IoV-Oriented Integrated Sensing, Computation, and Communication: System Design and Resource Allocation

Junhui Zhao, *Senior Member, IEEE*, Ruixing Ren, Dan Zou, Qingmiao Zhang,
and Wei Xu, *Senior Member, IEEE*

***Abstract*—In future mobile communication systems, the deep integration of communication, sensing, and computation has become a new trend. This article paper designs an integrated sensing, computation, and communication (ISCC) system for internet of vehicles (IoV) by leveraging mobile edge computing (MEC) and integrated sensing and communication (ISAC). Specifically, a collaborative sensing data fusion architecture is proposed, where vehicles collaborate with road side unit (RSU) to address the issue of limited sensing range for a single vehicle. Furthermore, the randomness of collaborative sensing tasks arrival in real-world traffic scenarios is simulated, with communication, sensing, and computation being modeled respectively. The joint optimization of wireless and edge computing resources in the ISCC system aims at maximizing the completion rate of collaborative sensing tasks while ensuring system service delay. To address the mixed-integer nonlinear optimization problem, a resource allocation scheme based on deep reinforcement learning (DRL) is developed, which enables adaptive allocation of wireless and edge computing resources through online learning. Simulation results show that agent of DRL can learn a smart resource allocation strategy from the interactive environment, and achieve better performance than conventional resource allocation schemes.**

***Index Terms*—Deep reinforcement learning (DRL), Internet of Vehicles (IoV), integrated sensing and communication (ISAC), mobile edge computing (MEC), resource allocation.**

I. Introduction

THE advent of fifth-generation (5G) has not only accelerated data transmission speeds but has also propelled the development of various novel applications. Its characteristics, including ultra-high data rates, ultra-low latency, and ultra-large capacity, have basically met the requirements for commercial viability. At the same time, artificial intelligence (AI) has achieved great success in various applications and continues to penetrate various industries with an explosive growth trend. Especially in the current 5G network, the introduction of AI can effectively solve some complex problems in the network. In the sixth-generation (6G) era, AI will be deeply integrated into network design to achieve the so-called AI native network. Therefore, the future 6G wireless communication network will surpass pure fragmented data transmission pipelines and become an intelligent platform that integrates sensing, communication, and computation to provide universal AI services and ubiquitous sensing services [1], [2], [3], [4].

In traditional schemes, sensing and communication processes are often designed separately, which not only leads to high costs but also low efficiency. Fortunately, in the upcoming beyond 5G and 6G eras, wireless sensing and communication systems are moving towards higher frequency bands, larger antenna arrays, and miniaturization, becoming increasingly similar in hardware architecture, channel characteristics, and signal processing, leading to the emergence of a popular research field known as integrated sensing and communication (ISAC) [5]. However, despite the existing achievements of ISAC research, which have already met the service requirements of some application domains, pervasive sensing services will encounter a demand for processing massive amounts of sensing data in future applications such as autonomous driving, internet of vehicles (IoV), and smart homes [6]. At this point, it is crucial to consider integrated sensing, computation, and communication (ISCC) research to accelerate data processing and provide users with a satisfying experience.

Recently, many computing-enabled technologies have been combined with ISAC technology to construct ISCC systems to improve the utilization of limited resources in wireless networks. For example, inspired by mobile edge computing (MEC) technology, an ISCC system was proposed in [7] by combining ISAC and MEC. A new wireless scheduling architecture was proposed in [8], which combined MEC and ISAC technology to construct an ISCC framework, exploring the inherent trade-offs between sensing, computation, and communication performance

Manuscript received 9 January 2024; revised 22 March 2024; accepted 24 June 2024. Date of publication 2 July 2024; date of current version 7 November 2024. This work was supported in part by the National Natural Science Foundation of China under Grant U2001213 and Grant 62261024, in part by the National Key Research and Development Program of China under Grant 2020YFB1807204, in part by the Key Laboratory of Universal Wireless Communications of Ministry of Education under Grant KFKT-2022101, and in part by the Jiangxi Provincial Department of Education Scientific Research Project under Grant GJJ2200655. The review of this article was coordinated by Dr. Ying He. *(Corresponding author: Junhui Zhao.)*

Junhui Zhao is with the School of Information Engineering, East China Jiaotong University, Nanchang 330013, China, and also with the School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China (e-mail: junhuizhao@hotmail.com).

Ruixing Ren, Dan Zou, and Qingmiao Zhang are with the School of Information Engineering, East China Jiaotong University, Nanchang 330013, China (e-mail: renruixing0604@163.com; 35230900@qq.com; zqm@ecjtu.edu.cn).

Wei Xu is with the National Mobile Communications Research Lab, Southeast University, Nanjing 210096, China, and also with Purple Mountain Laboratories, Nanjing 211111, China (e-mail: wxu@seu.edu.cn).

Digital Object Identifier 10.1109/TVT.2024.3422270

from the perspective of joint optimization. In addition to MEC enabling technology, over-the-air computing (AirComp) and ISAC were integrated in the design [9]. The Air-ISCC system was achieved through appropriate radar signal design, which was also the first opening work related to Air-ISCC. Subsequently, the combination of ISAC and AirComp was also studied in [10]. In [10], the authors combined ISAC with AirComp to develop a framework of integrated sensing, computation, and communication over-the-air (ISCCO) to improve spectral efficiency and sensing performance, and jointly optimized beamforming for sensing, communication, and computation.

In addition, driven by the development of AI, there are also some studies on ISCC related to edge intelligence. A new multi-device edge AI system was studied in [11], which combined AI model split inference technology and ISAC technology to achieve low latency intelligent services at the network edge. This is also the first attempt to design a task-oriented ISCC scheme for edge AI inference systems. In [12], federated edge learning (FEEL) was used to handle complex interwoven relationships between sensing, computation, and communication. In order to process computing signals, [13] utilized the shared learning process of aerial federated learning to train a global model through the collaboration of multiple intelligent sensors, and designed an ISCC framework to optimize the limited system resources of 6G wireless networks. In addition to the relevant research mentioned above, there have also been studies combining the ISCC system with other enabling technologies. In [14], an unmanned aerial vehicle (UAV) enabled ISCC (UISCC) framework was designed and demonstrated that UAV can fully explore their functions to improve sensing, computation, and communication capabilities. This is also a preliminary attempt to explore ISCC system that supports UAV. In [15], an ISCC system has been constructed to demonstrate the proposed use of intelligent reflective surface (IRS) reflection and scattering to achieve wireless radio frequency uplink transmission.

While ongoing research on ISCC is abundant, the majority of studies primarily focus on system modeling and performance optimization, with limited application to specific domains. Coincidentally, autonomous driving and IoV have become research hotspots recently, while traditional communication, sensing, computation, and other multiple systems using cut and conquer design and performance optimization methods can no longer meet the extremely low latency and highly reliable transmission requirements of broadband sensing information in the context of autonomous driving in the IoV.

In addition, existing research on ISCC performance optimization generally adopts traditional optimization methods [7], [8], [9], [10], [11], [12], [13], [14], [15], such as Lyapunov optimization or linear programming relaxation method. On the one hand, although traditional optimization methods have a wide range of applications and are still very classic and practical, there is no doubt that when faced with some complex optimization problems, they will have problems such as high time complexity and low efficiency, especially in complex IoV scenarios where resource usage is clearly dynamic, random, and irregular. On the other hand, the introduction of AI technology in communication systems can effectively solve some complex problems and provide an alternative solution for system optimization. Therefore, more and more researchers are constantly exploring the use of deep reinforcement learning (DRL) algorithms to solve resource allocation control and decision-making problems. Luong et al. [16] provided a comprehensive overview of the application of DRL in communication, and DRL resource allocation schemes have been widely used in the entire scientific and engineering fields. Therefore, the DRL-based scheme is regarded as a promising technology.

