Received 13 May 2021; accepted 28 May 2021. Date of publication 8 June 2021;
date of current version 28 June 2021. The review of this article was arranged by Guest Editor Yang Zhang.

*Digital Object Identifier 10.1109/OJVT.2021.3087355*

# Distributed Slice Selection-Based Computation Offloading for Intelligent Vehicular Networks

JIANHANG TANG [1], YIQUN DUAN [2], YI ZHOU [3], AND JIANGMING JIN [4]

[1] School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China
[2] CIBCI lab, Australian Artificial Intelligence Institute, School of Computer Science, University of Technology Sydney, Ultimo, NSW 2007, Australia
[3] School of Energy and Power Engineering, Wuhan University of Technology, Wuhan 430063, China
[4] HPC, TuSimple, San Diego, CA 92122 USA

CORRESPONDING AUTHOR: YI ZHOU (e-mail: zhouyi@whut.edu.cn)

**ABSTRACT** Distributed artificial intelligence (AI) is becoming an efficient approach to fulfill the high and diverse requirements for future vehicular networks. However, distributed intelligence tasks generated by vehicles often require diverse resources. A customized resource provision scheme is required to improve the utilization of multi-dimensional resources. In this work, a slice selection-based online offloading (SSOO) algorithm is proposed for distributed intelligence in future vehicular networks. First, the response time and energy consumption are reduced for processing tasks locally on the vehicles. Then, the offloading overheads, including latency and energy consumption, are calculated by considering the available resource amount, wireless channel states and vehicle conditions. The slice selection results is obtained by the deep reinforcement learning (DRL)-based method. Based on the selection solution, resource allocation results are achieved by KKT conditions and bisection method. Finally, the experimental results depict that the proposed SSOO algorithm outperforms other comparing algorithms in terms of energy consumption and task completion rate.

**INDEX TERMS** Resource slice, slice selection, computation offloading, distributed intelligence.

## I. INTRODUCTION

Artificial intelligence (AI) is becoming an efficient method to enhance future vehicular networks [1], [2]. With the feature of high mobility of vehicles, centralized AI models are not suitable for highly varying environments and diversified network architectures. Thus, distributed intelligence based frameworks are applied for systems with multiple vehicles. For instance, vehicles collect and process data for intelligence tasks to achieve autonomous control. It imposes a noticeable challenge on the vehicles with limited computation capacities to process the intelligence tasks. Cloud computing enhances the vehicles by providing a shared pool of storage and computation resources [3]. However, it is costly to offload intelligence tasks to the remote cloud. As a result, multi-access edge computing (MEC) can provide adequate resources near the vehicles [4]–[6].

In edge and cloud computing systems, sophisticated network management strategies are required to control the hierarchical network architecture comprehensively. Software-defined networking (SDN) can reduce network management costs and improve network flexibility [7]. SDN leverages the software-defined concepts to manage and control the network. Meanwhile, centralized SDN controllers are deployed to provide global views of the network. With network functions virtualization (NFV) [8] and SDN technologies, many service-oriented resource slices are offered for intelligence tasks over a common network infrastructure [9], [10].

Vehicles should select resource slices to offload the intelligence tasks to obtain the services provided by the edge layer. However, if the resources provided by resource slices are unable to fulfill the resource demands of tasks, the service qualities will be degraded. Moreover, resource utilization will be reduced by only considering the available resources of resource slices when tasks are offloaded. To address these issues, this work proposed a slice selection-based online offloading (SSOO) algorithm for distributed intelligence in future vehicular networks. The main contributions are listed as follows.

- A slice selection-based computation offloading problem for distributed AI is formulated by considering the available resource amount, wireless channel states and vehicle conditions. In the aforementioned problem, the objective function is to reduce the energy costs.
- A slice selection-based online offloading (SSOO) algorithm is proposed. A DRL-based method is leveraged to obtain the slice selection results, which can reduce the time complexity dramatically. Based on the selection solution, resource allocation results are achieved by KKT conditions and bisection method.
- We conduct extensive experiments in a testbed with six edge servers and real-world datasets. As observed from the experimental results, comparing with other baseline schemes, our SSOO algorithm can reduce the system energy costs and improve the task completion rate significantly.

The rest of the paper is organized as follows. Section II reviews the recent work. Section III proposes the system model of the slice selection-based computation offloading. Section IV presents the SSOO algorithm. In Section V, extensive experiments are conducted with six edge servers and real-world datasets. Finally, Section VI concludes the work.

## II. RELATED WORK

### A. DISTRIBUTED INTELLIGENCE

Distributed intelligence enables the effective management and orchestration of the multi-dimensional resources for future vehicular networks [11]–[15]. Yao *et al.* [11] proposed a cross-layer artificial intelligence-based architecture to fulfill the requirements on 5 G and beyond. For autonomous driving systems, base stations can learn the behaviors of vehicles to assist the transportation systems. Ning *et al.* [12] proposed an offloading scheme for vehicular networks by a DRL-based approach. The communication costs between the macrocell and vehicles were reduced significantly with the distributed DRL-based approach. Zhou *et al.* [13] proposed a secure computing framework, where the distributed intelligence system often suffers from byzantine attacks. In the proposed scheme, the blockchain technology is utilized to improve the security. Gopalswamy *et al.* [14] studied a novel distributed intelligence scheme for autonomous driving systems. In the proposed architecture, a bayesian network model is leveraged to assess the risks and benefits. Ioannou *et al.* [15] presented a decentralized intelligent algorithm to manage the generation of device-to-device (D2D) networks. The proposed algorithm can improve the data rate and reduce the energy costs.

Differ from the above work, this paper leverages the resource slices to offer customized resources for various distributed intelligence applications with different requirements.

### B. RESOURCE SLICING

Resource slicing is expected to provide customized resources for distributed intelligence applications [16]–[20]. Sun *et al.* [16] presented a dynamic resource slicing method for radio access networks. The virtual resources were controlled by the DRL-based algorithm to improve the average quality of service utility. Bega *et al.* [17] proposed an AI-based slicing framework, where AI technologies were introduced to improve the performance in the whole slicing life cycle. Van *et al.* [18] proposed a fast resource slicing architecture to maximize the long-term returns of network providers. The deep dueling Q-learning algorithm was leveraged to achieve the optimal results more quickly. Zhang *et al.* [19] studied a service-oriented soft resource slicing scheme for the vehicular networks. The resources were reused at inter-slice and intra-slice levels to improve resource utilization. Al-Khatib *et al.* [20] proposed a priority and reservation-based slicing scheme for different vehicular applications. The proposed resource slicing algorithm can improve network resource utilization.

Differ from the above work, this paper selects the optimal resource slices to offload intelligence applications in the vehicular networks to improve utilization of multi-dimensional resources.

### C. COMPUTATION OFFLOADING

DRL-based technologies are promising solutions for computation offloading in vehicular networks [21]–[25]. Ke *et al.* [21] proposed an adaptive offloading algorithm in heterogeneous vehicular networks, which was based on the DRL. The proposed algorithm considered the stochastic tasks and the variety of environments. Wang *et al.* [22] presented a mobility-aware partial offloading scheme in vehicular networks. Compared with full offloading, partial offloading can improve the flexibility of intelligence applications. Li *et al.* [23] presented a collaborative task offloading algorithm based on the DRL in vehicular networks. First, the execution order of a task was determined. Then, the task offloading and result delivery results were obtained by the DRL-based approach. Peng *et al.* [24] studied a distributed DRL-based resource management algorithm for vehicular networks with unmanned aerial vehicles. The centralized controller was not required to train the deep neural networks with multiple agents. Qiu *et al.* [25] presented a novel online offloading algorithm based on the DRL. The proposed algorithm can improve resource utilization of both mining and AI tasks in the blockchain-empowered edge computing environment.

