Dynamic Satellite Edge Computing Offloading Algorithm Based on Distributed Deep Learning

Jiaqi Shuai, Haixia Cui, *Senior Member, IEEE*, Yejun He, *Senior Member, IEEE*, and Mohsen Guizani, *Fellow, IEEE*

***Abstract*—Satellite communication networks with the characteristics of wide coverage, high-deployment flexibility, and seamless communication services can provide communication services to users who do not communicate with ground networks but directly communicate with satellites. In response to the increasing demand for user services, this article proposes a collaborative computing offloading scheme for satellite edge computing networks with a four-layer architecture. By utilizing collaborative computing between ground users and three layers of satellites (low-orbit satellites, edge, and cloud data centers), the service quality for ground users is improved. Considering the mobility of vehicles and satellite nodes, the frequent changes in link states further complicate the design and implementation of such systems, leading to increased latency and energy consumption. This article proposes to optimize the computation offloading decision while satisfying the constraint of satellite computing capabilities, aiming to improve the success rate of tasks and minimize the overall cost of the system. However, with the increase in the number of ground users and satellites, the formulated problem becomes a mixed-integer nonlinear programming (MINLP) problem, which is difficult to solve with general optimization algorithms. To address this issue, this article proposes a distributed deep learning-based dynamic offloading (DDLDO) algorithm based on distributed deep learning. The algorithm utilizes multiple parallel deep neural networks (DNNs) to dynamically learn computation offloading strategies. Simulation results demonstrate that the algorithm outperforms other benchmark algorithms in terms of latency, energy consumption, and successful execution efficiency.**

***Index Terms*—Computation offloading, deep neural networks (DNNs), edge computing, high dynamic, satellite networks.**

I. Introduction

GROUND mobile communication technology has developed rapidly in recent years, bringing many emerging applications and posing new challenges to existing networks. Traditional ground communication networks have limited coverage and are difficult to address key issues, such as geographical dispersion, communication interruptions, and remote areas [1]. They are unable to meet the global demand for "ubiquitous connectivity" [2]. In order to overcome these shortcomings of ground communication networks, satellite communication networks with the extensive coverage and high reliability have flourished to provide global Internet services to users. Satellite communication has been used in military, civilian, and commercial applications, especially in LEO (low-Earth orbit) satellite communication networks. LEO satellite communication networks have advantages, such as low-orbit height, short transmission delay, and low-path loss [3]. In recent years, some companies, such as SpaceX, Amazon, and OneWeb, have implemented their own low-Earth orbit communication satellite constellations [4]. It can be seen that the satellite-terrestrial integrated network (STIN) is a solution to address the growing business needs of users, with features, such as global coverage, high-deployment flexibility, and seamless communication services [5]. The satellite network and the ground network complement each other, jointly building an integrated information network, making STIN a critical part of future mobile communications.

The number of mobile users is continuously increasing with the flourishing development of the Internet. Some emerging applications that are computationally intensive and latency-sensitive consume a large amount of network resources and require low latency. Therefore, cloud computing may not be able to meet stringent requirements [6]. For example, during vehicle travel, one or more computationally intensive or latency-sensitive applications may be running simultaneously (such as navigation and autonomous driving under dynamically changing traffic conditions) [7]. Furthermore, not all mobile devices have the computational capability to execute computationally intensive and latency-sensitive tasks. Offloading tasks to cloud computing centers can result in high-communication latency [8]. In addition, in some areas with communication interruptions or in remote areas, mobile devices may exceed the communication range. Factors, such as bandwidth, environment, and technology, may preclude complete reliance on cloud computing [7]. In remote areas with limited coverage of ground communication networks, the tasks of mobile users need to be relayed through satellite-to-ground links to ground cloud servers for execution. Due to the constraints of satellite altitude, the communication latency of

Manuscript received 16 February 2024; revised 13 April 2024, 30 April 2024, and 10 May 2024; accepted 18 May 2024. Date of publication 24 May 2024; date of current version 8 August 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2023YFE0107900; in part by the National Natural Science Foundation of China under Grant 61871433, Grant 61828103, Grant 61201255, and Grant 62071306; in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2024A1515012052; and in part by the Research Platform of South China Normal University in Foshan. *(Corresponding author: Haixia Cui.)*

Jiaqi Shuai and Haixia Cui are with the School of Electronics and Information Engineering, South China Normal University, Foshan 528225, China (e-mail: 2022024822@m.scnu.edu.cn; cuihaixia@m.scnu.edu.cn).

Yejun He is with the College of Electronics and Information Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: heyejun@126.com).

Mohsen Guizani is with the Machine Learning Department, Mohamed Bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE (e-mail: mguizani@ieee.org).

Digital Object Identifier 10.1109/JIOT.2024.3404830

satellite-to-ground link is low, making it difficult to support latency-sensitive applications for high-speed mobile users [3]. Therefore, we need to explore new solutions, such as satellite edge computing. Taking inspiration from ground mobile edge computing (MEC) [9], MEC technology is being incorporated into satellite communication networks. Satellite edge computing provides computing services close to users or data inputs, where computing tasks are migrated to a location closer to the user from the original cloud computing center [10]. Users can forward tasks to satellites for processing, allowing users at the network edge to access computing resources from satellite edge nodes. The satellite communication network provides computing services to ground users and can also forward tasks to other satellites. By leveraging the advantages of high, medium, and low-Earth orbit satellites, enhancing data transmission performance to ensure real-time processing of computational tasks. Simultaneously, it can reduce frequent space-to-ground link transmissions [3], decrease the computational load on ground clouds, and optimize limited computing resources.

The satellite MEC (SMEC) network is seen as a crucial research focus for future network deployment [11]. When large-scale computations or complex tasks need to be performed, deploying appropriate satellites based on task characteristics and device performance can improve task execution efficiency and satisfy the high-quality communication requirements of computationally intensive or latency-sensitive applications [12]. This can enhance the efficiency and performance of satellite communication. Additionally, limited channel capacity will result in significant transmission latency between satellites and the ground [13]. Therefore, research on task offloading schemes is crucial in the satellite edge computing environment. Zhang et al. [14] proposed the three-layer satellite model with a computational offloading scheme using the DDPG algorithm. Qin et al. [15] investigated LEO satellite computational offloading under a hybrid computing offloading architecture with centralized training and distributed execution to minimize cost. Han et al. [16] proposed using RL algorithm to address the sensitive task offloading problem in satellite network architecture. Song et al. [17] presented the new terrestrial-satellite IoT MEC framework and adopted the energy-efficient computation offloading algorithm. Wei et al. [18] introduced a weighted comprehensive greedy strategy for task deployment to reduce the communication cost of the system. However, there are still some challenges regarding computation offloading in SMEC networks. The network shows a high degree of complexity, with the topology changing over time, and the relay hops between satellites at different locations can be considerable. Therefore, optimal path selection for task offloading becomes difficult. Additionally, the storage and computational capabilities of each satellite are limited, requiring rational allocation of tasks based on these resources. Furthermore, the status of ground users is dynamic and variable; for example, in the case of continuous vehicle movement, the distance between the vehicle and different satellites keeps changing, leading to potentially unstable communication connections. Moreover, the time spent by vehicles in the satellite communication coverage area varies with the vehicle's speed, resulting in changes in edge computing and resource scheduling [7]. Thus, satellite mobility and user mobility are indispensable parts of satellite edge computing and resource scheduling in dynamic and complex ground traffic environments. In the satellite network proposed by [14], computational resources between satellites can be shared through intersatellite links (ISLs). Both [15] and [16] only considered single-layer satellite networks and did not take into account cooperative computation between satellites through ISLs. In [15], satellites are considered quasi-static, neglecting task switching between satellites. The impact of the continuously changing topology of the satellite network on task offloading path selection was also overlooked. Some existing task offloading methods generally perform poorly in dynamic and uncertain systems. Furthermore, Qin et al. [15], Han et al. [16], and Song et al. [17] only considered static ground users and did not account for the influence of ground user mobility on task offloading decision-making. In [16], ultradense LEO satellites were deployed to ensure that each user is always covered by multiple satellites, neglecting the limited coverage time of satellites, which changes with ground user speed and affects optimal offloading path selection.

Deep learning utilizes multilayered deep neural networks (DNNs) to acquire data representations. When solving the computation offloading problems, breaking dimensionality constraints has become one of the research hotspots. The potential of deep learning in MEC opens up effective avenues for us. Typically, networks have high-dimensional state-action spaces. The DNNs approximate these relationships to achieve near-optimal results [19]. The algorithm introduced in [20] utilizes deep learning for vehicle edge computing offloading. The designed computation offloading optimization objective is considered to be NP-hard. Traditional algorithms [21], due to the dimensional curse, are not suitable for solving such problems, especially in the case of large-scale users. To approach optimal performance using traditional algorithms [22], [23], multiple iterations are required. It is unable to support time-sensitive applications generated by ground users such as vehicles. Most existing MEC network research is based on deep Q-networks, and its iterative process involves exhaustive decision-making and faces challenges in handling high-dimensional space problems [24], [25].