According to the specific research content, the integration of sensing, computation and communication for IoV can be divided into two different models. One is the functional fusion model, which specifically includes channel modeling, channel estimation, V2I beam alignment, and other sensing functions assisting communication and computation, as well as communication functions supporting beyond-line-of-sight sensing and computation. The other is the signal fusion model, which focuses on physical layer optimization and cross-layer joint optimization design. Based on the above considerations, this work mainly focuses on beyond-line-of-sight sensing, studying the fusion of the functional level in IoV ISCC system, as well as the performance optimization of the fusion system. Specifically, the communication function can effectively aggregate and transmit sensing information from different nodes on the traffic road to support multi-node cooperative sensing, thereby expanding the dimensions and depths of sensing for each vehicle. Edge collaborative nodes, such as road side unit (RSU), leverage their powerful computational capabilities to process sensing data in real-time. Considering the limited resources of the system, we model communication, sensing, and computation, and use DRL to optimize system resources to achieve optimal system performance. The main contributions of this paper are as follows.

- To address the limitation of the sensing range in a single vehicle, a vehicle-road collaboration-centered IoV-oriented ISCC system is constructed. Moreover, a collaborative sensing data fusion framework is proposed, facilitating the collaboration between vehicles and RSU in sensing tasks.
- The high randomness of vehicle collaborative sensing task arrivals in real-world traffic scenarios is simulated, with communication, sensing, and computation being modeled respectively. By jointly optimizing wireless and edge computing resources, the objective is to maximize the task completion rate while ensuring system service delay.
- In order to tackle the mixed-integer nonlinear optimization problem, a resource allocation scheme based on DRL is developed to dynamically allocate system resources in an adaptive manner. Simulation results show that the proposed scheme achieve better performance than other resource allocation schemes.

The rest of this paper is organized as follows. The system model and problem formulation are introduced in Section II. The developed DRL-based scheme and its details are introduced in Section III. The simulation results are presented in Section IV. Section V is the summary of this paper.

II. System Model and Problem Formulation

In future mobile communication systems, using terahertz technology and combining millimeter and submillimeter waves,

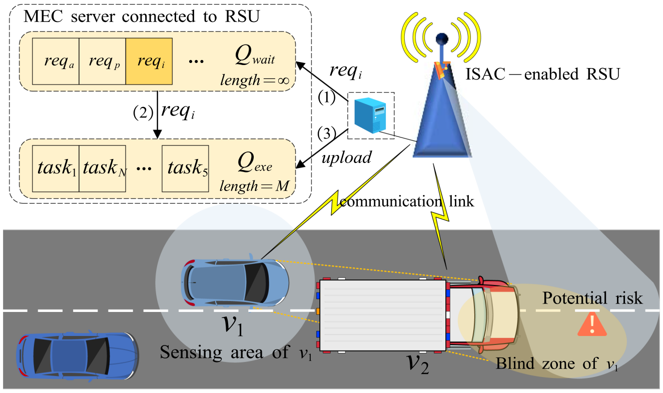

Fig. 1. IoV-oriented ISCC system model.

it is expected that a base station (BS) will have a coverage range of less than 200 m [17]. This means that the network will be densified, and dense small BSs will be ubiquitous. The ISAC device integrates sensing and communication functions and can be deployed on RSU and BS [18] to perform ubiquitous environmental sensing services, such as video surveillance, radar detection, and meteorological detection. As shown in Fig. 1, an ISCC system is considered for a traffic scenario, comprising of a set of N vehicles and an enhanced RSU that facilitates ISAC devices. Therefore, in addition to the traditional communication transmission function, the enhanced RSU has environmental sensing function and can handle various computational tasks for vehicle offloading. The collaborative sensing between enhanced RSU and vehicles can effectively eliminate blind spots in vehicle sensors. In Fig. 1, the right front view of the vehicle v_1 is obstructed by the vehicle v_2, which may prevent potential risks from being detected. In this case, collaboration with RSU can effectively detect potential risks and even obtain a wider sensing area.

A. Collaborative Sensing Data Fusion Framework

Although the collaborative can solve the problem of limited sensing range. However, the vast and multi-source sensing data from RSU and vehicles cannot be directly stacked together. They must undergo a data fusion process for integration. In general, there are two options for fusion. One is to transmit the sensing data of the RSU to the vehicles, perform data fusion on the vehicle side, and then consider offloading the fusion data to the RSU connected to the MEC server for processing, and finally return the results to the vehicles [18], [19]. However, on the one hand, this download-fusion-offload-again download method is cumbersome and can lead to significant latency. On the other hand, the amount of sensing data generated by RSU is often large, and directly transmitting the data to vehicles requires significant spectral resources. Therefore, the adoption of an alternative fusion method is carried out on the side of the RSU. The advantage of this is that the MEC server equipped at the RSU has sufficient computing power to support the computing power requirements of sensing data fusion and processing.

The proposed collaborative sensing data fusion framework is shown in Fig. 2. Specifically, the vehicle transmits locally sensed data via the uplink to the RSU, and utilizes the powerful computing power of the MEC server to the fuse with the RSU sensed data to generate fusion tasks and process them. The obtained computed results are returned to the vehicle via the downlink. A triplet set $\mathcal{R} = \{R^u, R^c, R^d\}$ is defined to represent wireless communication and edge computing resources in the system, where R^u and R^d represent the uplink and downlink bandwidth owned by the RSU, respectively, and R^c represents the edge computing resources owned by the MEC server at the RSU.

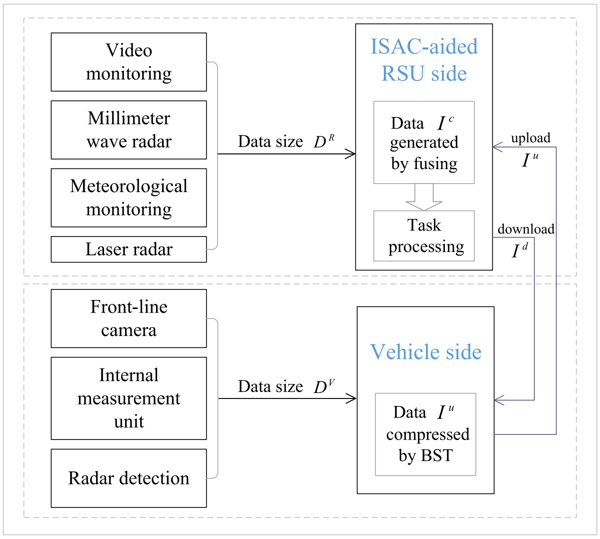

Fig. 2. Collaborative sensing data fusion framework.

B. Sensing Model

In the future, RSUs can deploy a combination of multiple types of sensors to sense the surrounding environment [20]. In this work, time division multiple access (TDMA) method is considered to perform radar sensing and communication with vehicles in RSU. Specifically, there are two types of time slots: sensing and communication, where the length of each time slot is τ and the time slot index is t. The delay in collecting sensing data is always fixed [7], so this work does not consider the process of collecting sensing data, i.e. assuming that the vehicle and RSU can obtain high-quality sensing data in advance. Vehicles can request RSU for collaborative sensing at any time, so the arrival of collaborative sensing tasks also has the high degree of randomness, including the randomness of task arrival time, task size, and delay tolerance. In addition, Only when the vehicle uploads all its sensing data to the RSU can it fuse with the sensing data at the RSU to form the final fusion sensing task.