Differ from the above work, the resource slices are leveraged for computation offloading. Each vehicle selects the optimal slice to conduct offloading to improve utilization of multi-dimensional resources in the vehicular networks.

## III. SYSTEM MODEL

### A. SYSTEM ARCHITECTURE

Fig. 1 describes the system architecture, including vehicles, edge servers, an SDN controller, and a slice orchestrator. Let $D_i$ indicate $i$-th vehicle, where $i \in M = \{1, 2, \ldots, NM\}$. Let $A_j$ denote $j$-th access point (AP), where $j \in O = \{1, 2, \ldots, NO\}$. Let $S_l$ represent $l$-th resource slice, where $l \in$

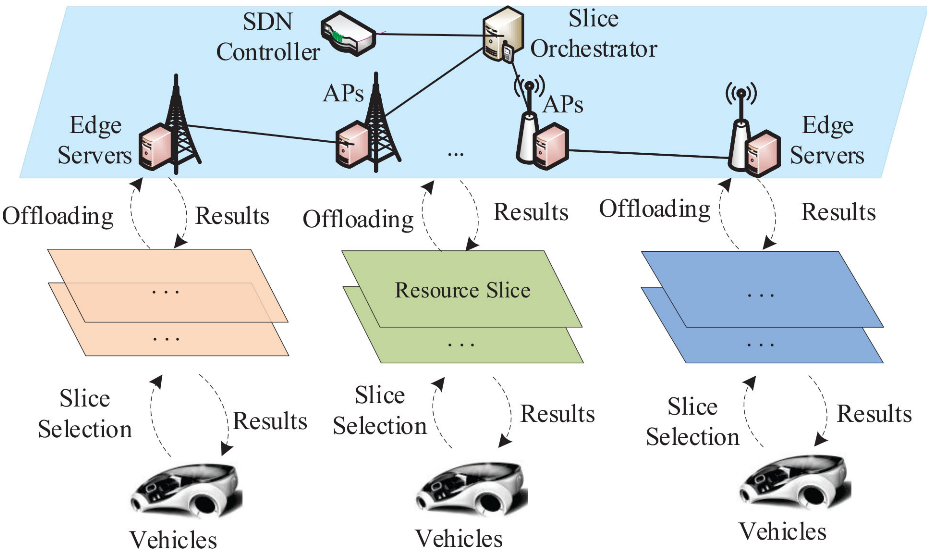


**FIGURE 1.** System architecture.

**TABLE 1.** Key Notations

| Notations | Descriptions |
|---|---|
| $x_i$ | An indicator variable to depict whether vehicle $D_i$ offload the AI task or not |
| $y_i^l$ | An indicator variable to depict whether vehicle $D_i$ select resource $S_l$ or not |
| $COM_l$ | The amount of computation resources of slice $S_l$ |
| $B_j^l$ | The bandwidth provided by AP $A_j$ in slice $S_l$ |
| $T_i^l$ | The execution delay of vehicle $D_i$ |
| $T_i^l$ | The execution energy costs of vehicle $D_i$ |
| $T_{i,j}^{tx,l}$ | The transmission delay of vehicle $D_i$ through AP $A_j$ in slice $S_l$ |
| $E_{i,j}^{tx,l}$ | The tranmission energy of vehicle $D_i$ through AP $A_j$ in slice $S_l$ |
| $T_i^{ex,l}$ | The execution delay in slice $S_l$ for vehicle $D_i$ |
| $E_i^l$ | The execution energy costs in slice $S_l$ for vehicle $D_i$ |

$P = \{1, 2, \ldots, NP\}$. The amount of computation resources of slice $S_l$ is denoted by $COM_l$ and the bandwidth of slice $S_l$ is denoted by $B_l$. The bandwidth provided by AP $A_j$ in slice $S_l$ is represented by $B_j^l$. Each vehicle generates an intelligence task with input $\lambda_i$ and deadline $\tau_i$. The tasks can be processed by vehicles locally or transmitted to the edge layer. The vehicle can process the task locally or offload the task to the edge layer. Let $x_i \in \{0, 1\}$ denote the indicator variable to depict whether the task is transmitted to the edge layer or not. Table 1 depicts the key notations.

### B. LOCAL COMPUTATION MODEL

If $x_i = 0$, the vehicle will execute the task locally. Let $Cyc_i$ indicate the required computation resources for processing one-bit task, which is measured by the number of clock cycles. The CPU-cycle frequency of vehicle $D_i$ is denoted by $f_i \in [0, f_{i,\max}]$, where $f_{i,\max}$ is the maximum value. Then, the task execution delay of vehicle $D_i$ is obtained as [26]

$$T_i^l = \frac{\lambda_i \cdot Cyc_i}{f_i}. \forall i \in M. \tag{1}$$

The energy costs of vehicle $D_i$ to process a task is calculated as [26]

$$E_i^l = \lambda_i \cdot Cyc_i \cdot C_i \cdot (f_i)^2, \forall i \in M. \tag{2}$$

where $C_i$ is the capacitance of vehicle $D_i$. As shown in Equations (1) and (2), the task execution overheads are mainly determined by the CPU-cycle frequency.

### C. SLICE SELECTION MODEL FOR COMPUTATION OFFLOADING

If a vehicle offloads a task to the edge layer, there are the following three steps. First, the vehicle should select a resource slice for computation offloading. Then, the task is transmitted and executed by the resources provided by the slice. Finally, the computed result is downloaded from the edge layer. We ignore the transmission latency of the computed results, since the size of the result is small enough [27], [28].

If $x_i = 1$, an indicator variable $y_i^l \in \{0, 1\}$ is introduced to show whether vehicle $D_i$ select resource slice $S_l$ to offload the task or not. In each slice, the wireless resources are offered by multiple APs. An AP can be accessed by various vehicles in a resource slice. Moreover, it can be accessed by vehicles in different resource slices. Thus, a binary variable $z_{i,j}^l$ is introduced to indicate whether vehicle $D_i$ select resource slice $S_l$ and use the wireless resources provided by AP $A_j$ or not. In the resource slice, a vehicle can only connect with an AP. Let $\rho_{i,j}^l$ denote the amount of wireless resources that are provided by AP $A_j$ and allocated to vehicle $D_i$ in resource slice $S_l$. Then, the achievable transmission rate for vehicle $D_i$ is achieved as [29], [30]

$$R_{i,j}^l = \rho_{i,j}^l \cdot B_{j,l} \log\left(1 + \frac{p_i \cdot h_{i,j}}{\sigma + \sum_{m\in M\setminus\{i\}:\sum_{l\in Q} z_{i,j}^l = 1} p_m \cdot h_{m,j}}\right), \tag{3}$$

where $B_{j,l}$ denotes the bandwidth, $p_i$ is the transmission power of vehicle $D_i$, $h_{i,j}$ is the channel gain between vehicle $D_i$ and AP $A_j$, and $\sigma$ represents the noise. As shown in Equation (3), the transmission rate is mainly calculated by the amount of wireless resources and the number of vehicles in the slice. Then, the task transmission delay of vehicle $D_i$ is achieved as

$$T_{i,j}^{tx,l} = \frac{\lambda_i}{R_{i,j}^l}, \forall R_{i,j}^l > 0. \tag{4}$$

The energy costs for task transmission is obtained as

$$E_{i,j}^{tx,l} = T_{i,j}^{tx,l} \cdot p_i, \forall R_{i,j}^l > 0. \tag{5}$$

When the task is transmitted to the edge layer, the selected slice will assign computation resources. As the amount of resources provided by an edge server is often limited, the edge servers cooperate in the slice. Let $\eta_i^l$ represent the amount of computation resources allocated to vehicle $D_i$ in slice $S_l$. Then, the task execution delay in resource slice $S_l$ is obtained as

$$T_i^{ex,l} = \frac{\lambda_i \cdot Cyc_i}{\eta_i^l \cdot Com_l}, \forall \eta_i^j > 0. \tag{6}$$

The energy consumption for task execution is achieved as

$$E_i^l = T_i^{ex,l} \cdot P_l, \forall \eta_i^j > 0, \tag{7}$$

where $P_l$ is the average task execution power of servers in resource slice $S_l$.