This article proposes a collaborative computation offloading strategy for a four-layer SMEC network. Ground vehicles can execute tasks locally, offload tasks to LEO satellite fog nodes, forward tasks to edge servers, or execute tasks in cloud computing centers. This method takes into account the exchange of information between satellites through ISLs and designs collaborative computation methods. Heavily loaded satellites can forward tasks to lightly loaded satellites or satellites from other layers for processing through ISLs [26]. This method can optimize limited computational resources. Taking into account the mobility of satellites and vehicular users, this article proposes a distributed deep learning-based dynamic offloading (DDLDO) algorithm for task offloading in this environment. The DDLDO algorithm dynamically adjusts offloading decisions. The communication status, speed, and position of vehicle users are constantly changing, and some traditional task offloading methods typically perform poorly in such dynamic and uncertain systems. Therefore, compared to traditional algorithms, this algorithm has advantages in real network environment. As the scale of the MEC system grows, the task offloading decision set grows exponentially. To find the optimal offloading decision, we propose the DDLDO algorithm, which utilizes K parallel DNN to obtain offloading decisions. These decisions are stored in a memory structure for training and improvement. In brief, the difference between our work and the related works is summarized in Table I. The contributions of this article are as follows.

1) A task offloading scheme is proposed in a four-layer satellite edge computing network. This article utilizes the powerful resources of edge and cloud servers to perform computations on satellites. Satellites through ISLs to achieve collaborative processing, balance the computational workload and optimize limited computational resources.
2) Due to the mobility of ground vehicle users, various situations arise, such as the changing distance between vehicles and satellites, going beyond communication range, unstable communication connections, and varying time spent by vehicles in different satellite coverage areas, due to changes in vehicle speed. These factors have an impact on edge computing and resource scheduling. Therefore, this article considers the mobility of vehicles. The fast convergence performance in the simulation experiment indicates that the proposed DDLDO algorithm can adapt to the dynamic environments and mitigate the impact of environment dynamics.
3) This article investigates the optimization problem with the objective of minimizing the weighted total cost of system. A dynamic offloading scheme based on distributed deep learning is proposed. Simulation experiments demonstrate that proposed solution outperforms baseline algorithms.

The remainder of this article is organized as follows. Section II focuses on the system model and problem formulation. Section III introduces distributed deep learning-based dynamic offload. Section IV gives the simulation results. Section V is the conclusion drawn from this article's research.

II. System Model and Problem Statement

In this section, we introduce the system model of the STIN under consideration, including the network model, coverage time model, mobility model, communication model, and computation model.

A. Network Model

Fig. 1 shows our considered network architecture for satellite edge computing, which includes cloud computing centers, edge data centers, satellite mist nodes, and vehicles (terrestrial users). The computational capability increases from the mist nodes to the cloud. The cloud computing center is constituted by 18 geostationary satellites, which have the highest orbit and computational capability. The edge data center is constituted by 24 medium-orbit satellites. The satellite fog nodes are comprised of 1000 low-orbit satellites, all of which have the lowest computational capability [18]. The satellite mist nodes deploy tasks based on the specified task offloading strategy. Computational tasks are sent to LEO satellites (Mist nodes) for execution through satellite-to-terrestrial links. Alternatively, LEO satellites (Mist nodes) can collaborate with cloud, edge data centers, or mist through ISLs to collectively process the tasks.

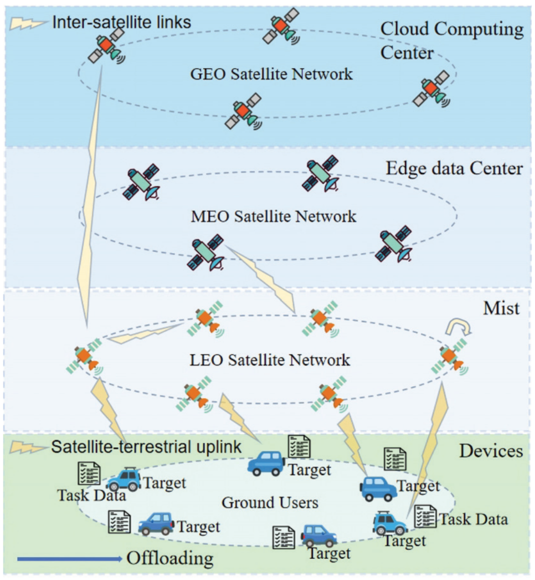


Fig. 1. Satellite-terrestrial network architecture.

The task list includes its parameters and its set can be represented as $G = \{g_1, g_2, \dots, g_k\}$, where each task g_k generated by each ground user $u \in \{0, 1, \dots, \Gamma\}$ can be described as two parts, namely, $d = \{m_g, n_g\}$. Here, m_g represents data volume of computing tasks, n_g indicates latency sensitivity of tasks. Let $X_g = \{a_g, b_{g,1}, \dots, b_{g,F_1}, c_{g,1}, \dots, c_{g,F_2}, d_{g,1}, \dots, d_{g,F_3}\}$ represent the computing offloading vector of ground user u, and $X = \{X_g, g \in G\}$ represent the computing offloading decisions for all ground users. F_1, F_2, F_3, F_4, and G, respectively, represent the number of LEO satellites, edge data centers, cloud data centers, and tasks. The offloading decision of each task g_k is represented as $a_g, b_g, c_g, d_g \in \{0, 1\}$. $a_g, b_g, c_g, d_g = 1$ indicates that the task g_k is executed locally, in LEO satellites, in edge data centers, and in cloud data centers, respectively. $a_g, b_g, c_g, d_g = 0$ indicates on the contrary.

Meanwhile, it should be noted that a single satellite has a limited coverage range in a mobile state, which leads to some changes in the network topology over time. As a result, the communication conditions between ground mobile users and satellite nodes are affected by the time of satellite network coverage [3]. Table II summarizes the main notations used in the remainder of this article.

B. Coverage Duration Model

1) For Satellite-to-Terrestrial Links: Due to the dynamic nature of satellites, the real-time connections between LEO satellites and users are subject to the geographical height

TABLE I
COMPARISON BETWEEN OUR WORK AND RELATED WORKS

Reference	LEO satellite computing	MEO Satellite(Edge Data Center) Computing	GEO Satellite(Cloud Data Center) Computing	Collaborative computing between multiple satellites	Dynamic users and satellites	The limited coverage time of the satellite
[3]	✓	×	×	✓	×	✓
[7]	×	×	×	×	✓	×
[13]	✓	×	×	×	×	×
[14]	✓	×	×	✓	×	×
[15]	✓	×	×	×	×	×
[16]	✓	×	×	×	×	×
[17]	✓	×	×	✓	×	×
[18]	✓	✓	✓	✓	×	×
Our work	✓	✓	✓	✓	✓	✓

TABLE II
KEY NOTATIONS

G	The set of tasks.
u	The set of vehicles.
m_g	The data size of task g_k.
n_g	The latency sensitivity of task g_k.
v_i, v_f	The satellite nodes.
s_u	The distance from vehicle u to LEO satellite.
h	The distance from vehicle u to LEO satellite orbit.
γ	The geocentric angle.
θ	The elevation angle between the vehicle u and LEO satellite.
R_{u,v_i}^{up}	The uplink transmission rate between the vehicle u and LEO satellite.
g_{u,v_i}	The channel power gain.
p_u	The uplink transmission power.
B	The available bandwidth.
m_u	The required CPU cycles of computation task g_k.
f_u^{local},f_{u,v_i}^{LEO}, $f_{vi,vf}^{LEO}$,f_{v_i,v_f}^{Edge},f_{v_i,v_f}^{Cloud}	The computation capacity of vehicle u, LEO satellite, Edge Data Center and Cloud.
T_u^{local},E_u^{local}	The delay and energy consumption of computation task g_k computed locally.
T_{u,v_i}^{LEO},E_{u,v_i}^{LEO},T_{v_i,v_f}^{LEO},E_{v_i,v_f}^{LEO}	The delay and energy consumption of computation task g_k computed at LEO satellites.
T_{v_i,v_f}^{Edge},E_{v_i,v_f}^{Edge}	The delay and energy consumption of computation task g_k computed at Edge Data Center.
T_{v_i,v_f}^{Cloud},E_{v_i,v_f}^{Cloud}	The delay and energy consumption of computation task g_k computed at Cloud Data Center.
d	$d = \{m_g, n_g\}$.
a_g, b_g, c_g, d_g	Whether/ where to offload the task of vehicle u
X	The offloading decision set for all $\{a_g, b_g, c_g, d_g\}, g \in G$.

restrictions. Therefore, the communication between LEO satellites and ground users is only possible under some specific conditions.

Fig. 2 illustrates the link relationship between ground users and LEO satellites, where R_{earth} represents the radius of Earth, s_u represents the distance between user and satellite node, h represents the distance from orbit to user, γ represents the central angle, and θ represents the elevation angle between satellite node and terminal user [27].