It is worth noting that the sensing data generated by vehicles are multimodal, with a data volume of up to 2 GB/s, which leads to unaffordable upload delays. In order to be closer to the real-world scene, the background subtraction technology (BST) [21] is used to compress and preprocess the generated sensing data on the vehicle side, and then upload them to save time. The ratio of compressed sensing data to the original generated sensing data is expressed as r^{sep}, and the CPU cycle requirement for the separation technique is defined M^{sep}. Therefore, the computational delay for compressing the original sensing data of vehicle i is

$$d_i^{sep} = \frac{D_i^V M^{sep}}{f_i} \tag{1}$$

where D_i^V and f_i respectively represent the size of sensing data generated by vehicle i and the local computing power of vehicle i. This work does not consider compression processing for the computed results of collaborative sensing tasks at RSU, as the size of computed results downloaded are much smaller than the uploaded data [22]. The size of the data that needs to be uploaded after compression can be expressed as

$$I_i^u = r^{sep} D_i^V \tag{2}$$

For the collaborative sensing task of vehicle i, a 6-tuple set is defined, namely $task_i = \{t_i, T_i^{\max}, I_i^u, I_i^c, I_i^d, \lambda_i\}$, where t_i represents the arrival time of RSU collaboration sensing requested by the vehicle i, $T_i^{\max}$ represents the delay threshold, I_i^u represents the size of sensing data to be uploaded on the vehicle side, $I_i^c = Fusion(I_i^u, D_i^R)$ represents the data size of the fusion sensing task to be processed, I_i^d represents the computed result data size, λ_i represents the priority weight.

C. Communication Model

Although the MEC server at the RSU has significant computing power, it is not infinite and cannot simultaneously handle an infinite number of computational tasks [23]. Therefore, as shown in Fig. 1, when the vehicle requests RSU for collaborative sensing, it will first send a request instruction, which will first enter the waiting queue Q_{wait} in the RSU. After there is an empty space in the execution queue Q_{exe}, Q_{wait} will sequentially instruct the corresponding vehicles to upload local sensing data to the RSU based on the priority of all requests. Furthermore, the possibility of early completion is taken into consideration, which allows the tasks arriving in the time domain to be divided into the following stages: waiting stage, preprocessing stage, upload stage, computing stage, download stage, and remaining stage.

The communication delay between the vehicle and RSU is influenced by the communication spectrum allocated to the vehicle and the data size being transferred. $R^u(t) \triangleq \{R_i^u\}_N$ and $R^d(t) \triangleq \{R_i^d\}_N$ are defined to respectively represent the uplink and downlink bandwidth allocation strategys at time slot t, and the allocated bandwidth should meet $\sum_{i \in N} R_i^u(t) \leq R^u$, $\sum_{i \in N} R_i^d(t) \leq R^d$, $\forall t$. There is no interference between all communication and sensing waves due to TDMA. Based on the preceding discussion, the uplink and downlink communication rates of vehicle i in time slot t can be derived, which are

$$v_i^u(t) = R_i^u(t) log_2 \left(1 + \frac{P_{i,RSU}(t) h_i(t)}{\sigma^2}\right) \tag{3}$$

$$v_i^d(t) = R_i^d(t) log_2 \left(1 + \frac{P_{RSU,i}(t) h_i(t)}{\sigma^2}\right) \tag{4}$$

where $P_{i,RSU}(t)$ and $P_{RSU,i}(t)$ respectively denote the transmission power of the vehicle i and RSU during data upload and download at time slot t. $h_i(t)$ represents the channel gain between the vehicle i and RSU at time slot t. σ^2 is Gaussian white noise.

D. Computation Model

The computational delay at the MEC server depends on the edge computing resources allocated to each collaborative sensing task and the inherent computing power of the server. $R^c(t) \triangleq \{R_i^c\}_N$ is defined as the edge computing resources allocation strategy at time slot t, and satisfy $\sum_{i \in N} R_i^c(t) \leq R^c$. Here three binary control variables (x_i^u, x_i^c, x_i^d) are introduced to represent the current stage of collaborative sensing task i, where $x_i^u, x_i^c, x_i^d = 1$ respectively represent the upload, computation, and download stages of the task, otherwise $x_i^u, x_i^c, x_i^d = 0$.

Given a edge computing resource allocation strategy, the size of the remaining computation data for the next time slot can be obtained, which can be calculated as

$$D_i^c(t+1) = D_i^c(t) - x_i^c \tau v_i^c(t) = I_i^c - \sum_{j=t_i}^{t} x_i^c \tau v_i^c(j) \tag{5}$$

where $D_i^c(t)$ and $v_i^c(t)$ represent the remaining computation data size and the computational rate of fusion sensing task i in time slot t. It is worth noting that the range of t satisfies $[t_i, t_i + T_i^{\max}]$, and $D_i^c(t_i) = I_i^c$. The total computational delay d_i^{comp} of the fusion sensing task is defined as the solution of $D_i^c(t) = 0$, which can be expressed as

$$d_i^{comp} = D_i^{c(-1)}(0) \tag{6}$$

Similarly, based on the given communication resource allocation strategy, the size of the remaining communication data in the next time slot can be calculated as

$$D_i^u(t+1) = D_i^u(t) - x_i^u \tau v_i^u(t) = I_i^u - \sum_{j=t_i}^{t} x_i^u \tau v_i^u(j) \tag{7}$$

$$D_i^d(t+1) = D_i^d(t) - x_i^d \tau v_i^d(t) = I_i^d - \sum_{j=t_i}^{t} x_i^d \tau v_i^d(j) \tag{8}$$

where $D_i^u(t)$ and $D_i^d(t)$ represent the remaining upload and download data sizes of sensing task i in time slot t. Then, the communication delay d_i^{comm} can be obtained as the solution of $D_i^u(t) = 0$ plus $D_i^d(t) = 0$. The range of t in both $D_i^u(t)$ and $D_i^d(t)$ satisfies $[t_i, t_i + T_i^{\max}]$, and $D_i^u(t_i) = I_i^u$, $D_i^d(t_i) = I_i^d$. The transmission delay of communication can be obtained as

$$d_i^{comm} = D_i^{u(-1)}(0) + D_i^{d(-1)}(0) \tag{9}$$

The delay in the vehicle sending request instruction can be omitted, as instruction size is usually several bits. After the instruction enters the waiting queue Q_{wait}, the waiting delay will be recorded. In summary, we can obtain the total system service delay of vehicle i's collaborative sensing task, which can be expressed as

$$d_i = d_i^{sep} + d_i^{wait} + d_i^{comm} + d_i^{comp} \tag{10}$$

E. Problem Formulation

Our goal is to improve the completion rate of collaborative sensing tasks by reasonably allocating limited wireless communication and edge computing resources in each time slot. It is worth mentioning that we do not consider optimizing system delay and energy consumption. Because our data and delay threshold settings are close to the real-world scenarios, optimizing delay while ensuring task completion rate is meaningless. In addition, a penalty term for delay in the setting of the reward function is added in the following text. The more timely the task is completed, the greater the reward obtained, which can indirectly save energy consumption. The variable y_i is defined to indicate whether the collaborative sensing task of vehicle i is completed within the specified delay threshold. If it is completed, then $y_i = 1$, otherwise it is 0. So our optimization problem can be expressed as

$$
\begin{aligned}
\mathcal{P}1: &\max_{R^u(t),R^c(t),R^d(t)} \frac{\sum_{i=1}^{N} y_i}{N} \\
\text{s.t. } & \text{C}1: x_i^k = \{0,1\}, \forall k \in \{u,c,d\}, \\
& \text{C}2: x_i^u + x_i^c + x_I^d \le 1, \\
& \text{C}3: 0 \le D_i^u(t) \le r^{sep} I_i^u, \forall t \in [t_i, t_i + T_i^{\max}], \\
& \text{C}4: 0 \le D_i^c(t) \le I_i^c, \forall t \in [t_i, t_i + T_i^{\max}], \\
& \text{C}5: 0 \le D_i^d(t) \le I_i^d, \forall t \in [t_i, t_i + T_i^{\max}], \\
& \text{C}6: \sum_{i=1}^{N} R_i^j(t) \le R^j, R_i^j(t) \ge 0, \forall j \in \{u,c,d\}, \\
& \text{C}7: d_i \le T_i^{\max}, \\
& \text{C}8: 0 < r^{sep} \le 1.
\end{aligned}
$$

where $\text{C}1 - \text{C}2$ represent the system stage in which task i is in. $\text{C}3 - \text{C}5$ represent the range of remaining data for task i in each state. C6 represents that the resources occupied by each time slot cannot exceed the upper and lower limits of the system. C7 represents the total latency of each task execution, which cannot exceed its maximum latency threshold. C8 represents the limit on the compression ratio before uploading.