### D. PROBLEM FORMULATION

The energy consumption can be considered as a key metric in future vehicular networks. Then, the total energy consumption of vehicles are denoted as

$$E_1 = \sum_{i\in M}(1-x_i)E_i^l + \sum_{i\in M}\sum_{j\in O}\sum_{l\in P} x_i \cdot y_i^l \cdot E_{i,j}^{tx,l}. \tag{8}$$

The total energy consumption of the edge layer is represented as

$$E_2 = \sum_{i\in M}\sum_{l\in P} x_i \cdot y_i^l \cdot E_i^l. \tag{9}$$

Consequently, the resource slice-based computation offloading problem is formulated as

$$\textbf{P1}: \min \quad E = E_1 + E_2 \tag{10a}$$

$$\text{s.t. } 0 \le f_i \le f_{i,\max}, 0 \le p_i \le p_{i,\max}, \forall i \in M, \tag{10b}$$

$$0 \le \sum_{i\in M} \rho_{i,j}^l \le 1, \forall j \in O, l \in P, \tag{10c}$$

$$\rho_{i,j}^l \le z_{i,j}^l \le v_1 \rho_{i,j}^l, \forall i \in M, j \in O, l \in P, \tag{10d}$$

$$x_i \ge \sum_{l\in P} y_i^l, \forall i \in M \tag{10e}$$

$$y_i^l \ge \sum_{j\in O} z_{i,j}^l, \forall i \in M, l \in P, \tag{10f}$$

$$0 \le \sum_{i\in M} \eta_i^l \le 1, \forall l \in P, \tag{10g}$$

$$\eta_i^l \le y_i^l \le v_2 \eta_i^l, \forall i \in M, l \in P, \tag{10h}$$

$$(1-x_i)\cdot T_i^l + \sum_{j\in O}\sum_{l\in P} x_i \cdot y_i^l \cdot (T_{i,j}^{tx,l} + T_i^{ex,l}) \le \tau_i, \forall i \in M, \tag{10i}$$

$$x_i, y_i^l, z_{i,j}^l \in 0, 1, \forall i \in M, j \in O, l \in P, \tag{10j}$$

$$\rho_{i,j}^l, \eta_i^l \in [0, 1], \forall i \in M, j \in O, l \in P, \tag{10k}$$

where $p_{i,\max}$ is the maximum transmission power of vehicle $D_i$, $v_1$ and $v_2$ are arbitrarily large numbers. Constraint (10b) ensures limitations of the CPU-cycle frequencies and transmission powers. Constraints (10c)-(10h) guarantee that vehicles leverage the resources provided by the selected slices. Constraint (10i) ensures that tasks are processed before their deadlines. Then, the NP-hardness of problem **P1** is discussed in Lemma 1.

*Lemma 1:* **P1** is an NP-hard problem.

*Proof:* Assume that all vehicles offload the tasks, and transmission power is set to maximum value, there is only one AP, the deadlines of tasks are enough large, and the wireless and computation resources are allocated to vehicles equally. That is to say, it holds that $|O| = 1$, $x_i = 1$, $\tau_i = +\infty$, $p_i = p_{i,\max}$, $\rho_{i,1}^l = 1/a_1$, and $\eta_i^l = 1/a_2$, where $a_1$ and $a_2$ are constants. In this case, the energy consumption of vehicle $D_i$ for selecting slice $S_l$ can be calculated as

$$E_i^{l,*} = \frac{a_1 \cdot \lambda_i \cdot p_{i,\max}}{B_l \cdot \log\left(1 + \frac{p_{i,\max}\cdot h_i}{\sum_{m\in M\setminus\{i\}} p_{m,\max}\cdot h_m}\right)}. \tag{11}$$

Then, **P1** is reduced to the following problem:

$$\textbf{P2}: \min \quad \sum_{i\in M}\sum_{l\in P} y_i^l \cdot E_i^{l,*} \tag{12a}$$

$$\text{s.t. } \sum_{l\in P} y_i^l = 1, \forall i \in M, y_i^l \in \{0, 1\}, \forall i \in M, l \in P, \tag{12b}$$

$$\sum_{i\in M}\sum_{l\in P} y_i^l \cdot \rho_{i,1}^l \le 1, \sum_{i\in M}\sum_{l\in P} y_i^l \cdot \eta_i^l \le 1. \tag{12c}$$

Then, the new optimization problem **P2** is mapped to a multi-dimensional multi-choice knapsack problem (MMKP) [31]. As **P2** is NP-hard, **P1** is a generalization of the MMKP, which is also NP-hard. ■

## IV. THE SSOO ALGORITHM

In this section, we present a slice selection-based online offloading (SSOO) algorithm to minimize the system energy costs. As observed from **P1**, it is hard to deal with the binary variables with a large feasible region. In this work, a DRL algorithm is applied to achieve the resource slice selection problem [32]. The DRL algorithm can make decisions by an agent with a deep neural network (DNN). Then, the best slice selection results are attained by efficiently learning the current system environment. The input of the DNN is composed of wireless channel gains, input sizes and deadlines of tasks. This input is collected by the SDN controller. In time slot $t$, the channel gain set is denoted by $H_t = \{h_{i,j(t)} | i \in M, j \in O\}$, the task input set is represented by $\lambda(t) = \{\lambda_i(t) | i \in M\}$, and the deadline set is defined as $\tau(t) = \{\tau_i | i \in M\}$, where $t = 1, 2, \cdots$. The slot length is defined as $T$. To guarantee the quality of services, it holds that $T \leq \max\{\tau_1, \tau_2, \ldots, \tau_{NM}\}$. The initial parameter setting of the DNN is denoted by $\theta_1$. The output of DNN is the relaxed slice selection set, which is denoted by $\overline{Z} = \{\overline{z}^l_{i,j} \in [0, 1], i \in M, j \in O, l \in P\}$. For the DNN, the activation functions of both hidden and output layers are rectified linear unit (ReLU) and sigmoid function respectively. After achieving the relaxed slice selection set, these relaxed decisions should be transformed into integral solutions, as follows.

$$g_K : \overline{Z}(t) \leftarrow \left\{Z_k | Z_k \in 0, 1^N, k = 1, 2, \ldots, K\right\}, \tag{13}$$

where $Z_k$ denotes the slice selection sets, $K \in [1, 2_N]$, and $N = NM \cdot NP \cdot NO$ represents the size of decision space. To reduce the complexity of the proposed algorithm, it holds that $1 \leq K \leq N + 1$. If $\overline{z}^l_{i,j} \geq \overline{z}^{l'}_{i,'j'}$, it holds that $z^{l,k}_{i,j} \geq z^{l,'k'}_{i,'j'}$. The iterative process for obtaining slice selection decision set is summarized as follows.