The maximum communication time between satellites and users is represented as [27]

$$T_{\text{com}} = \frac{L_{\text{com}}}{v_{s_2}} \tag{1}$$

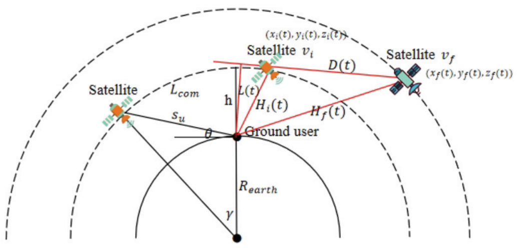


Fig. 2. Geometric relationship of satellite links.

where v_{s_2} is the speed of the low-orbit satellite and L_{com} is the arc length of communication between the low-orbit satellite and the user which is given by

$$L_{\text{com}} = 2 \cdot (R_{\text{earth}} + h) \cdot \gamma. \tag{2}$$

Different satellites can forward tasks and share information through ISLs, allowing satellites to collaborate in processing the computing tasks.

2) For Satellite-Based Links: As the satellite nodes continuously move, the topology of the network undergoes constant changes. Consequently, the distance between satellite nodes can be characterized as a time-dependent function, denoted as $D(t)$ [28]. The total number of available satellites is represented as N, and the satellite nodes are denoted as $V = \{v_i | i \in \{1, 2, \ldots, N\}\}$. The distance between satellite nodes v_i and v_f at time t is expressed as $D(t) = \{d_{if} | i, f \in \{1, 2, \ldots, N\}\}$, where [29]

$$d_{if}(t) = \sqrt{\left[x_i(t) - x_f(t)\right]^2 + \left[y_i(t) - y_f(t)\right]^2 + \left[z_i(t) - z_f(t)\right]^2} \tag{3}$$

where $(x_i(t), y_i(t), z_i(t))$ represents the coordinates of satellite node v_i and $(x_f(t), y_f(t), z_f(t))$ represents the coordinates of satellite node v_f.

Therefore, the altitudes of v_i and v_f are, respectively, represented as

$$H_i(t) = \sqrt{x_i^2(t) + y_i^2(t) + z_i^2(t)} \tag{4}$$

and

$$H_f(t) = \sqrt{x_f^2(t) + y_f^2(t) + z_f^2(t)}. \tag{5}$$

The geometric relationship between satellite nodes v_i and v_f and the ground user is illustrated in Fig. 2.

Using the Heron's formula, the geometric area formed by satellite nodes v_i, v_f, and the ground is given by

$$S(t) = \sqrt{p(t) * (p(t) - H_i(t)) * (p(t) - D(t)) * \left(p(t) - H_f(t)\right)} \tag{6}$$

where $P(t) = ([H_i(t) + D(t) + H_f(t)]/2)$ represents the semi-perimeter of the geometric shape formed by nodes v_i, v_f, and the ground.

The distance between the ISL formed by v_i and v_f and the ground is then given by

$$L(t) = \frac{2S(t)}{D(t)} \tag{7}$$

where $L(t) > H_{\min} + R_{\text{earth}}$ indicates that satellite nodes v_i and v_f can communicate and $H_{\min}$ is minimum altitude of the satellite.

C. Mobility Model

We consider the mobility of vehicles in this section. The position of the ground vehicle user u at time t is expressed as $P_u(t) = (p_u^x(t), p_u^y(t), p_u^z(t))$. From [7], after a time interval t', the position of vehicle user u, $P_u(t') = (p_u^x(t'), p_u^y(t'), p_u^z(t'))$, can be represented as

$$p_u^x\left(t'\right) = p_u^x(t) + D_v \cos\theta_v \tag{8}$$
$$p_u^y\left(t'\right) = p_u^y(t) + D_v \sin\theta_v \tag{9}$$
$$p_u^z\left(t'\right) = p_u^z(t) \tag{10}$$

and

$$D_v = v\left(t' - t\right) \tag{11}$$

where D_v is the distance that the vehicle u moved from the time slot t to t', θ_v is the changing angle of vehicle u, and v is the speed of vehicle u.

At time t, the distance s_u between LEO satellite node v_i and vehicle is represented as

$$s_u = \sqrt{\left(p_u^x(t) - x_i(t)\right)^2 + \left(p_u^y(t) - y_i(t)\right)^2 + \left(p_u^z(t) - z_i(t)\right)^2} \tag{12}$$

where $(x_i(t), y_i(t), z_i(t))$ represents the coordinates of satellite node v_i.

Consider the mobility of satellites, their position coordinate can be generated by the satellite tool kit (STK), and the 3-D coordinate position of each satellite at a certain time also can be obtained through STK.

D. Communication Model

1) For Satellite-to-Terrestrial Links: For satellite-to-terrestrial links between the LEO satellite node v_i and user u, the uplink transmission rate for offloading computation is given by [3]

$$R_{u,v_i}^{\text{up}} = B\log_2\left(1 + \frac{p_u g_{u,v_i}}{\alpha s_u \sigma^2}\right), u \in \Gamma \tag{13}$$

where g_{u,v_i} indicates the power gain from u to v_i by taking into account the large-scale fading and shadowing effects [30]. p_u represents the uplink transmission power of u and B represents the bandwidth. α represents the channel loss. σ^2 represents noise power [3].

2) For Intersatellite Links: The transmission rate is represented as

$$R_{v_i,v_f} = B_{v_i,v_f}\log_2\left(1 + \frac{p_{v_i,v_f} g_{v_i,v_f}}{\sigma^2}\right) \tag{14}$$

where B_{v_i,v_f} represents the bandwidth of ISL, p_{v_i,v_f} indicates the link transmission power, and g_{v_i,v_f} represents the channel power gain.

E. Computation Model

For the tasks generated by the ground users, there are four computation offloading schemes with different corresponding processing times and energy consumption.

1) Local Computing: The task is executed on the edge device u. Let m_u and f_u^{local}, indicate the number of CPU cycles required to execute the tasks and the computing capability of edge devices, respectively. The total delay for the local computation, T_u^{local}, is given by [3]

$$T_u^{\text{local}} = \frac{m_u}{f_u^{\text{local}}} \tag{15}$$

and the energy consumption for the local computation on edge device u, E_u^{local}, is represented as [3]

$$E_u^{\text{local}} = m_u\left(f_u^{\text{local}}\right)^2. \tag{16}$$

2) LEO Satellite (Mist Nodes) Computing: For the satellite-to-terrestrial links, the task is executed on the LEO satellite (mist node). Let f_{u,v_i}^{LEO} represent the computation capability of LEO satellite node v_i. The total delay, T_{u,v_i}^{LEO}, for the ground user u to execute the task on LEO satellite node v_i is given by

$$T_{u,v_i}^{\text{LEO}} = \frac{D(t)}{c} + \frac{m_g}{R_{u,v_i}^{\text{up}}} + \frac{m_u}{f_{u,v_i}^{\text{LEO}}} \tag{17}$$

where $[D(t)/c]$ represents the propagation delay between u and node v_i. $(m_g/R_{i,v_i}^{\text{up}})$ represent uplink transmission delay of user u to node v_i. $(m_u/f_{i,v_i}^{\text{LEO}})$ represent the computation delay for completing the task on node v_i.

The transmission energy consumption is represented as [31]

$$E_{tx}\left(m_g, D(t)\right) = \frac{m_g \times E_{\text{elec}} + \left(m_g \times E_{fs} \times D^2(t)\right)}{s} \tag{18}$$

where E_{fs} and E_{elec} represent the energy consumption of the transmit amplifier and per transmitted bit in the free-space channel model, respectively, and s represents the unit conversion symbol. Therefore, the received data energy consumption is denoted as [31]

$$E_{tx}\left(m_g\right) = \frac{m_g \times E_{\text{elec}}}{s}. \tag{19}$$

The energy consumption of CPU is denoted as

$$E_c = \frac{E_i + (E_m - E_i) \times U \times I}{s} \tag{20}$$

where E_i represents the energy consumption per second of the CPU in an idle state. E_m represents the energy consumption per second of the CPU utilization reaches 100% utilization. U represents the utilization of CPU, which refers to the time proportion during a certain period when the CPU is busy. I represents the update interval of energy [29]. It is the interval at which the system periodically updates the energy consumption information. Multiplying the energy consumption by the time interval can get the correct energy consumption value. $E_m \times U$ can be used to calculate the energy consumption of CPU at the current utilization rate. Therefore, the total energy consumption E_{u,v_i}^{LEO} for ground user u to offload a task of size m_g to LEO satellite v_i is expressed as

$$E_{u,v_i}^{\text{LEO}} = E_{tx}\big(m_g, D(t)\big) + E_{tx}\big(m_g\big) + E_c. \tag{21}$$

For the ISLs, different satellites can collaborate and share information through ISLs to collectively process a large number of computational tasks.

The task from LEO satellite node v_i is forwarded to LEO satellite node v_f for execution, and the total latency includes both transmission and computation delay. The computational capability of LEO satellite node v_i is denoted as f_{v_i,v_f}^{LEO}.