III. Resource Allocation Scheme Based on DRL

In order to solve the mixed-integer nonlinear programming problem $\mathcal{P}1$, this section uses a resource allocation algorithm based on DRL to jointly optimize ISCC system resources to improve the completion rate of collaborative sensing tasks. The following contents in this section will sequentially present the design of the agent, the formulation of the DRL algorithm, its execution details and complexity analysis.

A. Agent Design

1) Agent Environment Observation: The observed environmental states of the agent include the priority state $s_i^{\lambda}(t)$ of waiting tasks in the waiting queue Q_{exe}, the upload state $s_i^u(t)$, computing state $s_i^c(t)$, and download state $s_i^d(t)$ of tasks in the execution queue Q_{exe}, and the global time state $s_i^T(t)$ of the collaborative sensing tasks. The time state $s_i^T(t)$ is defined as the normalized remaining executable time of task i, which is

$$s_i^T(t) = \frac{t_i + T_i^{\max} - t}{T_{norm}}, t_i \ge t \tag{11}$$

where T_{norm} is the maximum value of the initial delay threshold for all collaborative sensing tasks. Similarly, the states of the three stages are defined as the normalized remaining data size of each stage, which are

$$s_i^j(t) = \frac{D_i^j(t)}{D_{norm}^j}, j \in \{u,c,d\} \tag{12}$$

where D_{norm}^j is the maximum value of the initial task data for stage j. The task priority state $s_i^{\lambda}(t)$ in the waiting queue Q_{wait} is defined as

$$s_i^{\lambda}(t) = \frac{s_i^D(t)}{s_i^T(t) + \varepsilon} \tag{13}$$

where $s_i^D(t)$ is the average remaining data size for the three states. If the average remaining data size is larger and the remaining executable time is less, the priority of the collaborative sensing task for vehicle i in waiting queue Q_{wait} will be higher. Considering that when $t = t_i + T_i^{\max}$, a small value ε is added.

2) Agent Action Space: The action space of the agent includes uplink bandwidth $a_i^u(t)$, MEC server computing resource $a_i^c(t)$, and downlink bandwidth $a_i^d(t)$. Here we can provide a detailed explanation of the reasons for setting up waiting queues and execution queues. On the one hand, it is a recognized fact that the computing power of the MEC server at the RSU is not infinite, so there is a certain threshold for the number of tasks that the system can handle at the same time. On the other hand, the dimensions of the state space and action space observed by the agent in DRL are fixed, which is set to M in this paper. The size of M depends on the size of resources available to the system.

3) Agent Reward Function: To meet the optimization problem $\mathbb{P}_1$ while avoiding reward sparsity [24], it is necessary to define the reward function for each time slot. The reward function is defined as the weighted sum of rewards for collaborative sensing tasks in the execution queue Q_{exe} for each time slot t, which is

$$r(t) = \sum_{i \in M} \left[\gamma_1 r_i^{ok}(t) - \gamma_2 r_i^T(t) r_i^D(t)\right] \tag{14}$$

where γ_1, γ_2 is the weight factor. $r_i^{ok}(t) = 1$ denotes that the collaborative sensing task i is completed, otherwise is 0. To maintain numerical stability, we adopt $r_i^T(t) = 2 - s_i^T(t)$ to represent the time consumed by executing task i. $r_i^D(t) = s_i^D(t)$ represents the average remaining data of the task of the three stages. By adding this penalty term, $\gamma_2 r_i^T(t) r_i^D(t)$, the agent can be stimulated to quickly complete the task.

B. DRL Algorithm

The DRL algorithm used by the agent in this paper is shown in Fig. 3. Because the environment is more suitable for using

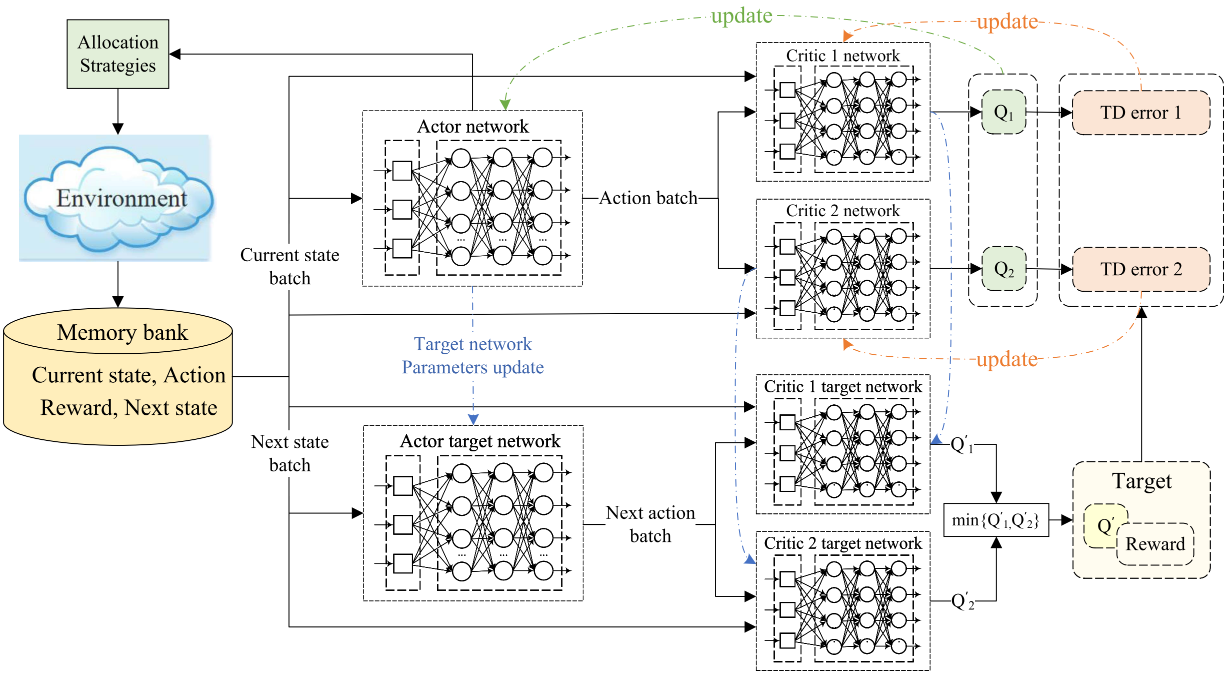

Fig. 3. Design of DRL algorithm for agent.

continuous action control strategies, the actor-critic architecture [25] is used. As is well known, using temporal-difference (TD) learning updates can lead to overestimation problem, and overestimation is non-uniform, which can have a significant impact on the final decision-making of agents.

1) Target Network: Originally, critic network itself is used to calculate the TD target, and then update itself with the TD target, which is bootstrapping. While [26] uses a target network to calculate TD target, which can alleviate the overestimation problem caused by bootstrapping. The parameters of the target critic network are recorded as w'. The TD target can be represented as

$$y(t) = r(t) + \gamma \cdot Q'(s(t+1), a(t+1); w') \tag{15}$$

Then calculate the TD error and update the parameter w of the original critic network.

2) Double Critic Network: Although small modifications to the Double DQN are largely beneficial in alleviating overestimation problem. However, it has also been proven in [27] that the Double DQN solution is ineffective in actor-critic architecture due to slow policy changes in actor-critic architectures, and the current and target value estimates are still too similar to avoid maximizing bias. Therefore, a pair of independently trained critic networks are proposed, as shown in Fig. 3.