- First, the initial slice selection decision is obtained according to the output of the DNN, as follows.

$$z^{l,1}_{i,j} = \begin{cases} 1, & \overline{z}^l_{i,j} \geq 0.5, \\ 0, & \overline{z}^l_{i,j} \leq 0.5, \end{cases} \tag{14}$$

  where $i \in M$, $j \in O, l \in P$.
- Then, the relaxed solutions are sorted by the following rule.

$$\left|\overline{z}_{(1)}(t) - 0.5\right| \leq \cdots \leq \left|\overline{z}_{(n)}(t) - 0.5\right| \leq \cdots \left|\overline{z}_{(N)}(t) - 0.5\right|, \tag{15}$$

  where $\overline{z}_{(n)}(t)$ is the $n$-th relaxed slice selection result.
- Finally, the other $K - 1$ slice selection results are achieved according to the aforementioned sequence, as follows.

$$z^{l,k}_{i,j} = \begin{cases} 1, & \overline{z}^l_{i,j}(t) \geq \overline{z}_{(k-1)}(t), \\ 1, & \overline{z}^l_{i,j}(t) = \overline{z}_{(k-1)}(t) \text{ and } \overline{z}_{(k-1)}(t) \leq 0.5, \\ 0, & \overline{z}^l_{i,j}(t) = \overline{z}_{(k-1)}(t) \text{ and } \overline{z}_{(k-1)}(t) > 0.5 \\ 0, & \overline{z}^l_{i,j}(t) < 0.5, \end{cases} \tag{16}$$

  where $i \in M$, $j \in O, l \in P$.

After obtaining the slice selection decisions, **P1** is divided into tree sub-problems, including the resource allocation problem for vehicles, the wireless resource allocation problem, and the resource allocation problem for the edge layer. Let $f(t) = \{f_i(t) | 0 \leq f_i(t) \leq f_{i,\max}, \forall i \in M\}$ denote the resource allocation decision set for vehicles. Let $p_(t) = \{p_i(t) | 0 \leq p_i(t) \leq p_{i,\max}, \forall i \in M\}$ be the transmission power set for vehicles. Let $\eta(t) = \{\eta^l_i(t) | \eta^l_i(t) \in [0, 1], i \in M, l \in P\}$ indicate the resource allocation decision set for the edge layer. Let $\rho(t) = \{\rho^l_{i,j}(t) | \rho^l_{i,j}(t) \in [0, 1], i \in M, j \in O, l \in P\}$ represent the wireless resource allocation decision set. When slice selection decision $Z_k$ is given, a resource allocation decision $\{f_k(t), p_k(t), \rho_k(t), \eta_k(t)\}$ and the energy consumption $E_k$ can be obtained. In time slot $t$, the optimal resource allocation decision, $\{Z^*(t), f^*(t), p^*(t), \rho^*(t), \eta^*(t)\}$, is the one with minimal energy consumption.

If $\sum_{j \in O} \sum_{l \in P} z^{l,k}_{i,j} = 0$, vehicle $D_i$ will process the task locally. As shown in equations (1) and (2), these two equations are convex functions of $f_i$. Thus, we use the Karush-Kuhn-Tucker (KKT) conditions as an efficient method for computation resource management problem of vehicles.

If If $\sum_{j \in O} \sum_{l \in P} z^{l,k}_{i,j} = 1$, vehicle $D_i$ will offload task to the edge layer. As observed from equations (6) and (7), these two equations are convex functions of $\eta^l_i$. Similarly, we use the KKT conditions as an efficient method for computation resource management problem of the edge layer. Then, the joint transmission setting and wireless resource management problem is formulated as

$$\textbf{P3}: \min \sum_{i \in M} \sum_{j \in O} \sum_{l \in P} E^{tx,l}_{i,j} \tag{17a}$$

$$\text{s.t. } \rho^l_{i,j} \in [0, 1], \forall i \in M, j \in O, l \in P, \tag{17b}$$

$$0 \leq p_i \leq p_{i,\max}, \forall i \in M. \tag{17c}$$

It is difficult to solve the optimal problem **P3** due to equation (3). To simplify the problem, we use the following equation instead of equation (3).

$$R^l_{i,j} = \rho^l_{i,j} \cdot B_{j,l} \log\left(1 + \frac{p_i \cdot h_{i,j}}{\sigma + \sum\limits_{m \in M \setminus \{i\} : \sum_{l \in Q} z^l_{i,j} = 1} p_{m,max} \cdot h_{m,j}}\right). \tag{18}$$

Equation (18) is a quasi-convex function of $\rho^l_{i,j}$ and $p_i$. The optimal wireless allocation and transmission power setting results are calculated by the bisection method, which can be denoted by $\rho^*_k(t)$ and $p^*_k(t)$.

Finally, optimal slice selection solution $Z^*(t)$ is obtained with minimum energy consumption. Moreover, the data $\{H(t), \lambda(t), \tau(t), Z^*(t)\}$ is added to the sample dataset. The parameters of the DNN are updated periodically by selecting a training dataset $A(t)$. In this work, the mean square error is used as the loss function. To update the parameters of the DNN, we use Adam as an optimizer.

**Algorithm 1:** The SSOO Algorithm.

**Input:** Wireless channel gain $H(t)$, task input size $\lambda(t)$, and task deadline $\tau(t)$

**Output:** Offloading result $\{Z^*(t), f^*(t), p^*(t), \rho^*(t), \eta^*(t)\}$

1: Initialize the parameter $\theta_1$ of the DNN;
2: Set the maximum iteration number $\zeta$ and training interval $\delta$;
3: **for** $t = 1, 2, \ldots, \zeta$ **do**
4:     Obtain relaxed slice selection solution $\overline{Z}^{(}t)$ according to $H(t)$, $\lambda(t)$, and $\tau(t)$;
5:     Generate the initial slice selection result;
6:     Generate $K-1$ slice selection results according to the initial result;
7:     **for** $k = 1, 2, \ldots, N+1$ **do**
8:         Obtain computation resource allocation solutions $f_k(t)$ and $\eta_k(t)$ with KKT conditions;
9:         Obtain the wireless resource allocation solution $\rho_k(t)$ and transmission power setting result $p_k(t)$ with bisection method;
10:         Calculate the system energy consumption $E_k$;
11:     **end for**
12: **end for**
13: Select the optimal offloading result $\{Z^*(t), f^*(t), p^*(t), \rho^*(t), \eta^*(t)\}$ with minimum energy consumption;
14: Add $\{H(t), \lambda(t), \tau(t), Z^*(t)\}$ to the sample dateset;
15: **if** $t \mod \delta = 0$ **then**
16:     Select a training dataset $A(t)$ randomly;
17:     Obtain the optimal parameter $\theta_t^*$ with MSE and Adam optimizer;
18: **end if**
19: **return** $\{Z^*(t), f^*(t), p^*(t), \rho^*(t), \eta^*(t)\}$

Algorithm 1 summarizes the aforementioned ideas. First, the relaxed slice selection solution is achieved by wireless channel gains, input data sizes, and task deadlines (line 4). Then, $K$ slice selection results are obtained based on the relaxed slice selection solution (line 5-line 6). With each slice selection result, the optimal resource allocation solution is achieved. The slice selection and resource allocation result with minimum energy consumption is the optimal offloading solution (line 7-line 11). Finally, the parameters of the DNN are updated periodically (line 12-line18).