The total delay is given by

$$T_{v_i,v_f}^{\text{LEO}} = \frac{m_g}{R_{v_i,v_f}} + \frac{m_u}{f_{v_i,v_f}^{\text{LEO}}}. \tag{22}$$

The energy consumption for LEO satellite node v_i to offload the task to LEO satellite node v_f includes transmission, reception, and CPU energy consumption.

The total energy consumption is

$$E_{v_i,v_f}^{\text{LEO}} = E_{tx}\big(m_g, D(t)\big) + E_{tx}\big(m_g\big) + E_c. \tag{23}$$

3) MEO Satellite (Edge Data Center) Computing: The task from the LEO satellite node v_i is offloaded to the satellite node v_f at the edge data center. The total latency includes both transmission and computation delay. Let $f_{v_i,v_f}^{\text{Edge}}$ represent the computational capability of the edge data center.

The total delay is represented as

$$T_{v_i,v_f}^{\text{Edge}} = \frac{m_g}{R_{v_i,v_f}} + \frac{m_u}{f_{v_i,v_f}^{\text{Edge}}}. \tag{24}$$

The total energy consumption of LEO satellite node v_i when forwarding the task to satellite node v_f in the edge data center, including the transmission, reception, and CPU energy consumption, can be written as

$$E_{v_i,v_f}^{\text{Edge}} = E_{tx}\big(m_g, D(t)\big) + E_{tx}\big(m_g\big) + E_c. \tag{25}$$

4) GEO Satellite (Cloud Data Center) Computing: The task from the LEO satellite node v_i is offloaded to the satellite node v_f at cloud data center for execution. Let $f_{v_i,v_f}^{\text{Cloud}}$ represent the computational capability of the cloud. The total latency consists of transmission and computation delay, is given by

$$T_{v_i,v_f}^{\text{Cloud}} = \frac{m_g}{R_{v_i,v_f}} + \frac{m_u}{f_{v_i,v_f}^{\text{Cloud}}}. \tag{26}$$

The energy consumption of GEO satellite node v_i when forwarding the task to satellite node v_f in the cloud data center includes transmission, reception, and CPU energy consumption. It can be represented as

$$E_{v_i,v_f}^{\text{Cloud}} = E_{tx}\big(m_g, D(t)\big) + E_{tx}\big(m_g\big) + E_c. \tag{27}$$

F. Problem Formulation

The problem of satellite collaborative computation offloading is formulated with the aim of reducing the system latency and energy consumption. $X = \{X_g, g \in G\}$ represents the computing offloading decisions for all ground users. Then, the objective function can be modeled as the system cost, including the latency and energy consumption, expressed as

$$\begin{aligned}\min_X Q(d, X) = \sum_{g=1}^{G}\sum_{f=1}^{F} \varphi_1\Big(& a_g T_u^{\text{local}} + b_g T_{u,v_i}^{\text{LEO}} \\ &+ c_g T_{v_i,v_f}^{\text{Edge}} + d_g T_{v_i,v_f}^{\text{Cloud}}\Big) + \varphi_2\Big(a_g E_u^{\text{local}} + b_g E_{u,v_i}^{\text{LEO}} \\ &+ c_g E_{v_i,v_f}^{\text{Edge}} + d_g E_{v_i,v_f}^{\text{Cloud}}\Big)\end{aligned} \tag{28}$$

s.t.

$$a_g, b_g, c_g, d_g \in \{0, 1\} \tag{29}$$

$$a_g + b_g + c_g + d_g \le 2 \tag{30}$$

$$a_g T_u^{\text{local}} + b_g T_{u,v_i}^{\text{LEO}} + c_g T_{v_i,v_f}^{\text{Edge}} + d_g T_{v_i,v_f}^{\text{Cloud}} \le T_{\max} \tag{31}$$

$$a_g E_u^{\text{local}} + b_g E_{u,v_i}^{\text{LEO}} + c_g E_{v_i,v_f}^{\text{Edge}} + d_g E_{v_i,v_f}^{\text{Cloud}} \le E_{\max} \tag{32}$$

$$f_{u,v_i}^{\text{LEO}} \le f_{\max}^{\text{LEO}}. \tag{33}$$

Constraint (29) indicates whether the task of vehicle u is executed/offloaded at this node. The ground user's computing tasks can be offloaded in four ways.

1) Let $a_g \in \{0, 1\}$ represent whether the computing task of a user is offloaded locally. $a_g = 1$ represents the task is offloaded, otherwise $a_g = 0$.
2) Let $b_g \in \{0, 1\}$ represent whether the computing task of a user is offloaded at the LEO satellite mist node. Here, $b_g = 1$ means the task is offloaded, otherwise b_g-0.
3) Let $c_g \in \{0, 1\}$ represent whether the computing task of a user is offloaded at the edge data center, where $c_g = 1$ means the task is offloaded, otherwise $c_g = 0$.
4) Let $d_g \in \{0, 1\}$ represent whether the computing task of a user is offloaded at the cloud data center, where $d_g = 1$ means the task is offloaded, otherwise $d_g = 0$.

Constraint (30) indicates that the vehicle user u adopts one offloading decision to execute the computation task. φ_1 and φ_2 represent the weights for delay and energy consumption, respectively. As our optimization objective is related to the timeliness, the weight of the evaluation indicators related to the real-time performance is relatively high. The weight ratio of delay to energy consumption is designed as a variable and the specific value is decided by the user requirement. $T_{\max}$ indicates the maximum allowable delay for the task. $E_{\max}$ indicates the maximum energy consumption. Constraint (31) indicates that the processing time of tasks at the local, LEO satellite fog node, edge data center, or cloud data center must not exceed the maximum allowable delay for the tasks. Constraint (32) is the energy consumption limit. Constraint (33) indicates that the computation offloading requests of the vehicle users cannot exceed the computation capacity of LEO satellites, where $f_{\max}^{\text{LEO}}$ is the maximum computational capability of a satellite.

To solve the problem in (28), the optimal offloading decision for each task needs to be found, while satisfying the given constraints to minimize energy consumption or delay. The offloading decision in the objective function are integer variables, while delay or energy consumption are continuous variables. When the number of users increases, the decision-making for offloading becomes exponentially complex. Therefore, the objective function can be described as a computationally challenging mixed-integer nonlinear programming (MINLP) problem. Traditional optimization algorithms require a lot of iterations to get the optimal result. It has high-computational complexity. To solve this issue, in the next section, the DDLDO algorithm is proposed.

III. Distributed Deep Learning-Based Dynamic Offloading

To address the mixed integer nonlinear programming problem in (28), we employ a deep learning-based dynamic offloading (DDLDO) strategy. This approach utilizes the parallel DNNs to obtain the candidate offloading actions of all computational tasks.

A. DNNs

The DNN model is an extension of perceptron and its architecture is depicted in Fig. 3(a). By learning the relationship between the input and output, it generates intermediate the output results, as represented by [26]

$$z = \sum_{l=1}^{3} w_l x_l + b \tag{34}$$

where w_l represents the weights and b represents the bias. The output of the perceptron is given by

$$y = \Psi(z) \tag{35}$$

where $\Psi(\cdot)$ represents the activation function of neuron, which is used to obtain the desired output. Activation function of perceptron has limited handling capabilities [26]. In neural networks, the alternative activation functions are typically employed to enhance the expressive power of network.

The neural network extends the model of the perceptron. For example, multiple hidden layers enhance the model's expressive power, and the output layer has multiple neuron outputs. By utilizing appropriate activation functions, the expressive power of neural network can be further enhanced. The internal neural network layers of the DNN are shown in Fig. 3(b). The DNN consists of hidden layers, input and output layers. Each neuron in the previous layer is completely connected to every neuron in the subsequent layer in a DNN [32], [33]. The forward propagation algorithm in a DNN uses the output of previous layer to compute the output of next layer. The algorithm uses a series of operations with several weights w_l and biases b to process input values. Its final output is calculated from the results of each layer. Before performing the DNN backpropagation algorithm, a loss function needs to evaluate sample, compute loss between the output and the true training sample output. Through the gradient descent algorithm, the parameters w_l and b of each layer are iteratively solved to minimize the loss function. Train DNN to optimize this loss and obtain the desired model [26].

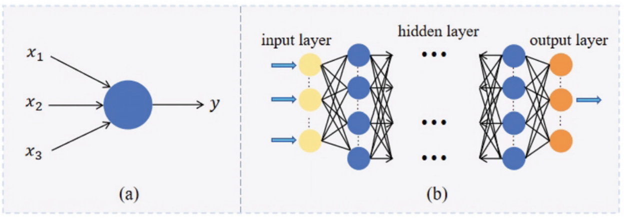


Fig. 3. Perceptron and DNN model. (a) Architecture. (b) Internal neural network layers of the DNN.

B. DDLDO Algorithm

Given the task data sizes for the ground users, represented as d. The labeled tasks obtain K offloading decisions through a neural network. The parameter form of the offloading policy function π is expressed as follows [19], [26], thereby solving the problem (28) to obtain the optimal offloading decision x^*

$$\pi : d \rightarrow x^*. \tag{36}$$

As the number of ground users and tasks in SMEC network increase, the target offloading decision set grows exponentially. Since finding the optimal offloading action is NP-hard, we adopt a parameterized function approximation represented by DNN for π.