In the target network, the estimated values Q_1' and Q_2' are calculated as the minimum, i.e., $Q' = \min(Q_1', Q_2')$, and this minimum value Q' is utilized as the target for updating both Critic 1 and Critic 2 networks. Therefore, the TD target at this point is represented as follow

$$y(t) = r(t) + \gamma \min_{i=1,2} Q'(s(t+1), a(t+1)'; w_i') \tag{16}$$

The loss function using mean squared error is

$$\begin{aligned} loss_i &= B^{-1} \sum \frac{1}{2} \left[Q_i(s(t), a(t); w_i) - y(t) \right]^2 \\ &= B^{-1} \sum \frac{1}{2} \delta_i^2, \quad i = 1,2 \end{aligned} \tag{17}$$

where δ_i is TD error and B represents the size of the mini-batch. Then update the gradient expression of the critic networks parameters as

$$\frac{\partial loss_i}{\partial w_i} = B^{-1} \sum \delta_i \cdot \nabla_{w_i} Q_i(s(t), a(t); w_i), i = 1,2 \tag{18}$$

Then update the network parameters for Critic 1 and Critic 2 as follows

$$w_i \leftarrow w_i - \alpha \cdot \frac{\partial loss_i}{\partial w_i}, \quad i = 1,2 \tag{19}$$

where α is the learning rate of the critic networks.

3) Delayed Update Policy for Actor Network: The delayed update policy of the actor network is equivalent to updating the Actor network after multiple updates of Critic 1 and Critic 2 network. Because the Q value of critic networks is constantly changing during the learning process. If the critic networks are unstable, the unstable change in Q value will lead to the Actor network constantly seeking various optimal strategies, resulting in oscillation phenomenon. Therefore, it is possible to set a lower update frequency for the Actor network compared to the Critic 1 and 2 networks. Specifically, the Actor network can wait for the critic networks to stabilize before initiating its own update. The expression for updating actor network parameters using deterministic gradients is

$$\nabla_\theta J(\theta) = B^{-1} \sum \nabla_a Q_i(s(t), a(t); w_i)$$

$$\cdot \nabla_{\theta}\pi(s(t);\theta), \quad i=1 \ \text{ or} \tag{20}$$

where $\pi(s(t);\theta)$ represents the policy function, θ represents the parameters of the Actor network. The Actor network parameters can be updated using the Q value of Critic 1 or Critic 2 network. Because as the update progresses, Critic 1 and Critic 2 network will become increasingly similar, whether using Q_1 or Q_2, this does not affect the update of the Actor network. We set to update the actor network every k steps, and the formula for updating the Actor network parameters is

$$\theta \leftarrow \theta + \beta \cdot \nabla_{\theta}J(\theta) \tag{21}$$

Finally, in order to ensure that the TD error remains small and make the training process more stable, the soft update and delayed update intervals are used to slowly update the respective target networks, i.e.

$$w_i' \leftarrow \rho w_i + (1-\rho)w_i', \quad i=1,2 \tag{22}$$

$$\theta' \leftarrow \rho\theta + (1-\rho)\theta' \tag{23}$$

where ρ represents the soft policy coefficient, and $0 < \rho << 1$. w_i' and θ' are parameters of the Critic and Actor target networks, respectively.

C. Algorithm Execution

The environment used for online training is not exactly the same as the environment used for offline deployment. Offline deployment is closer to real-world traffic scenarios and has integrity characteristics. The online training environment is more diverse, and it can also be considered that the environment where tasks are deployed offline is a subset of the online training environment.

The objective of online training is to achieve a proficient agent capable of handling diverse IoV scenarios. Moreover, the allocation strategies need to be specific and accurate for each time slot. Therefore, it is necessary to initialize the diversity of tasks in the training environment so that the agent can learn more experience and make smart strategies. In summary, the task initialization features in online training environments include [28]

- Random initialization of the stage. The initialization task can be randomly in the upload, computation, or download stage.
- Random initialization of state during the stage. The state of the initialization task at a certain stage is random, i.e. the remaining amount of data to be processed at that stage is random.
- Unify the arrival time of different tasks. Although initialization states and stages of collaborative sensing tasks are random, the arrival time of different tasks can still be considered the same. When external conditions remain consistent, they all have the same subsequent decision chain. This fully conforms to the definition of Markov processes regarding state transitions: the future is only about the present, not history [29].

Algorithm 1: Offline Deployment Testing.

```
1  Initialization: number of vehicles $N$, collaborative
     sensing task, $Q_{wait}$ and $Q_{exe}$.
2  for each time-slot t=1,...,$T$ do
3      for i=1,...,$N$ do
4          if $t$=$t_i$ then
5              Add $req_i$ into $Q_{wait}$.
6          end
7      end
8      Calculate the priority weight of tasks in $Q_{wait}$ by
         (17).
9      while 0 < length($Q_{exe}$) < $M$ do
10         Select $task_i$ from $Q_{wait}$ with maximum
             priority weight.
11         if $t_i \le t \le t_i + T_i^{max}$ then
12             Add $task_i$ into $Q_{exe}$ and remove $task_i$
                 from $Q_{wait}$.
13         end
14         else
15             Remove $task_i$ from $Q_{wait}$.
16         end
17     end
18     Obtain state $s(t)$ from $Q_{exe}$ and generate
         corresponding action $a(t)$ by the target actor
         network of the well-trained DRL agent.
19     Execute the resource allocation strategy according
         $a(t)$ and run the ISCC system for a time slot.
20     for $task_i$ in $Q_{exe}$ do
21         if $t \ge t_i + T_i^{max}$ or $task_i$ finish then
22             Remove $task_i$ from $Q_{exe}$.
23         end
24     end
25 end
```

The above setting details can not only eliminate the dependence on prior information of task arrival time, but also enhance the diversity of states in the training environment, enabling the agent to cope with various situations and enhance its knowledge reserve. The offline deployment testing environment completely simulates the arrival and completion process of every collaborative sensing task. The main details of offline testing are shown in Algorithm 1, where lines 2 to 17 represent the process of task scheduling between the waiting queue Q_{wait} and the execution queue Q_{exe}.

D. Complexity Analysis

The complexity of the proposed DRL algorithm mainly comes from the multiplication times of the neural network, assuming Γ_i^a represents the number of neurons in the i-th layer of the actor networks. Due to the same network structure as Critic 1 and Critic 2, using Γ_i^c represents the number of neurons in the i-th layer of the critic networks.

1) Training Complexity: Let E denotes the total number of training episodes, and I represents the number of steps per episode. During the training process, both the actor and the critic networks require training, with the most intuitive complexity arising from backpropagation. Additionally, the training process necessitates predictions from the target actor and critic networks.

Therefore, under normal circumstances, the complexity of a single backpropagation training step for the proposed DRL algorithm is

$$\mathcal{O}\left(\sum_{i=L_a}^{i=1} 2\Gamma_i^a\Gamma_{i+1}^a + \sum_{i=1}^{i=L_c-1} 4\Gamma_i^c\Gamma_{i+1}^c\right) \tag{24}$$

where L_a and L_c represent the number of layers in the actor and critic networks, respectively. The training process of the proposed algorithm commences only when the buffer size exceeds ϱB. Furthermore, the algorithm entails updating the actor network half as frequently as the critic network, with the actor network being updated $EI/2$ times. Consequently, the overall training complexity of the proposed algorithm is characterized by

$$\mathcal{O}\left(\varrho B \sum_{i=1}^{i=L_a-1} \Gamma_i^a\Gamma_{i+1}^a + (EI-\varrho B)\left(\frac{1}{2}B \sum_{i=1}^{i=L_a-1} \Gamma_i^a\Gamma_{i+1}^a \right.\right.$$
$$\left.\left. + \sum_{i=1}^{i=L_a-1} \Gamma_i^a\Gamma_{i+1}^a + B \sum_{i=1}^{i=L_a-1} 4\Gamma_i^c\Gamma_{i+1}^c\right)\right) \tag{25}$$

2) Testing Complexity: During the testing process, only the trained actor is utilized. For any given input environmental state, the trained actor network yields corresponding actions. Based on the analysis of training complexity, the complexity of the testing process from input to output can be expressed as

$$\mathcal{O}\left(\sum_{i=1}^{i=L_a-1} \Gamma_i^a\Gamma_{i+1}^a\right) \tag{26}$$

IV. Simulation

This section presents simulation results that validate the effectiveness of the proposed DRL resource allocation scheme in the context of the ISCC system, particularly in complex IoV scenarios. All training and testing experiments in this paper are completed on an Intel (R) Core (TM) i5-9300H CPU @ 2.40 GHz, 8 GB RAM, and the relevant software configuration version is Anaconda3-2021.11-Windows-x86 64, Python 3.8, TensorFlow 2.12, Keras 2.12. The fluctuation range of training duration under different strategies is mainly between four to eight hours.