As shown in Algorithm 1, the time complexity of obtaining the relaxed slice selection result is $O(NM \cdot NP \cdot NO)$. The time consumption for achieving the initial slice selection solution is also $O(NM \cdot NP \cdot NO)$, while the time complexity of generating other $K-1$ slice selection results is $O(K-1)$. The time complexity of calculating the computation resource allocation results is $O(NM \cdot NP) + O(NM)$. The time consumption of calculating the wireless resource allocation and transmission power setting results is $O(NM \cdot NP \cdot NO)$. In summary, the total complexity of the SSOO algorithm is $O(NM \cdot NP \cdot NO)$

**TABLE 2.** Configurations of End Devices and Servers

| Device Type | CPU | Memory |
|---|---|---|
| ES 1~2 | Intel(R) Xeon(R) Gold 5117 CPU @ 2.00GHz | 4GB |
| ES 3~4 | Intel(R) Core(TM) i5-2450M CPU @ 2.50 GHz | 4GB |
| ES 5~6 | Intel(R) Core(TM) i5-4210M CPU @ 2.60GHz | 4GB |
| Dell Inspiron 5000 Fit | Intel Core i7-10510U @ 1.80GHz | 8GB |
| Lenovo Xiaoxin Pro 13 | Intel Core i5-10210U @ 1.6GHz | 8GB |
| Xiaomi 9 | Snapdragon 855 | 6GB |
| Hongmi 9 | Snapdragon 660 | 3GB |
| Galaxy Note 8 | Snapdragon 835 | 6GB |
| Honor 9i | Kirin 659 | 4GB |

## V. EVALUATION

### A. EXPERIMENTAL ENVIRONMENT

The experimental environment includes some end devices, six edge servers (ESs), an SDN controller and a slice orchestrator. The configurations of end devices and servers are shown in Table 2. The SDN controller is installed based on OpenDaylight. The ESs are controlled by Kubernetes. Finally, an End-to-End Orchestrator [33] is installed as a slice orchestrator. Fig. 2 describes the detailed experimental environment.

We apply vehicle detection and viewpoint prediction tasks as the benchmark tasks to depict the benefits of the proposed SSOO algorithm for distributed intelligence. The input of the vehicle detection tasks is the KITTI dataset, while the input of viewpoint prediction tasks is the video viewing dataset.

### B. BASELINE SCHEMES AND EVALUATION METRICS

The proposed SSOO algorithm is compared with the following three baseline schemes, including the energy and time efficient task offloading and resource allocation (ETCORA) algorithm [34], deep reinforcement learning-based computation offloading and resource allocation (DRL-CORA) algorithm [35], and genetic algorithm-based network slice selection (GA-NSS) algorithm [36].

In experiments, evaluation metrics are composed of average device energy consumption, total energy consumption, task completion rate and the number of tasks processed by slices. The average device energy costs are measured by the amount of energy consumed by the devices for executing and transmitting tasks. The system energy costs are determined by the amount of energy consumed by both end devices and edge

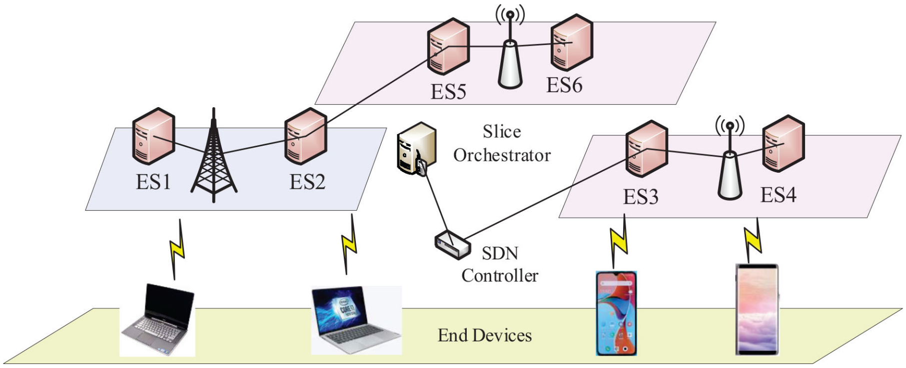


**FIGURE 2.** Experimental environment.

servers. The task completion rate is calculated by the number of tasks that are completed within the deadlines. The number of tasks processed by slices is achieved by the number of tasks received by each slice.

### C. BASIC SETTINGS

The basic settings of the experiment are shown in this subsection. The number of resource slices for vehicle detection tasks is 2, including 28 vCPU cores and 20 MHz bandwidth. The number of resource slices for viewpoint prediction tasks is 2, including 14 vCPU cores and 40 MHz bandwidth. In each experimental group, average result values are calculated to reduce the randomness impacts. For the DNN, the number of hidden layers is 2.

### D. IMPACTS OF NUMBERS OF END DEVICES

In this experimental group, the deadlines of vehicle detection and viewpoint prediction tasks are set to 0.8 s and 1 s respectively. The inputs of these two tasks follow exponential distributions with the means of 16 MB and 32 MB. Then, how the number of end devices affects system performance is investigated.

As shown in Fig. 3(a), the total energy costs increase with the increase of the number of end devices. This is since more energy is required for processing more tasks. When the number of devices is 5, there are no significant differences among these four schemes. Compared with GA-NSS algorithm, DRL-CORA algorithm and GA-NSS algorithm, the proposed SSOO algorithm can reduce the total energy consumption by 25.56%, 39.12% and 50.84% when the number of devices is 25. Fig. 3(b) plots the increasing trends of the average device energy consumption. This is because more energy is needed for transmitting tasks to the edge layer with the increase of the number of end devices. When the number of devices is 25, compared with GA-NSS algorithm, the SSOO algorithm can reduce the average device energy consumption by 22.31%. This is because the SSOO algorithm can improve resource utilization for distributed intelligence by selecting optimal slices.

As observed from Fig. 4, the task completion rate reduces when the number of devices increases. This is due to the fact that the amount of resources provided by the system is limited. It is hard to offer adequate resources for each task. Compared with the ETCORA scheme, the SSOO algorithm can increase the task completion rate by 17.69% when the number of devices is 15. This is because the ETCORA algorithm cannot provide customized resources for tasks with various requirements, which results in lower task completion rates. As shown in Fig. 5, the number of tasks processed by

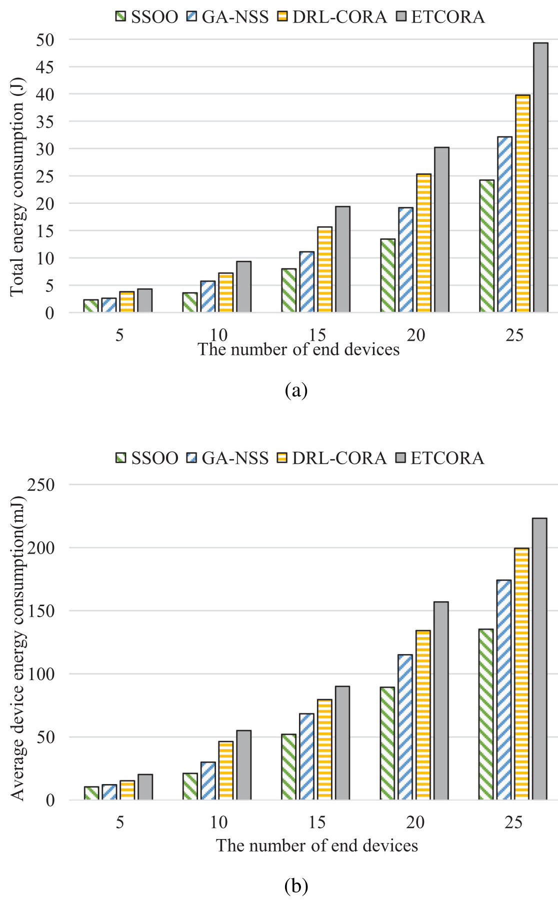


**FIGURE 3.** Impacts of numbers of end devices on (a) the total energy consumption, (b) the average device energy consumption.