Then, we propose a DDLDO algorithm for the SMEC network, as illustrated in Fig. 4, where the DNN k is used to generate the offloading actions and the embedding parameter of DNN k is expressed as θ_k. Each DNN has the corresponding parameters, such as the weight of the connected hidden neurons. For each input d, using K parallel neural networks effectively generates K candidate offloading decisions $\{x_k | k \in K\}$, where $K = \{1, 2, \ldots, K\}$. Subsequently, the offloading decision that minimizes this objective function in (28) is final output, represented as x^*. The resulting data item (d, x^*) is then stored in a memory structure.

Take a batch of samples to train the K DNNs from memory. The parameters of DNNs are updated according to the loss function. Repeat the above until the whole is in a stable state.

C. Generation of Offloading Decision Options

Input data d and use the Kth offloading actor DNN to generate the Kth candidate offloading actions x_k. This process can be expressed as a parameterized function f_{θ_k} as

$$f_{\theta_k} : d \rightarrow x_k \tag{37}$$

where θ_k represents the parameters of the Kth DNN. For each input data d, the K parameters θ_k are initialized randomly. The parameters θ_k for each DNN are different.

After obtaining the K candidate solutions from the outputs of the K parallel DNNs, the variables are substituted into (28) to obtain the optimal decision

$$x^* = \text{argmin} Q^*(d, x_k) \tag{38}$$

where $Q(X)$ represents the optimization objective function in (28). The x^* is the final output of the optimal offloading decision.

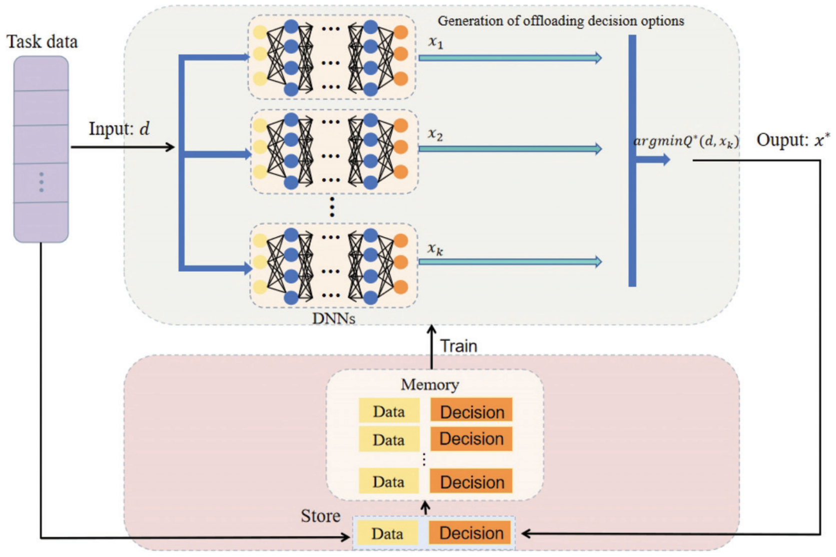


Fig. 4. Structure of DDLDO algorithm.

D. Deep Learning

The x^* calculated from (38) along with its corresponding input data d are saved as new entries of labeled data in a finite-sized memory structure. When the cache capacity reaches its limit, the oldest unused entry is deleted before writing a new entry. Subsequently, all K DNNs are trained using the generated labeled data. All DNNs share the same memory, and each DNN randomly extracts a batch of training samples from the memory. The parameters θ_k of each DNN are optimized using the gradient descent algorithm to minimize cross-entropy loss. It is expressed as

$$L(\theta_k) = -x^T \log f_{\theta_k}(d) - (1-x)^T \log\big(1 - f_{\theta_k}(d)\big). \quad (39)$$

The whole process of proposed DDLDO algorithm is summarized in Algorithm 1. Its computational complexity can be approximated by the training process of DNN, *i.e.,*, $O(U * V * F * G)$, where the DNN contains U fully connected layers, V represents the number of neurons in each layer of DNN, F represents the feature dimension, and G represents the number of tasks.

Algorithm 1 Distributed Deep Learning-Based Dynamic Offloading (DDLDO) Algorithm

```
Input: Input task data $d_t$;
Output: Offloading decision $x_t^*$;
Initialization:
    Initialize the parameters of $K$ DNN networks $\theta_k$ with
    random weights, empty the memory structure;
for $t = 1, 2, , T$ do
    Input task data $d$ to all $K$ DNNs;
    Generate $k$th candidate offloading decision
    $x_k = f_{\theta_{k,t}(d_t)}$ from the $k$th DNN;
    Compute the $Q^*(d_t, x_k)$ according to the $x_k$;
    Choose the Optimal offloading decision $x_t^*$ as the output;
    $x_t^* = \text{argmin}Q^*(d_t, x_t^*)$;
    Store the $(d_t, x_t^*)$ in a memory structure;
    DNN $k$ randomly extracts a batch of samples $(d_t, x_t^*)$
    from the memory structure;
    Train $K$ DNNs with the same structure and update the
    network parameters $\theta_{k,t}$;
end for
```

IV. Simulation Results

A. Simulation Settings

In this section, the proposed task offloading strategy (DDLDO algorithm) is evaluated for its performance in edge computing scenarios using the SatEdgeSim simulator and compared with the following strategies.

1) *WEIGHT_GREEDY:* Greedy algorithm [18].
2) *ROUND_ROBIN:* Deploy the task on the satellite with the lowest workload.
3) *TRADE_OFF:* Multiply the unnormalized and unweighted values of the evaluation indicators, and deploy the task to the satellite corresponding to the lowest value.
4) *TRADI_POLLING:* Traditional polling deployment strategy. Select satellites from a list in sequence for task deployment.
5) *TaskClfQLearning:* Reinforcement learning-based offloading algorithm (Q-learning) [29].

Furthermore, we provide the simulation environment and parameters in detail as follows. The coordinates of satellite at a specific moment can be generated using STK. The defined satellite orbit data and resource configuration parameters of constellation are shown in Table III. To verify the scalability of our proposed algorithm, the number of devices in each simulation is increased by 66, up to 990, with 15 simulation runs. The main simulation parameters of the SatEdgeSim satellite edge computing environment are shown in Table IV

TABLE III
SATELLITE PARAMETERS FOR DIFFERENT RESOURCE TYPES

Resource type	Satellite height	Orbit inclination	Orbit inclination
cloud	35000km	0deg	6/1/0
cloud	36000km	60deg	12/4/2
edge	2150km	56deg	24/6/2
mist	1150km	53deg	160/32/2
mist	1110km	53.8deg	160/32/2
mist	1130km	74deg	40/8/2
mist	1275km	81deg	40/5/2
mist	1325km	70deg	48/6/2
mist	550km	45deg	552/46/2

TABLE IV
SATEDGESIM SIMULATION PARAMETERS

Parameter	Value
Simulation time	10(minutes)
CPU utilization and energy consumption update interval	1(seconds)
Network update interval	1(seconds)
Earth radius	6378137(meters)
Satellite min height	400000(meters)
Network bandwidth	1000(Mbps)
Number of cloud	18
Number of edge	24
Number of mist	1000
Number of devices	1000
devices count times	10
Architectures	ALL
Orchestration algorithms	ROUND ROBIN, TRADE OFF, TRADI POLLING, WEIGHT GREEDY

TABLE V
SATEDGESIM APPLICATIONS PARAMETERS

	Type1	Type2	Type3	Type4	Type5
Generation rate(task/m)	40	40	40	40	40
Maximum delay(s)	5	200	5	200	200
Maximum delay(s)	60000	200000	15000	60000	15000

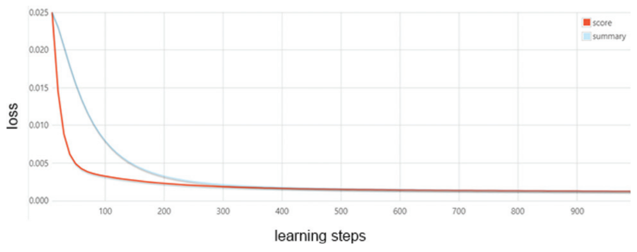


Fig. 5. Learning loss versus learning steps.

and the parameters for different task types are shown in Table V.

In addition, we set the energy consumption per transmission bit for devices during data reception and transmission as $E_{\text{elec}} = 0.5 * 10^{-7}$(J/bit). Under the free space channel model, the energy consumption of the transmit amplifier is $E_{fs} = 0.1 * 10^{-10}$(J/bit/m^2). In the proposed DDLDO algorithm, the DNN layers are fully connected, including two hidden layers. The epochs is 1000, batch size is 256, and learning rate is 0.0001.

B. Simulation Results

The relationship between the learning loss and the training steps of the proposed DDLDO algorithm is shown in Fig. 5.