In the experimental simulation, the uplink and downlink wireless resource bandwidth of the Roadside Unit (RSU) are set to 10 MHz and 5 MHz, respectively. The vehicle users' transmission power is set to 27 dBm, while the RSU's transmission power is set to 30 dBm. The data transmission follows the path loss model in the 3GPP TR 38.901 RMa scenario [30], which is 128.1+37.6$\log_{10}d$, where the distance d between the vehicle and the RSU is 35 m. The noise power spectral density is -174 dBm/Hz. The number of time slots for each episode is set to 100, which is the maximum number of steps, and the length of each time-slot is 10 ms. Based on the resources in our ISCC system, the length M of the task execution queue is set to 5. For the testing environment, the collaborative sensing task for each vehicle is set to arrive randomly within 0 to 500 ms. In addition, γ_1 and γ_2 are set to 5 and 1 respectively, the soft policy factor ρ is set to 0.05, the learning rates α and β for the critic and actor networks are both set to 0.0001, B is set to 128, and the discount factor γ is set to 0.98. The other environmental parameters of the IoV-oriented ISCC system are shown in Table I.

TABLE I
Environmental Parameters of the IoV-Oriented ISCC System

Parameter	Value
Length of execution queue M	5
Vehicle sensing data size D^V	50–150 Mbits
Computation data size I^c	50–150 Mcycles
Download data size I^d	5–15 Mbits
Uplink bandwidth of RSU R^u	10 MHz
Computing resources of MEC server R^c	20 GHz
Downlink bandwidth of RSU R^d	5 MHz
Computing resource of each vehicle f_v	2 GHz
Default compression coefficient r^{sep}	0.2

Following the convention observed by numerous mainstream actor-critic architecture DRL algorithms, the critic network is updated once when the agent interacts with the environment. In addition, for the sake of fairness, the training frequency of the actor network is set to update every two steps to avoid weakening the learning if the number of iterations is too small [27]. The update frequency of the target network for all experimental settings is set to 0.01, which means updating every 100 steps. According to our observation, due to the consistent training strategy, all schemes will converge at around 40000 episodes. Therefore, the total number of episodes in the experiment is set to 60000 for better observation results.

To evaluate the convergence and effectiveness of our proposed DRL-based resource allocation scheme, two sets of comparative experimental schemes are designed to verify the effectiveness.

1) Comparison of the effectiveness of actor network delay update policy. In alignment with the other specifications outlined in the proposed scheme,

- A scheme. There is no delay update policy for the actor network, but rather the use of the target networks.
- B scheme. There is the delay update policy for the actor network, but no target networks.
- C scheme. There are no actor network delay update policy and target networks.

2) Comparison of the effectiveness of double critic networks. The effectiveness of the double critic networks in the proposed scheme is evaluated by comparing it with the classic single critic network used in the DDPG algorithm. At the same time, two traditional baseline methods are also compared to verify the intelligence of the DRL adaptive resource allocation algorithm. The settings are as follows:

- DDPG scheme. Based on the original DDPG [31], we add the same actor network delay update policy as the proposed scheme, and the critic and actor network structures are consistent with the proposed scheme.
- Average allocation scheme. The uplink and downlink bandwidth and MEC server computing resources are allocated in an equal proportion.

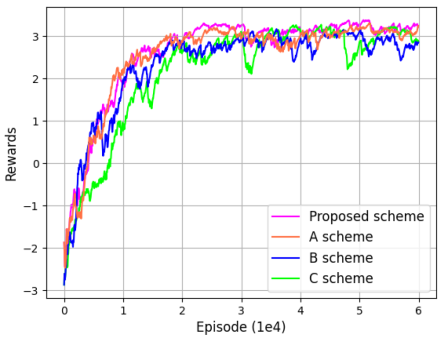

Fig. 4. The process of reward changes during training phase under different schemes.

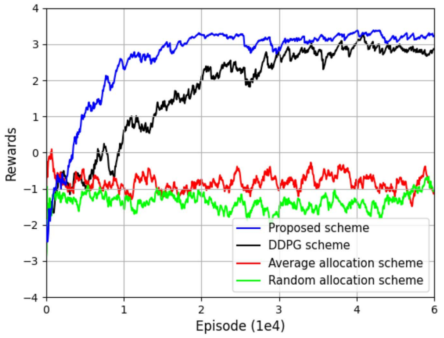

Fig. 5. Training phase reward value of the proposed vs. DDPG vs. baseline schemes.

- Random allocation scheme. The uplink and downlink bandwidth and MEC server computing resources are randomly allocated.

Fig. 4 shows the convergence of rewards during different training schemes. Firstly, B scheme shows significant shaking at locations such as 41000 and 51000, while C scheme shows significant shaking at locations such as 31000 and 48000. This is because the B and C schemes do not use target networks, resulting in training instability. The B scheme adopts an actor network delay update policy to avoid unstable replacement of actions caused by frequent updates. Therefore, compared to the C scheme, the reward trend is relatively stable. Due to the use of no target networks, B and C schemes may exhibit bootstrapping, resulting in a slightly lower convergence reward value compared to the proposed and A schemes.

In addition, all target networks of B and C schemes are updated simultaneously, which also incurs significant training costs in our actual simulation. The training duration is increased by an average of about 160% compared to the proposed scheme and A scheme. Although A scheme has almost similar rewards as the proposed scheme, it is worth noting that the number of times the proposed scheme trains the actor network is only half of the number of times that A scheme trains the actor network. This resulted in the 23.7% improvement in training efficiency compared to A scheme. Therefore, the proposed scheme has advantages in terms of performance and efficiency.

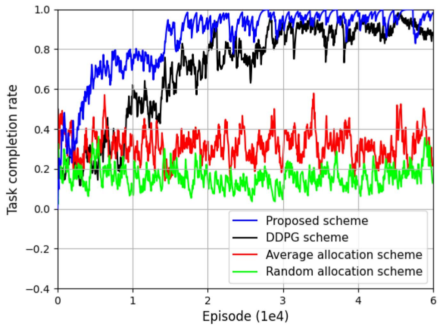

Fig. 6. Training phase task completion rate of the proposed vs. DDPG vs. baseline scheme.

Fig. 5 shows that when other conditions remain consistent, the comparison between the proposed scheme and the DDPG scheme, as well as the comparison with the two baseline schemes. It first can be seen that the proposed scheme outperforms the DDPG scheme in terms of convergence speed and performance. This is because the TD learning of the single critic network in DDPG scheme tends to maximization, leading to overestimation of the Q value, thereby affecting the final convergence performance. The proposed scheme uses the double critic networks, which can alleviate the overestimation problem caused by maximization and bootstrapping, thus achieving better performance. In addition, the reward convergence values based on DRL schemes are superior to the two baseline schemes. This indicates that DRL agents can learn smart strategies from complex IoV environments through training, thereby obtaining larger reward values. The average allocation scheme is rigid, and the random distribution scheme is arbitrary. No matter what situation they encounter, they only allocate system resources evenly or randomly, resulting in lower reward values.