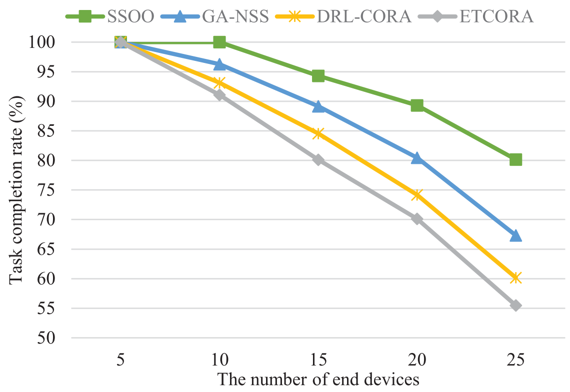


**FIGURE 4.** Impacts of numbers of end devices on the task completion rate.

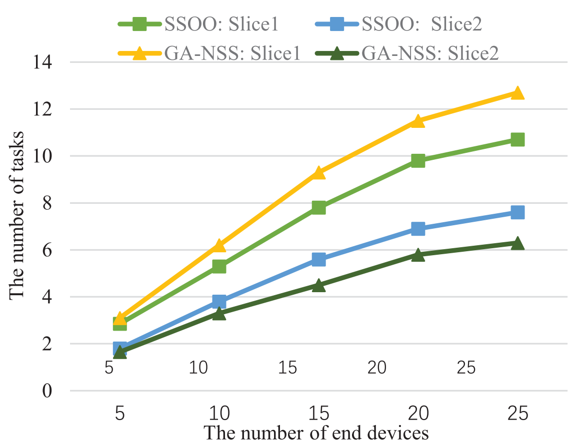


**FIGURE 5.** Impacts of numbers of devices on the number of tasks processed by each slice.

each slice increases when the number of devices increases. Under the proposed SSOO algorithm, the ratio of the number of tasks processed by Slice 1 to that of tasks processed by Slice 2 is 3 : 2.

### *E. IMPACTS OF TASK DEADLINES*

In this experimental group, the number of devices is 20. The inputs of these two tasks follow exponential distributions with the means of 16 MB and 32 MB. Then, how task deadline affects system performance is studied.

Fig. 6(a) plots the decreasing trends of the total energy consumption when the task deadline increases. This is because the tasks with larger deadlines can be processed with fewer resources. When the deadline is set to 1.2 s/1.4 s, there is not any difference between the GA-NSS algorithm and SSOO algorithm. When the deadline is set to 0.4 s/0.6 s, compared with the GA-NSS algorithm, DRL-CORA algorithm and ETCORA algorithm, the SSOO algorithm can reduce the total energy consumption by 30.25%, 42.16% and 49.82%. As shown in Fig. 6(b), the average device energy consumption increases rapidly with the decrease of the task deadlines. Compared with the GA-NSS algorithm, our proposed SSOO algorithm

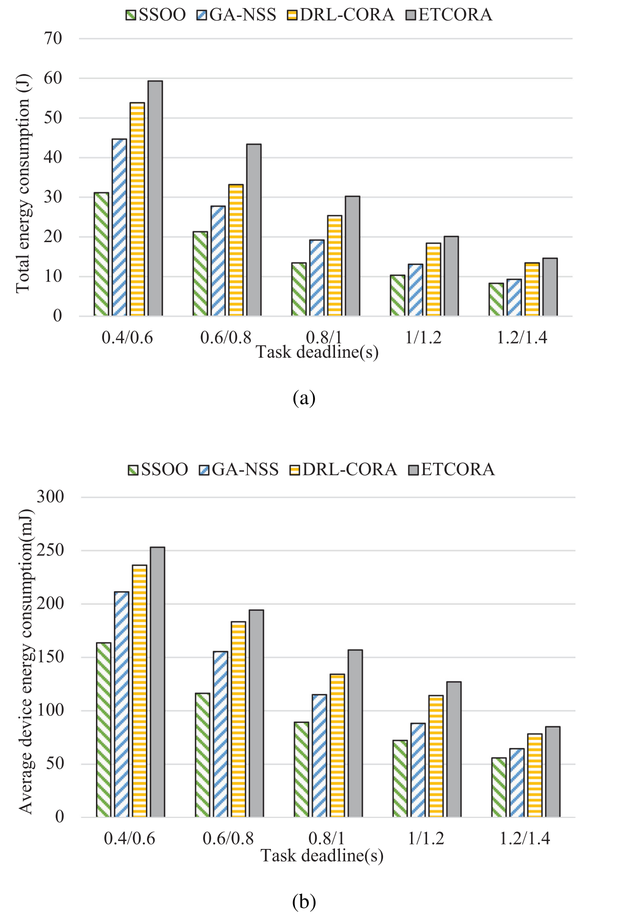


**FIGURE 6.** Impacts of task deadlines on (a) the total energy consumption, (b) the average device energy consumption.

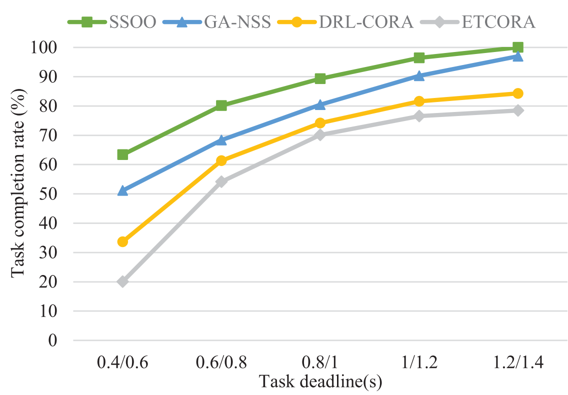


**FIGURE 7.** Impacts of task deadlines on the task completion rate.

reduces the average device energy consumption by 13.49% when the deadline is set to 1.2 s/1.4 s. This is because the SSOO algorithm can improve the resource utilization of each slice.

As shown in Fig. 7, the task completion rate acts as an increasing function of the task deadline. When the deadline is set to 1.2 s/1.4 s, the task completion rates under both

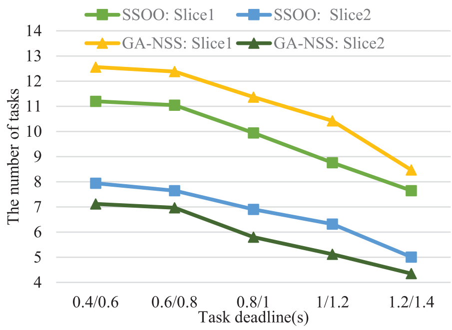


**FIGURE 8.** Impacts of task deadlines on the number of tasks processed by each slice.

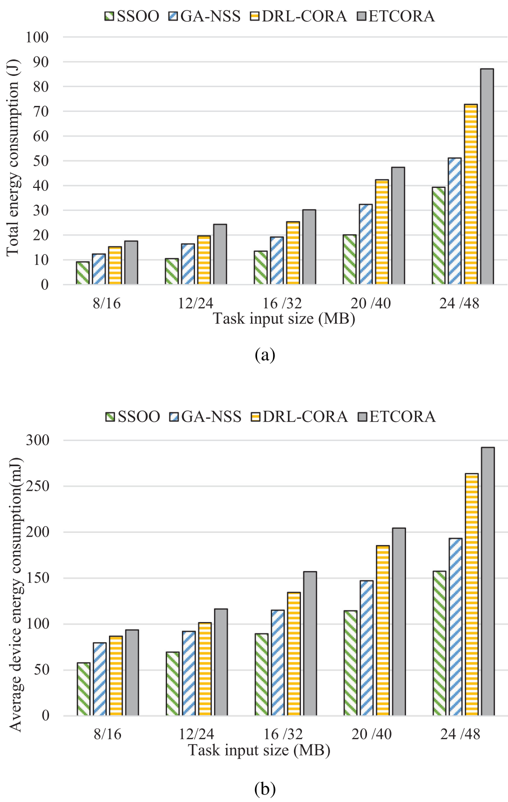


**FIGURE 9.** Impacts of task input sizes on (a) the total energy consumption, (b) the average device energy consumption.