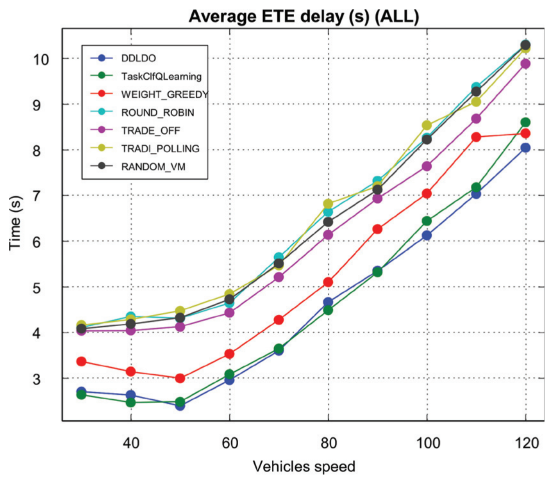


Fig. 6. Average end-to-end delay versus vehicle speed.

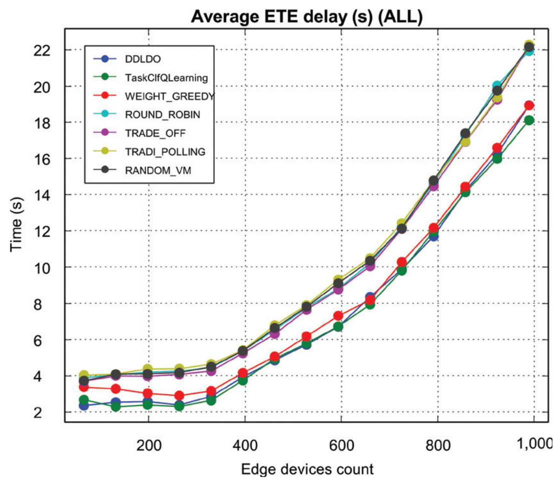


Fig. 7. Average end-to-end delay versus vehicle count.

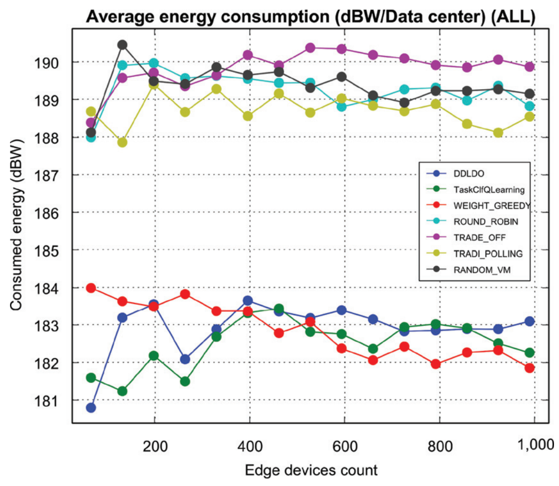


Fig. 8. Average energy consumption of each device versus vehicle count.

The value of K (the number of DNNs) is set to 3 in this experiment. As shown in Fig. 5, the learning loss no longer decreases significantly as the learning steps increase. After the 300th learning step, the learning loss tends to stabilize, indicating that the model has converged. The convergence

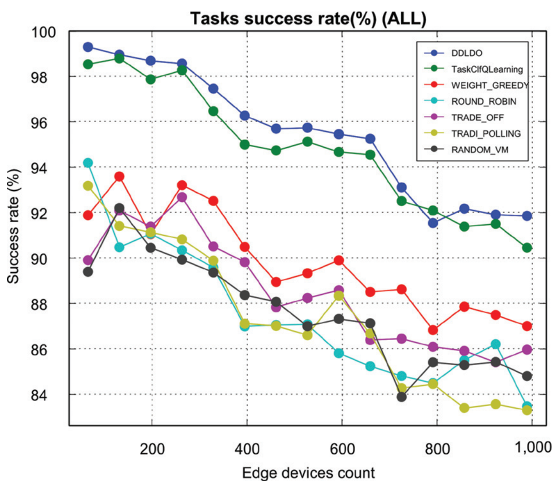


Fig. 9. Task success rate versus vehicle count.

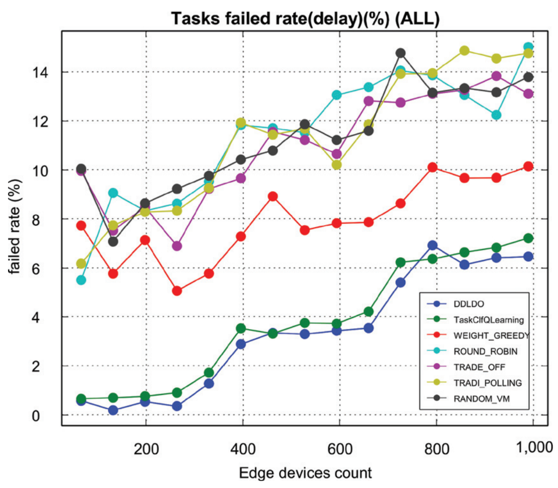


Fig. 10. Tasks failed rate in delay versus vehicle count.

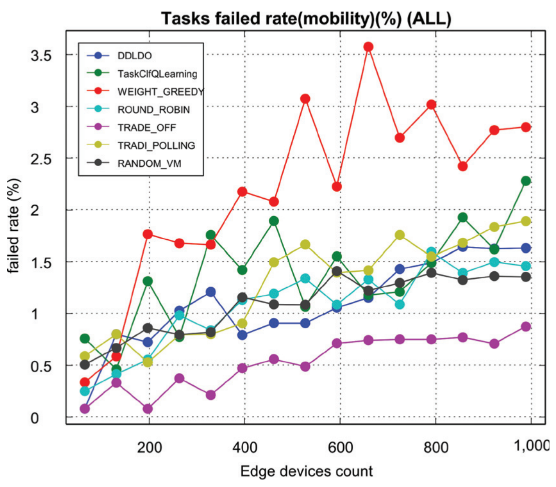


Fig. 11. Tasks failed rate in mobility versus vehicle count.

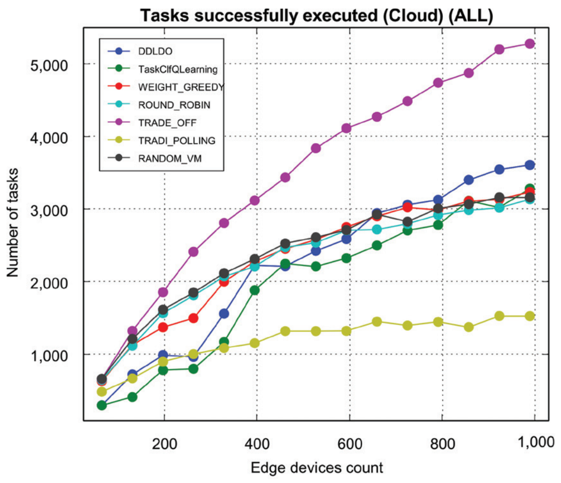
speed of the algorithm will increase with the increase of the number of DNNs. Although each DNN has the same structure, it has different network parameters. Therefore, the difference in output results among different DNNs can accelerate the convergence speed of the algorithm. Using multiple DNNs can accelerate the convergence speed of the algorithm and achieve better performance. As shown in Fig. 5, using a small number of DNNs (greater than or equal to 3) can converge to a local optimum. However, using more DNNs can only slightly improve the convergence speed of the algorithm and requires more computational resources.

Fig. 6 compares the latency under different vehicle user speeds, while keeping the number of vehicle users fixed. Ground users are affected by propagation delay due to transmission distance when communicating with LEO satellites. Consider that different satellites can share information and collaborate to complete computational tasks, the transmission delay of computational tasks between LEO satellites can be ignored when the transmission rate of ISLs is high. But, the transmission delay between satellites and the ground cannot be ignored. As shown in Fig. 6, the latency increases with the increasing velocity of the vehicle users. This is because as the speed increases, the time that vehicle users spend in different satellite coverage areas decreases with the increase in speed, leading to the situations where the communication ranges are exceeded or the communication connections become unstable. This may result in a reduction in selectable satellite nodes and an increase in system delay. Compared with other methods, the proposed DDLDO algorithm has lower delay. Fig. 7 compares the latency under different vehicle user numbers, while keeping the speed of vehicle users fixed. As the number of vehicle users increases, the tasks generated by users also increase, resulting in an exponential growth of the target offloading decision set. The Q-learning is affected by dimensionality issues in more dynamic and complex networks. Since we find that the optimal offloading action is NP-hard, the approach proposed in this article uses a parameterized function approximation of the offloading policy function based on DNN. It is applicable to systems with large-scale state-action spaces and can achieve performance close to optimal. Therefore, the proposed approach has lower delay.

