22 095–22 108, 2022.

[40] S. K. Singh, S. Kumar, and P. S. Mehra, "Chat gpt & google bard ai: A review," in *2023 International Conference on IoT, Communication and Automation Technology (ICICAT)*, 2023, pp. 1–6.

[41] Y. Otoum, Y. Wan, and A. Nayak, "Transfer learning-driven intrusion detection for internet of vehicles (iov)," in *2022 International Wireless Communications and Mobile Computing (IWCMC)*, 2022, pp. 342–347.

[42] E. Li, L. Zeng, Z. Zhou, and X. Chen, "Edge ai: On-demand accelerating deep neural network inference via edge computing," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 447–457, 2020.

[43] J. Fang, Y. He, F. R. Yu, J. Li, and V. C. Leung, "Large language models (llms) inference offloading and resource allocation in cloud-edge networks: An active inference approach," in *2023 IEEE 98th Vehicular Technology Conference (VTC2023-Fall)*, 2023, pp. 1–5.

[44] L. Lin, X. Liao, H. Jin, and P. Li, "Computation offloading toward edge computing," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1584–1607, 2019.

[45] J. Lee and W. Na, "A survey on vehicular edge computing architectures," in *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*, 2022, pp. 2198–2200.

[46] S. Yu, X. Chen, Z. Zhou, X. Gong, and D. Wu, "When deep reinforcement learning meets federated learning: Intelligent multitimescale resource management for multiaccess edge computing in 5g ultradense network," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2238–2251, 2021.

[47] Q. Wu, Y. Zhao, Q. Fan, P. Fan, J. Wang, and C. Zhang, "Mobility-aware cooperative caching in vehicular edge computing based on asynchronous federated and deep reinforcement learning," *IEEE Journal of Selected Topics in Signal Processing*, vol. 17, no. 1, pp. 66–81, 2023.

[48] T. Liu, T. Zhang, J. Loo, and Y. Wang, "Deep reinforcement learning-based resource allocation for uav-enabled federated edge learning," *Journal of Communications and Information Networks*, vol. 8, no. 1, pp. 1–12, 2023.

[49] G. Wang, C. Wu, Z. Du, T. Yoshinaga, R. Yin, and L. Zhong, "Drl-assisted network selection for federated iov," *IEEE Internet of Things Magazine*, vol. 6, no. 3, pp. 86–90, 2023.

[50] N. M. Al-Maslamani, M. Abdallah, and B. S. Ciftler, "Reputation-aware multi-agent drl for secure hierarchical federated learning in iot," *IEEE Open Journal of the Communications Society*, vol. 4, pp. 1274–1284, 2023.

[51] K. Zheng, F. Liu, Q. Zheng, W. Xiang, and W. Wang, "A graph-based cooperative scheduling scheme for vehicular networks," *IEEE transactions on vehicular technology*, vol. 62, no. 4, pp. 1450–1458, 2013.

[52] P. Luoto, M. Bennis, P. Pirinen, S. Samarakoon, K. Horneman, and M. Latva-aho, "Vehicle clustering for improving enhanced lte-v2x network performance," in *2017 European Conference on Networks and Communications (EuCNC)*, 2017, pp. 1–5.

[53] S. Nikou, J. Mezei, and H. Bouwman, "Analytic hierarchy process (ahp) approach for selecting mobile service category (consumers' preferences)," in *2011 10th International Conference on Mobile Business*, 2011, pp. 119–128.

[54] H. Wang, X. Li, H. Ji, and H. Zhang, "Federated offloading scheme to minimize latency in mec-enabled vehicular networks," in *2018 IEEE Globecom Workshops (GC Wkshps)*, 2018, pp. 1–6.

[55] D. Mazza, D. Tarchi, and G. E. Corazza, "A partial offloading technique for wireless mobile cloud computing in smart cities," in *2014 European Conference on Networks and Communications (EuCNC)*, 2014, pp. 1–5.

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

[57] O. Muñoz, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 10, pp. 4738–4755, 2015.

[58] J. Wang, J. Hu, G. Min, A. Y. Zomaya, and N. Georgalas, "Fast adaptive task offloading in edge computing based on meta reinforcement learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 1, pp. 242–253, 2020.

Huaming Wu (Senior Member, IEEE) received the B.E. and M.S. degrees from Harbin Institute of Technology, China in 2009 and 2011, respectively, both in electrical engineering. He received the Ph.D. degree with the highest honor in computer science at Freie Universität Berlin, Germany in 2015. He is currently a professor at the Center for Applied Mathematics, Tianjin University, China. His research interests include mobile cloud computing, edge computing, internet of things, deep learning, complex networks, and DNA storage.

Anqi Gu received the Bachelor's degree in Mathematics and Applied Mathematics from Hainan University, China, in 2022. She is currently pursuing a master's degree at the School of Mathematics, Tianjin University, China. Her research interests include edge computing, Internet of Things, federated learning, and graph neural networks.

Yonghui Liang received the Bachelor's degree in Information and Computational Science from Jiangxi University of Finance and Economics, China, in 2021. He is currently pursuing a master's degree at the School of Mathematics, Tianjin University, China. His research interests include the Internet of Things, mobile edge computing, vehicular networks and deep learning.