GA-NSS algorithm and SSOO algorithm are near 100%. This is because the resource slice can improve resource utilization. When the deadline is set to 0.8 s/1s, compared with the ETCORA algorithm, the proposed SSOO algorithm can improve the task completion rate by 27.33%. Fig. 8 plots a decreasing trend of the task completion rate when the task deadline increases. With larger deadlines, more tasks can be finished within the deadlines.

### *F. IMPACTS OF TASK INPUT SIZES*

In this experimental group, the deadlines of vehicle detection and viewpoint prediction tasks are set to 0.8 s and 1 s

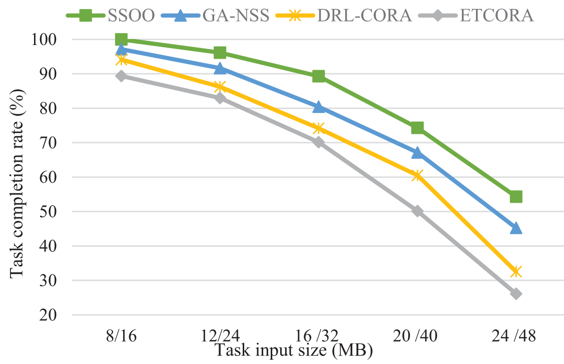


**FIGURE 10.** Impacts of task input sizes on the task completion rate.

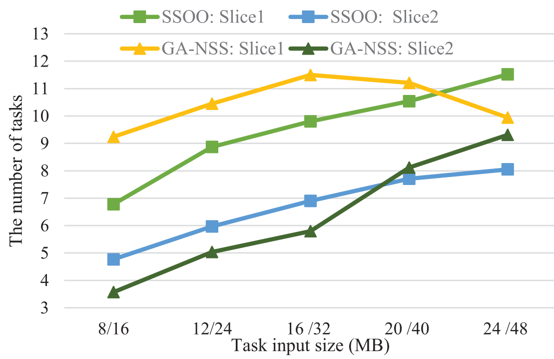


**FIGURE 11.** Impacts of task input sizes on the number of tasks processed by each slice.

respectively. The number of end devices is set to 20. Then, how the task input sizes affect the system performance is investigated.

As shown in Fig. 9(a), the total energy consumption increases when the task input size increases. This is because more network and computation resources are required to process these tasks. When the input size is set to 8 MB/16 MB, there are no significant differences among all schemes. This is because all tasks can be processed within the deadlines. Compared with the GA-NSS algorithm, the SSOO algorithm can reduce the total energy consumption by 23.06% when the input size is set to 24 MB/48 MB. As observed from Fig. 9(b), the average device energy consumption acts as an increasing function of the task input sizes. This is because end devices need more resources to process the tasks with larger inputs.

Fig. 10 plots a decreasing trend of the task completion rate when the task input sizes increase. This is due to the fact that fewer tasks are finished within deadlines with larger inputs. When the input size is set to 8 MB/16 MB, the performance of all schemes is near. When the input size is set to 24 MB/48 MB, the performance of the GA-NSS algorithm and SSOO algorithm is better than that of DRL-CORA algorithm and ETCORA algorithm. In Fig. 11, the ratio of task numbers in Slice 1 and Slice 2 is near 3:2 under the proposed SSOO algorithm, while the ratio of task numbers in Slice 1 and Slice 2 changes under the GA-NSS algorithm. This is because the proposed SSOO algorithm can select the optimal slices for tasks.

## VI. CONCLUSION

In this work, the SSOO algorithm for distributed intelligence tasks in future vehicular networks is proposed. First, the dynamic voltage and frequency scaling technology is applied to reduce the response time and energy consumption for processing tasks locally. Then, the response time and energy consumption for offloading are calculated by considering the available resource amount, wireless channel states and vehicle conditions. A DRL-based method is leveraged to obtain the slice selection results. Based on the selection solution, resource allocation results are achieved by KKT conditions and bisection method. Finally, the experimental results depict that the proposed SSOO algorithm outperforms other comparing algorithms in terms of energy consumption and task completion rate. In future studies, more virtual network functions of the resource slices will be included and discussed for the distributed AI.

## REFERENCES

[1] K. Zhang, S. Leng, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, “Artificial intelligence inspired transmission scheduling in cognitive vehicular communications and networks,” *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1987–1997, Apr. 2018.

[2] R. Q. Hu, L. Hanzo, and L. Hanzo, “Twin-timescale artificial intelligence aided mobility-aware edge caching and computing in vehicular networks,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3086–3099, Apr. 2019.

[3] A. H. Sodhro, S. Pirbhulal, and V. H. C. De Albuquerque, “Artificial intelligence-driven mechanism for edge computing-based industrial applications,” *IEEE Trans. Ind. Inform.*, vol. 15, no. 7, pp. 4235–4243, Jul. 2019.

[4] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, “Edge intelligence: Paving the last mile of artificial intelligence with edge computing,” *Proc. IEEE*, vol. 107, no. 8, pp. 1738–1762, 2019.

[5] Y. Liu, M. Peng, G. Shou, Y. Chen, and S. Chen, “Toward edge intelligence: Multiaccess edge computing for 5G and Internet of Things,” *IEEE Internet Things J.*, vol. 7, no. 8, pp. 6722–6747, Aug. 2020.

[6] X. Wang, Y. Han, V. C. Leung, D. Niyato, X. Yan, and X. Chen, “Convergence of edge computing and deep learning: A comprehensive survey,” *IEEE Commun. Surv. Tut.*, vol. 22, no. 2, pp. 869–904, Apr.–Jun. 2020.

[7] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, Q. Zhang, and K.-K. R. Choo, “An energy-efficient SDN controller architecture for iot networks with blockchain-based security,” *IEEE Trans. Serv. Comput.*, vol. 13, no. 4, pp. 625–638, Jul./Aug. 2020.

[8] I. Alam *et al.*, “A survey of network virtualization techniques for internet of things using SDN and NFV,” *ACM Comput. Surv.*, vol. 53, no. 2, pp. 1–40, 2020.

[9] M. Hu, T. Shen, J. Men, Z. Yu, and Y. Liu, “Crsm: An effective blockchain consensus resource slicing model for real-time distributed energy trading,” *IEEE Access*, vol. 8, pp. 206 876–206 887, 2020.

[10] J. Feng, Q. Pei, F. R. Yu, X. Chu, J. Du, and L. Zhu, “Dynamic network slicing and resource allocation in mobile edge computing systems,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7863–7878, Jul. 2020.

[11] M. Yao, M. Sohul, V. Marojevic, and J. H. Reed, “Artificial intelligence defined 5 G radio access networks,” *IEEE Commun. Mag.*, vol. 57, no. 3, pp. 14–20, Mar. 2019.

[12] Z. Ning *et al.*, “When deep reinforcement learning meets 5G-enabled vehicular networks: A distributed offloading framework for traffic big data,” *IEEE Trans. Ind. Inform.*, vol. 16, no. 2, pp. 1352–1361, Feb. 2020.

[13] S. Zhou, H. Huang, W. Chen, P. Zhou, Z. Zheng, and S. Guo, “Pirate: A blockchain-based secure framework of distributed machine learning in 5G networks,” *IEEE Netw.*, vol. 34, no. 6, pp. 84–91, Nov./Dec. 2020.

[14] S. Gopalswamy and S. Rathinam, “Infrastructure enabled autonomy: A distributed intelligence architecture for autonomous vehicles,” in *Proc. IEEE Intell. Veh. Symp.*, 2018, pp. 986–992.