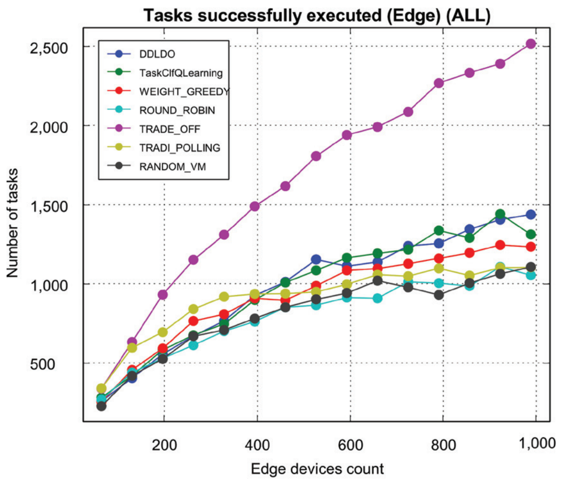
Fig. 8 illustrates the average energy consumption per device and the energy consumption of each device is relatively stable. The differences between the Greedy algorithm, Q-learning, and DDLDO algorithm are small. In general, all devices have low-energy consumption. Our task deployment method aims to reduce the energy consumption while significantly improving other performance metrics.

The task success rates of all algorithms are shown in Fig. 9. Task success rate is a crucial result, as it has the most significant impact on the end users in practical scenarios. Task failures occur due to two reasons: 1) task failures caused by long delays and 2) task failures caused by satellite movement. As the number of tasks increases, effective resource utilization becomes more critical. The proposed algorithm can adapt to high-density, dynamic satellite scenarios with a large number of tasks. The task success rate for our strategy is above 92%, exceeding the task success rates of other algorithms.

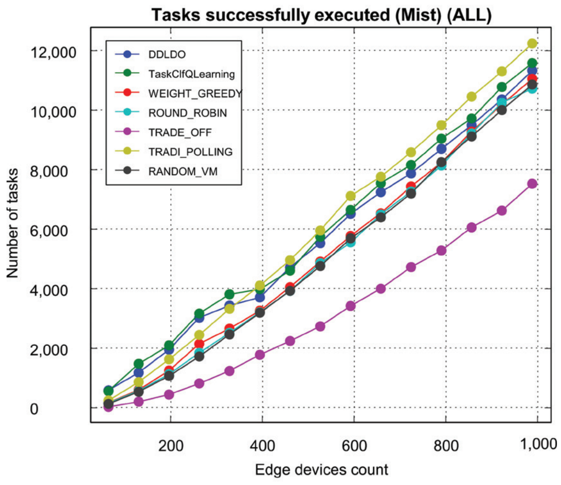
Figs. 10 and 11 demonstrate the task failure rates caused by delay and mobility problems, respectively. Compared with other algorithms, the proposed algorithm has a lower task



(a)



(b)



(c)

Fig. 12. Number of processing tasks for each resource type. (a) Cloud. (b) Edge. (c) Mist.

failure rate. Fig. 12(a)–(c) shows the number of tasks handled by each resource type. As shown in Fig. 12, the proposed strategy tends to offload more tasks to LEO satellite fog nodes, resulting in lower propagation delay. Therefore, as a result, there are fewer latency-related failures, as shown in Fig. 10. The mobility-related failures are caused by LEO satellites moving too fast. Our proposed task deployment strategy considers the dynamic changes in the positions of satellites and vehicle users, as users cannot communicate with low-Earth orbit satellites at all times. We consider the link relationships between satellites and ground users, as well as the link relationships between satellites, to ensure communication and avoid failures caused by satellite movement, as shown in Fig. 11. TRADE_OFF handles the fewest tasks in the mist (LEO satellites), as shown in Fig. 12. However, the TRADE_OFF results in most tasks being processed in the cloud data center, leading to high latency that makes it difficult to meet the requirements of latency-sensitive tasks. Furthermore, the task failure rate increases significantly due to delays, as shown in Fig. 11. Therefore, the overall failure rate of our algorithm is low.

V. Conclusion

In this article, a collaborative computation offloading scheme based on a four-layer architecture for satellite edge computing networks is proposed, leveraging cooperative computation between ground users and a three-layer satellite constellation (low-Earth orbit satellites, edge, and cloud data centers). Considering the mobility of vehicles and satellite nodes, the frequent changes in link states further complicate the design and implementation of such systems, leading to increased cost. In this work, under the constraint of satisfying satellite computational capabilities, this article optimizes the computation offloading decisions to minimize the total cost, while improving the success rate of tasks. Some traditional computation offloading methods often perform poorly in such dynamic and uncertain systems. Therefore, a dynamic edge computation offloading scheme (DDLDO) is proposed in this article, which employs multiple parallel DNNs to learn computation offloading policies. Finally, simulation results demonstrate the advantages of our approach in reducing communication delay, lowering energy consumption, and improving task execution success rates.

References

[1] M. Tropea, "State-of-the-art in satellite communication networks," *Electronics*, vol. 11, no. 9, p. 1368, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/9/1368

[2] T. de Cola and I. Bisio, "QoS optimisation of eMBB services in converged 5G-satellite networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 12098–12110, Oct. 2020.

[3] Q. Tang, Z. Fei, B. Li, and Z. Han, "Computation offloading in LEO satellite networks with hybrid cloud and edge computing," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 9164–9176, Jun. 2021.

[4] O. Kodheli et al., "Satellite communications in the new space era: A survey and future challenges," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 1, pp. 70–109, 1st Quart., 2021.

[5] C.-Q. Dai, Y. Liu, S. Fu, J. Wu, and Q. Chen, "Dynamic handover in satellite-terrestrial integrated networks," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, 2019, pp. 1–6.

[6] C. Chen, L. Liu, T. Qiu, K. Yang, F. Gong, and H. Song, "ASGR: An artificial spider-Web-based geographic routing in heterogeneous vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 5, pp. 1604–1620, May 2019.

[7] L. Yao, X. Xu, M. Bilal, and H. Wang, “Dynamic edge computation offloading for Internet of Vehicles with deep reinforcement learning,” *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 11, pp. 12991–12999, Nov. 2023.
[8] X. Zhu and C. Jiang, “Delay optimization for cooperative multi-tier computing in integrated satellite-terrestrial networks,” *IEEE J. Select. Areas Commun.*, vol. 41, no. 2, pp. 366–380, Feb. 2023.
[9] X. Chen, L. Jiao, W. Li, and X. Fu, “Efficient multi-user computation offloading for mobile-edge cloud computing,” *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.
[10] R. Xie, Q. Tang, Q. Wang, X. Liu, F. R. Yu, and T. Huang, “Satellite-terrestrial integrated edge computing networks: Architecture, challenges, and open issues,” *IEEE Netw.*, vol. 34, no. 3, pp. 224–231, May/Jun. 2020.
[11] A. Alsharoa and M.-S. Alouini, “Improvement of the global connectivity using integrated satellite-airborne-terrestrial networks with resource optimization,” *IEEE Trans. Wireless Commun.*, vol. 19, no. 8, pp. 5088–5100, Aug. 2020.
[12] Y. Qiu et al., “Mobile edge computing in space-air-ground integrated networks: Architectures, key technologies and challenges,” *J. Sens. Actuator Netw.*, vol. 11, no. 4, p. 57, 2022. [Online]. Available: https://www.mdpi.com/2224-2708/11/4/57
[13] H. Li, C. Chen, C. Li, L. Liu, and G. Gui, “Aerial computing offloading by distributed deep learning in collaborative satellite-terrestrial networks,” in *Proc. 13th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, 2021, pp. 1–6.
[14] H. Zhang, R. Liu, A. Kaushik, and X. Gao, “Satellite edge computing with collaborative computation offloading: An intelligent deep deterministic policy gradient approach,” *IEEE Internet Things J.*, vol. 10, no. 10, pp. 9092–9107, May 2023.
[15] Z. Qin, H. Yao, T. Mai, D. Wu, N. Zhang, and S. Guo, “Multi-agent reinforcement learning aided computation offloading in aerial computing for the Internet-of-Things,” *IEEE Trans. Services Comput.*, vol. 16, no. 3, pp. 1976–1986, May/Jun. 2023.
[16] D. Han et al., “Two-timescale learning-based task offloading for remote IoT in integrated satellite–terrestrial networks,” *IEEE Internet Things J.*, vol. 10, no. 12, pp. 10131–10145, Jun. 2023.
[17] Z. Song, Y. Hao, Y. Liu, and X. Sun, “Energy-efficient multiaccess edge computing for terrestrial-satellite Internet of Things,” *IEEE Internet Things J.*, vol. 8, no. 18, pp. 14202–14218, Sep. 2021.
[18] J. Wei, S. Cao, S. Pan, J. Han, L. Yan, and L. Zhang, “SatEdgeSim: A toolkit for modeling and simulation of performance evaluation in satellite edge computing environments,” in *Proc. 12th Int. Conf. Commun. Softw. Netw. (ICCSN)*, 2020, pp. 307–313.
[19] L. Huang, X. Feng, A. Feng, Y. Huang, and L. P. Qian, “Distributed deep learning-based offloading for mobile edge computing networks,” *Mobile Netw. Appl.*, vol. 27, pp. 1123–1130, Jun. 2022.
[20] M. Khayyat, I. A. Elgendy, A. Muthanna, A. S. Alshahrani, S. Alharbi, and A. Koucheryavy, “Advanced deep learning-based computational offloading for multilevel vehicular edge-cloud computing networks,” *IEEE Access*, vol. 8, pp. 137052–137062, 2020.
[21] J. Guo, H. Zhang, L. Yang, H. Ji, and X. Li, “Decentralized computation offloading in mobile edge computing empowered small-cell networks,” in *Proc. IEEE Globecom Workshops (GC Wkshps)*, 2017, pp. 1–6.
[22] S. Zhang, G. Cui, Y. Long, and W. Wang, “Joint computing and communication resource allocation for satellite communication networks with edge computing,” *China Commun.*, vol. 18, no. 7, pp. 236–252, Jul. 2021.
[23] Y. Hu, W. Gong, and F. Zhou, “A Lyapunov-Optimized dynamic task offloading strategy for satellite edge computing,” *Appl. Sci.*, vol. 13, no. 7, p. 4281, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/7/4281
[24] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, and W. Zhuang, “Learning-based computation offloading for IoT devices with energy harvesting,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1930–1941, Feb. 2019.
[25] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, “Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning,” *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4005–4018, Jun. 2019.
[26] Q. Tang, Z. Fei, and B. Li, “Distributed deep learning for cooperative computation offloading in low Earth orbit satellite networks,” *China Commun.*, vol. 19, no. 4, pp. 230–243, Apr. 2022.
[27] B. R. Elbert, *Introduction to Satellite Communication*. Boston, MA, USA: Artech House, 1987. [Online]. Available: https://api.semanticscholar.org/CorpusID:108475259
[28] Q. Zhu, H. Tao, Y. Cao, and X. Li, “Laser inter-satellite link visibility and topology optimization for mega constellation,” *Electronics*, vol. 11, no. 14, p. 2232, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/14/2232
[29] J. Shuai, B. Xie, H. Cui, J. Wang, and W. Wen, “*Q*-learning-based task offloading strategy for satellite edge computing,” *Int. J. Commun. Syst.*, vol. 37, no. 5, 2024, Art. no. e5691. [Online]. Available: https://api.semanticscholar.org/CorpusID:266288390
[30] A. Abdi, W. C. Lau, M.-S. Alouini, and M. Kaveh, “A new simple model for land mobile satellite channels: First- and second-order statistics,” *IEEE Trans. Wireless Commun.*, vol. 2, no. 3, pp. 519–528, May 2003.
[31] M. S. Dawood, S. S. Benazer, S. V. Saravanan, and V. Karthik, “Energy efficient distance based clustering protocol for heterogeneous wireless sensor networks,” *Mat. Today Proc.*, vol. 45, pp. 2599–2602, Jan. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214785320389525
[32] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, “Learning to optimize: Training deep neural networks for wireless resource management,” in *Proc. IEEE 18th Int. Workshop Signal Proc. Adv. Wireless Commun. (SPAWC)*, 2017, pp. 1–6.
[33] H. Ye, G. Y. Li, and B.-H. Juang, “Power of deep learning for channel estimation and signal detection in OFDM systems,” *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, Feb. 2018.