Fig. 6 shows the convergence process of collaborative sensing task completion rate during the training process for the four schemes, which corresponds to the convergence process of reward values in Fig. 5. It is evident that the completion rate of the collaborative sensing task convergence for the proposed scheme approach 1, which is consistent with our optimization goal $\mathbb{P}_1$. Due to the overestimation problem, the convergence completion rate of the DDPG scheme is slightly lower than that of the proposed scheme. The completion rates of the average allocation scheme and the random allocation scheme have always been in the states of oscillation and non-convergence, and the completion rate is relatively low. This undoubtedly highlights the superiority of the DRL algorithm.

Fig. 7 shows the trend of training rewards under different lengths M of the execution queue Q_{exe}, with the condition that the resources of the ISCC system for IoV remain unchanged. For the experiment, M values of 2, 4, 5, 6, and 8 are chosen. When the value of M is equal to 5, which is the selected value of the proposed scheme, the convergence speed and final convergence performance are rewarded to achieve the best results. This indicates that the system can handle collaborative sensing tasks in parallel and the number of tasks that achieve optimal performance is 5. It can be seen that when the value of M is greater than 5, due to resource constraints, the agent cannot learn strategies with high reward values no matter how they

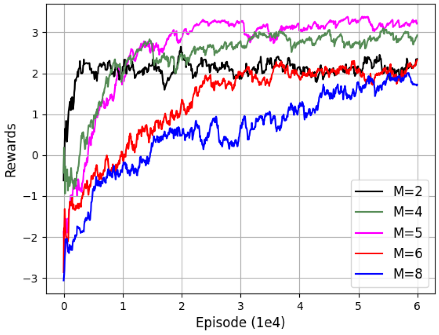

Fig. 7. Training reward trend under different lengths M of the execution queue Q_{exe}.

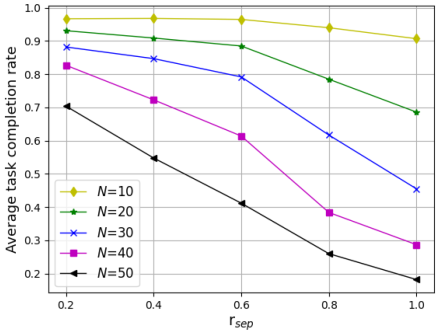

Fig. 9. Average task completion rate vs. different compression factors r_{sep}.

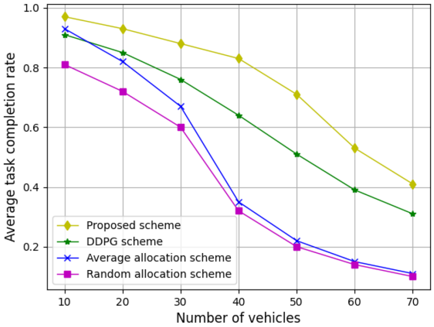

Fig. 8. Average task completion rate vs. number of vehicles N.

learn. When the value of M is less than 5, the phenomenon of remaining resources will occur when the number of parallel processing tasks is small, leading to low execution efficiency of the system and poor convergence reward value. Additionally, it can be observed from Fig. 7 that for values of M less than 5, the initial reward value is higher than or equal to 5. This is because the system has fewer parallel processing tasks under the same resources, so the average allocation of resources to each task will be more, resulting in a higher completion rate and a relatively larger reward value.

Fig. 8 shows the trend of the number of vehicles and collaborative sensing average task completion rate under different schemes, where the average of 100 test results is taken under the same conditions as the final performance to avoid the randomness of single round testing. When other conditions remain unchanged, the average completion rate of collaborative sensing tasks for all schemes decreases with the increase of the number of vehicles. Because the resources of the system are limited, making it impossible to process more collaborative sensing tasks simultaneously. Nevertheless, the proposed scheme achieves the highest task completion rate under varying number of vehicles, with 29.7%, 137.1%, and 159.4% higher completion rates compared to the other three schemes respectively, when the number of vehicles was 40. The DDPG scheme is superior to the baseline schemes, indicating that although the DDPG scheme has overestimation issues, it can still learn smarter strategies and adaptively allocate resources. The baseline schemes showed a sudden decrease when the number of vehicles is 40, and the completion rates are similar when the number of vehicles was 50, 60, and 70. It can be inferred from the analysis conducted that this situation arises when the two schemes reach their threshold in terms of task handling capacity. Consequently, the addition of further tasks often leads to timeouts. The trend of the DRL schemes is relatively smooth, which also highlights the robustness of the DRL schemes compared to the baseline schemes.

Fig. 9 shows the relationship between different compression factors and average completion rate, and different number of vehicles. As the compression factor increases, the average completion rate of collaborative sensing tasks shows a downward trend, while the average service time shows an upward trend. This is because the larger the compression factor, the more sensing data the vehicle needs to be uploaded, which results in the occupation of more wireless resources. If more vehicles send requests at a certain moment, it will cause wireless transmission congestion and more tasks waiting, thereby resulting in lower average completion rates and higher service time.

According to observations, when the number of vehicles N is 10, the change in average completion rate is relatively small under different compression factors. However, when N is equal to 20, the average completion rate shows a significant change. This indicates that, under the configured system resources, the system capacity, i.e., the number of collaborative sensing tasks that can be processed with a higher completion rate, falls within the range of 10 to 20. When the number of vehicles exceeds the threshold of system capacity, the average completion rate deteriorate significantly and are sensitive to compression factors.

V. Conclusion

In this paper, an IoV-oriented ISCC system was designed, and a collaborative sensing data fusion framework was proposed. The limited sensing range for a single vehicle was addressed by enabling vehicles to collaborate with RSU. The randomness of the arrival of vehicle collaborative sensing tasks in real-world traffic scenarios was simulated, with modeling of communication, sensing, and computation, respectively. An optimization problem was formulated to maximize the completion rate of collaborative sensing tasks. To solve the mixed-integer nonlinear optimization problem, a resource allocation scheme based on DRL was developed, which ensured optimal completion rate of vehicle collaborative sensing tasks while maintaining system service delay within acceptable limits. In the simulation experiment section, the reward design and training mechanism were continuously adjusted to enable the agent to learn the optimal resource allocation strategy from the complex IoV environment. The performance and robustness of the proposed scheme surpassed those of alternative schemes.

References

[1] J. Zhao, S. Ni, L. Yang, Z. Zhang, Y. Gong, and X. You, "Multiband cooperation for 5G HetNets: A promising network paradigm," *IEEE Veh. Technol. Mag.*, vol. 14, no. 4, pp. 85–93, Dec. 2019.

[2] W. Xu, Z. Yang, D. W. K. Ng, M. Levorato, Y. C. Eldar, and M. Debbah, "Edge learning for B5G networks with distributed signal processing: Semantic communication, edge computing, and wireless sensing," *IEEE J. Sel. Topics Signal Process.*, vol. 17, no. 1, pp. 9–39, Jan. 2023.

[3] G. Zhu et al., "Pushing AI to wireless network edge: An overview on integrated sensing, communication, and computation towards 6G," *Sci. China Inf. Sci.*, vol. 66, 2023, Art. no. 130301.

[4] J. Zhao, J. Liu, L. Yang, B. Ai, and S. Ni, "Future 5G-oriented system for urban rail transit: Opportunities and challenges," *China Commun.*, vol. 18, no. 2, pp. 1–12, Feb. 2021.

[5] F. Liu et al., "Integrated sensing and communications: Toward dual-functional wireless networks for 6G and beyond," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 6, pp. 1728–1767, Jun. 2022.

[6] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7944–7956, Aug. 2019.

[7] Y. He, G. Yu, Y. Cai, and H. Luo, "Integrated sensing, computation, and communication: System framework and performance optimization," *IEEE Trans. Wireless Commun.*, vol. 23, no. 2, pp. 1114–1128, Feb. 2024.

[8] L. Zhao, D. Wu, L. Zhou, and Y. Qian, "Radio resource allocation for integrated sensing, communication, and computation networks," *IEEE Trans. Wireless Commun.*, vol. 21, no. 10, pp. 8675–8687, Oct. 2022.