[15] I. Ioannou, V. Vassiliou, C. Christophorou, and A. Pitsillides, “Distributed artificial intelligence solution for D2D communication in 5G networks,” *IEEE Syst. J.*, vol. 14, no. 3, pp. 4232–4241, Sep. 2020.

[16] G. Sun, Z. T. Gebrekidan, G. O. Boateng, D. Ayepah-Mensah, and W. Jiang, “Dynamic reservation and deep reinforcement learning based autonomous resource slicing for virtualized radio access networks,” *IEEE Access*, vol. 7, pp. 45 758–45 772, 2019.

[17] D. Bega, M. Gramaglia, A. Garcia-Saavedra, M. Fiore, A. Banchs, and X. Costa-Perez, “Network slicing meets artificial intelligence: An AI-based framework for slice management,” *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 32–38, Jun. 2020.

[18] N. Van Huynh, D. T. Hoang, D. N. Nguyen, and E. Dutkiewicz, “Optimal and fast real-time resource slicing with deep dueling neural networks,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1455–1470, Jun. 2019.

[19] S. Zhang, H. Luo, J. Li, W. Shi, and X. Shen, “Hierarchical soft slicing to meet multi-dimensional QoS demand in cache-enabled vehicular networks,” *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2150–2162, Mar. 2020.

[20] A. A. Al-Khatib and A. Khelil, “Priority-and reservation-based slicing for future vehicular networks,” in *Proc. IEEE Conf. Netw. Softwarization: Bridg. Gap Between AI Netw. Softwarization*, 2020, pp. 36–42.

[21] H. Ke, J. Wang, L. Deng, Y. Ge, and H. Wang, “Deep reinforcement learning-based adaptive computation offloading for MEC in heterogeneous vehicular networks,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7916–7929, Jul. 2020.

[22] J. Wang, T. Lv, P. Huang, and P. T. Mathiopoulos, “Mobility-aware partial computation offloading in vehicular networks: A deep reinforcement learning based scheme,” *China Commun.*, vol. 17, no. 10, pp. 31–49, 2020.

[23] M. Li, J. Gao, L. Zhao, and X. Shen, “Deep reinforcement learning for collaborative edge computing in vehicular networks,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 4, pp. 1122–1135, Dec. 2020.

[24] H. Peng and X. Shen, “Multi-agent reinforcement learning based resource management in mec-and uav-assisted vehicular networks,” *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 131–141, Jan. 2021.

[25] X. Qiu, L. Liu, W. Chen, Z. Hong, and Z. Zheng, “Online deep reinforcement learning for computation offloading in blockchain-empowered mobile edge computing,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8050–8062, Aug. 2019.

[26] W. Zhang, Z. Zhang, S. Zeadally, H.-C. Chao, and V. C. Leung, “Energy-efficient workload allocation and computation resource configuration in distributed cloud/edge computing systems with stochastic workloads,” *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1118–1132, Jun. 2020.

[27] S. Bi and Y. J. Zhang, “Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading,” *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4177–4190, Jun. 2018.

[28] Y. Mao, J. Zhang, and K. B. Letaief, “Dynamic computation offloading for mobile-edge computing with energy harvesting devices,” *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.

[29] Q. Ye, W. Zhuang, S. Zhang, A.-L. Jin, X. Shen, and X. Li, “Dynamic radio resource slicing for a two-tier heterogeneous wireless network,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 10, pp. 9896–9910, Oct. 2018.

[30] R. Riggio, A. Bradai, D. Harutyunyan, T. Rasheed, and T. Ahmed, “Scheduling wireless virtual networks functions,” *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 2, pp. 240–252, Jun. 2016.

[31] A. Mkaouar, S. Htiouech, and H. Chabchoub, “Solving the multiple choice multidimensional knapsack problem with ABC algorithm,” in *Proc. IEEE Congr. Evol. Comput.*, 2020, pp. 1–6.

[32] L. Huang, S. Bi, and Y.-J. A. Zhang, “Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks,” *IEEE Trans. Mobile Comput.*, vol. 19, no. 11, pp. 2581–2593, Nov. 2020.

[33] J. X. Salvat, L. Zanzi, A. Garcia-Saavedra, V. Sciancalepore, and X. Costa-Perez, “Overbooking network slices through yield-driven end-to-end orchestration,” in *Proc. Int. Conf. Emerg. Netw. EXp Technol.*, 2018, pp. 353–365.

[34] H. Sun, H. Yu, G. Fan, and L. Chen, “Energy and time efficient task offloading and resource allocation on the generic IoT-fog-cloud architecture,” *Peer-to-Peer Netw. Appl.*, vol. 13, no. 2, pp. 548–563, 2020.
[35] J. Li, H. Gao, T. Lv, and Y. Lu, “Deep reinforcement learning based computation offloading and resource allocation for MEC,” in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*., 2018, pp. 1–6.
[36] G. Zhao, S. Qin, G. Feng, and Y. Sun, “Network slice selection in softwarization-based mobile networks,” *Trans. Emerg. Telecommun. Technol.*, vol. 31, no. 1, 2020, Art. no. e3617.

**JIANHANG TANG** received the M.S. degree in applied statistics from Lanzhou University, Lanzhou, China, in 2015 and the Ph.D. degree in computer science from the Wuhan University of Technology, Wuhan, China. He was a Visiting Ph.D. Student with the School of Engineering, The University of British Columbia (Okanagan Campus), Kelowna, BC, Canada. He is currently a Lecturer with Yanshan University, Qinhuangdao, China. His research interests include edge computing and edge intelligence.

**YIQUN DUAN** received the B.Eng. degree in electrical engineering from Hohai University, Nanjing, China and the M.A.Sc. degree in electrical engineering from The University of British Columbia, Vancouver, BC, Canada. He is currently working toward the Ph.D. degree with CIBCI lab, Australian Artificial Intelligence Institute, School of Computer Science, University of Technology Sydney, Ultimo, NSW, Australia. His current research interests involve vision and language scenery understanding for embodied robotic system and brain computer interface.

**YI ZHOU** received the B.Eng. degree in aeronautics engineering and the M.Eng. degree in power engineering from the Beijing University of Aeronautics and Astronautics, Beijing, China, in 2008 and 2010, respectively, and the Ph.D. degree from Nanyang Technological University, Singapore, in 2015. She is currently an Associate Professor with the School of Energy and Power Engineering. Her current research interests include intelligent industrial systems, micro- and nano-scale fluid mehcanics, dynamic systems, and optimization methods.

**JIANGMING JIN** graduated from Nanyang Technological University, Singapore, in 2013, He received the bachelor’s degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2008 and the Ph.D. degree in computer engineering. He started his career with J. P. Morgan (Singapore & Beijing). In J. P. Morgan, he worked with the most sophisticated and large-scale financial computing system and also worked with JPM Credit Portfolio Group in credit derivative calculations. In 2017, he begins a venture journey in autonomous driving with TuSimple (NASDAQ: TSP). In TuSimple, he was the Director of HPC and Senior Expert of Engineering to oversee projects of high performance robotic middleware and heterogeneous computing systems. He has wide and deep knowledge in Hi-Tech areas including Fintech, AI chips and software, big data and machine learning systems, 5G, and IoT. In these area, he has authored or coauthored more than 20 research papers in top conferences or journals and applied more than ten US/CN patents. He was also the Session-Chair or keynote-speaker in several AI Summits/Forums. He is an Executive Member of CCF-TCHPC and CCF-TCARCH. He also holds several advisory positions in Universities and Non-Profit Institute, including Shanghai Tech University, Shanghai, China, Tianjin University, Beijing, China, and Shenzhen Fintech Association.