Jiaqi Shuai is currently pursuing the M.S. degree with the School of Electronic and Information Engineering, South China Normal University, Guangzhou, China.

Her current research interests are in the areas of task offloading and satellite edge computing.

Haixia Cui (Senior Member, IEEE) received the M.S. and Ph.D. degrees in communication engineering from South China University of Technology, Guangzhou, China, in 2005 and 2011, respectively.

She is currently a Full Professor with the School of Electronics and Information Engineering, South China Normal University. From July 2014 to July 2015, she was an Advanced Visiting Scholar (Visiting Associate Professor) with the Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC, Canada. She has authored or coauthored more than 70 refereed journal and conference papers and one books. Her also holds about 20 patents. Her current research interests are in the areas of mobile edge computing, vehicular networks, cooperative communication, wireless resource allocation, 5G/6G, multiple access control, and power control in wireless networks.

Yejun He (Senior Member, IEEE) received the Ph.D. degree in information and communication engineering from Huazhong University of Science and Technology, Wuhan, China, in 2005.

From 2005 to 2006, he was a Research Associate with the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong. From 2006 to 2007, he was a Research Associate with the Department of Electronic Engineering, Faculty of Engineering, The Chinese University of Hong Kong, Hong Kong. In 2012, he was a Visiting Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. From 2013 to 2015, he was an Advanced Visiting Scholar (Visiting Professor) with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA. From 2023 to 2024, he is an Advanced Research Scholar (Visiting Professor) with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore. Since 2006, he has been a Faculty of Shenzhen University, Shenzhen, China, where he is currently a Full Professor with the College of Electronics and Information Engineering; the Director of Sino-British Antennas and Propagation Joint Laboratory, Ministry of Science and Technology, People's Republic of China; the Director of Guangdong Engineering Research Center of Base Station Antennas and Propagation; and the Director of Shenzhen Key Laboratory of Antennas and Propagation. He was selected as a Leading Talent in "Guangdong Special Support Program" and Shenzhen "Pengcheng Scholar" Distinguished Professor, China, in 2024 and 2020, respectively. He has authored or coauthored more than 300 refereed journal and conference papers and seven books. He holds about 20 patents. His research interests include wireless communications, antennas, and radio frequency.

Dr. He was also a recipient of Shenzhen Overseas High-Caliber Personnel Level B (Peacock Plan Award B) and Shenzhen High-Level Professional Talent (Local Leading Talent). He received the Second Prize of Shenzhen Science and Technology Progress Award in 2017, the Three Prize of Guangdong Provincial Science and Technology Progress Award in 2018, the Second Prize of Guangdong Provincial Science and Technology Progress Award in 2023, and the 10th Guangdong Provincial Patent Excellence Award in 2023. He is currently the Chair of IEEE Antennas and Propagation Society-Shenzhen Chapter and obtained the 2022 IEEE APS Outstanding Chapter Award. He has served as a Technical Program Committee Member or a Session Chair for various conferences, including the IEEE Global Telecommunications Conference, the IEEE International Conference on Communications, the IEEE Wireless Communication Networking Conference, and the IEEE Vehicular Technology Conference. He served as the TPC Chair for IEEE ComComAp 2021 and the General Chair for IEEE ComComAp 2019. He was selected as a Board Member of the IEEE Wireless and Optical Communications Conference (WOCC). He served as the TPC Co-Chair for WOCC 2023/2022/2019/2015, APCAP 2023, UCMMT 2023, ACES-China2023, NEMO 2020, and so on. He acted as the Publicity Chair of several international conferences, such as the IEEE PIMRC 2012. He is serving as an Executive Chair for 2024 IEEE International Workshop of Radio Frequency and Antenna Technologies. He is the Principal Investigator for over 40 current or finished research projects, including the National Natural Science Foundation of China, the Science and Technology Program of Guangdong Province, and the Science and Technology Program of Shenzhen City. He has served as a reviewer for various journals, such as the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, the IEEE TRANSACTIONS ON COMMUNICATIONS, the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, the IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION, the IEEE WIRELESS COMMUNICATIONS, the IEEE COMMUNICATIONS LETTERS, the *International Journal of Communication Systems*, and *Wireless Personal Communications*. He is serving as an Associate Editor for IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION, IEEE TRANSACTIONS ON MOBILE COMPUTING, *IEEE Antennas and Propagation Magazine*, IEEE ANTENNAS AND WIRELESS PROPAGATION LETTERS, *International Journal of Communication Systems, China Communications,* and *ZTE Communications*. He is a Fellow of IET, and a Senior Member of the China Institute of Communications and the China Institute of Electronics.

Mohsen Guizani (Fellow, IEEE) received the B.S. (with Distinction) and M.S. and Ph.D. degrees in electrical and computer engineering from Syracuse University, Syracuse, NY, USA, in 1985, 1987, and 1990, respectively.

He is currently a Professor of Machine Learning with the Mohamed Bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE. Previously, he worked in different institutions in USA. He is the author of 11 books, more than 1000 publications and several U.S. patents. His research interests include applied machine learning and artificial intelligence, Internet of Things, intelligent autonomous systems, smart city, and cybersecurity.

Dr. Guizani has won several research awards, including the "2015 IEEE Communications Society Best Survey Paper Award," the Best ComSoc Journal Paper Award in 2021, as well five Best Paper Awards from ICC and Globecom Conferences. He is also the recipient of the 2017 IEEE Communications Society Wireless Technical Committee Recognition Award, the 2018 AdHoc Technical Committee Recognition Award, and the 2019 IEEE Communications and Information Security Technical Recognition Award. He was listed as a Clarivate Analytics Highly Cited Researcher in Computer Science from 2019 to 2022. He served as the Editor-in-Chief for IEEE NETWORK and is currently serving on the editorial boards for many IEEE transactions and magazines. He was the Chair of the IEEE Communications Society Wireless Technical Committee and the Chair of the TAOS Technical Committee. He served as the IEEE Computer Society Distinguished Speaker and is currently the IEEE ComSoc Distinguished Lecturer.