[9] X. Li, Y. Gong, K. Huang, and Z. Niu, "Over-the-air integrated sensing, communication, and computation in IoT networks," *IEEE Wireless Commun.*, vol. 30, no. 1, pp. 32–38, Feb. 2023.

[10] X. Li et al., "Integrated sensing, communication, and computation over-the-air: MIMO beamforming design," *IEEE Trans. Wireless Commun.*, vol. 22, no. 8, pp. 5383–5398, Aug. 2023.

[11] D. Wen et al., "Task-oriented sensing, computation, and communication integration for multi-device edge AI," *IEEE Trans. Wireless Commun.*, vol. 23, no. 3, pp. 2486–2502, Mar. 2024.

[12] P. Liu et al., "Toward ambient intelligence: Federated edge learning with task-oriented sensing, computation, and communication integration," *IEEE J. Sel. Topics Signal Process.*, vol. 17, no. 1, pp. 158–172, Jan. 2023.

[13] Q. Qi, X. Chen, A. Khalili, C. Zhong, Z. Zhang, and D. W. K. Ng, "Integrating sensing, computing, and communication in 6G wireless networks: Design and optimization," *IEEE Trans. Commun.*, vol. 70, no. 9, pp. 6212–6227, Sep. 2022.

[14] Y. Xu, T. Zhang, Y. Liu, and D. Yang, "UAV-enabled integrated sensing, computing, and communication: A fundamental trade-off," *IEEE Wireless Commun. Lett.*, vol. 12, no. 5, pp. 843–847, May 2023.

[15] S. Xu, Y. Du, J. Zhang, J. Liu, J. Wang, and J. Zhang, "Intelligent reflecting surface enabled integrated sensing, communication and computation," *IEEE Trans. Wireless Commun.*, vol. 23, no. 3, pp. 2212–2225, Mar. 2024.

[16] N. C. Luong et al., "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Commun. Surv. Tuts.*, vol. 21, no. 4, pp. 3133–3174, Fourthquarter 2019.

[17] H. Tataria, M. Shafi, A. F. Molisch, M. Dohler, H. Sjöland, and F. Tufvesson, "6G wireless systems: Vision, requirements, challenges, insights, and opportunities," *Proc. IEEE*, vol. 109, no. 7, pp. 1166–1199, Jul. 2021.

[18] Q. Liu, R. Luo, H. Liang, and Q. Liu, "Energy-efficient joint computation offloading and resource allocation strategy for ISAC-aided 6G V2X networks," *IEEE Trans. Green Commun. Netw.*, vol. 7, no. 1, pp. 413–423, Mar. 2023.

[19] Q. Zhang, Z. Chen, B. Xia, X. Jiang, and C. Xiong, "Design and optimization of edge computing for data fusion in V2I cooperative systems," in *Proc. IEEE/CIC Int. Conf. Commun. China*, 2020, pp. 466–471.

[20] W. Yue, C. Li, G. Mao, N. Cheng, and D. Zhou, "Evolution of road traffic congestion control: A survey from perspective of sensing, communication, and computation," *China Commun.*, vol. 18, no. 12, pp. 151–177, Dec. 2021.

[21] S. Markowitz, C. Snyder, Y. C. Eldar, and M. N. Do, "Multimodal unrolled robust PCA for background foreground separation," *IEEE Trans. Image Process.*, vol. 31, pp. 3553–3564, 2022.

[22] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[23] J. Zhao, X. Sun, X. Ma, H. Zhang, F. R. Yu, and Y. Hu, "Online distributed optimization for energy-efficient computation offloading in air-ground integrated networks," *IEEE Trans. Veh. Technol.*, vol. 72, no. 4, pp. 5110–5124, Apr. 2023.

[24] M. Riedmiller et al., "Learning by playing solving sparse reward tasks from scratch," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, vol. 80, pp. 4344–4353.

[25] J. Zhao, L. He, D. Zhang, and X. Gao, "A TP-DDPG algorithm based on cache assistance for task offloading in urban rail transit," *IEEE Trans. Veh. Technol.*, vol. 72, no. 8, pp. 10671–10681, Aug. 2023.

[26] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, and M. G. Bellemare, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[27] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, vol. 80, pp. 1587–1596.

[28] J. Zhao, H. Quan, M. Xia, and D. Wang, "Adaptive resource allocation for mobile edge computing in internet of vehicles: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 73, no. 4, pp. 5834–5848, Apr. 2024.

[29] S. N. Ethier and T. G. Kurtz, *Markov Processes: Characterization and Convergence*. Hoboken, NJ, USA: Wiley, 2009.

[30] 3rd Generation Partnership Project (3GPP), "Study on channel model for frequencies from 0.5 to 100 GHz, release 16,v16.1.0," 3GPP, Sophia Antipolis, France, Tech. Rep. TR 38.901, Dec. 2019.

[31] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

Junhui Zhao (Senior Member, IEEE) received the M.S. and Ph.D. degrees from Southeast University, Nanjing, China, in 1998 and 2004, respectively. From 1998 to 1999, he was with the Nanjing Institute of Engineers, ZTE Corporation, Shenzhen, China. In 2004, he was an Assistant Professor with the Faculty of Information Technology, Macao University of Science and Technology, Cotai, China, and continued there as an Associate Professor till 2007. He was also a short term Visiting Scholar with Yonsei University, Seoul, South Korea, in 2004, and Visiting Scholar with Nanyang Technological University, Singapore, from 2013 to 2014. In 2008, he was with Beijing Jiaotong University, Beijing, China, as an Associate Professor, where he is currently a Professor with the School of Electronics and Information Engineering. Since 2016, he has also been with the School of Information Engineering, East China Jiaotong University, Nanchang, China. His research interests include wireless and mobile communications and related applications.

Ruixing Ren received the B.E. degree in communication engineering from Changchun University, Changchun, China, in 2022. He is currently working toward the master's degree in information and communication engineering with East China Jiaotong University, Nanchang, China. His research interests include integrated sensing, communication, and computation, Internet of Vehicles, mobile edge computing, resource management, and channel estimation.

Dan Zou received the M.S. degree in computer application technology in 2008, from East China Jiaotong University, Jiangxi, China, where she is currently working toward the Ph.D. degree in control science and engineering. Her research interests include vehicular networks, mobile edge computing, and resource allocation.

Qingmiao Zhang received the M.S. degree in computer application technology in 2008, from East China Jiaotong University, Jiangxi, China, where she is currently working toward the Ph.D. degree in control science and engineering. Her research interests include train-ground communication technologies in communication-based train ground communication systems, and performance enhancements for wireless communication for train control.

Wei Xu (Senior Member, IEEE) received the B.Sc. degree in electrical engineering and the M.S. and Ph.D. degrees in communication and information engineering from Southeast University, Nanjing, China, in 2003, 2006, and 2009, respectively. Between 2009 and 2010, he was a Postdoctoral Research Fellow with the University of Victoria, Victoria, BC, Canada. He was an Adjunct Professor with the University of Victoria, from 2017 to 2020, and a Distinguished Visiting Fellow of the Royal Academy of Engineering, U.K., in 2019. He is currently a Professor with Southeast University. His research interests include information theory, signal processing, and machine learning for wireless communications.

Dr. Xu was the recipient of the Youth Science and Technology Award of China Institute of Communications in 2018, Science and Technology Award of the Chinese Institute of Electronics (Second Prize) in 2019, National Natural Science Foundation of China for Outstanding Young Scholars in 2020, IEEE Communications Society Heinrich Hertz Award in 2023, and Best Paper Awards at IEEE Globecom 2014, IEEE ICCC 2014, ISWCS 2018, and WCSP 2017, 2021. He was the Editor of IEEE COMMUNICATIONS LETTERS from 2012 to 2017, and the Editor of IEEE TRANSACTIONS ON COMMUNICATIONS from 2018 to 2023. He is the Senior Editor of IEEE COMMUNICATIONS LETTERS. He is a Fellow of IET.