Hindawi
Security and Communication Networks
Volume 2022, Article ID 2332769, 14 pages
https://doi.org/10.1155/2022/2332769

Research Article

Telematics Collaborative Resource Allocation Algorithm Based on Cloud Sidecar

Zheng Zhang,[1] **Yanling Shao**,[1] **Xing Liu**,[2] **and Yibo Han**[3]

[1]*School of Computer and Software, Nanyang Institute of Technology, Nanyang, Henan 473000, China*
[2]*Frontier Information Technology Research Institute, Zhongyuan University of Technology, Zhengzhou, Henan 450000, China*
[3]*Nanyang Institute of Big Data Research, Nanyang Institute of Technology, Nanyang, Henan 473000, China*

Correspondence should be addressed to Zheng Zhang; zhangzheng@nyist.edu.cn

Received 13 July 2022; Revised 4 August 2022; Accepted 25 August 2022; Published 8 September 2022

Academic Editor: Ke Gu

This paper provides an in-depth study and analysis of a distributed allocation algorithm for collaborative resources for cloud-edge-vehicle-based Telematics. The approach starts from the emerging application of urban environmental monitoring based on vehicular networking, with an integrated design of data sensing detection and transmission, and collaborative monitoring of vehicle swarm intelligence based on urban air quality collection to avoid redundancy of information and communication overload. A hybrid routing method with minimal delay for reliable data transmission is proposed. The power adjustment algorithm divides the channel into 3 states. When the CBR is less than 0.5, the channel is in an idle state, and when the CBR is greater than 0.5 and less than 0.8, the channel is in an active state. The algorithm designs redundancy strategies based on coding mechanisms to improve the reliability of data transmission, combines coding mechanisms with routing design, incorporates routing switching ideas, and performs probability-based routing decisions to minimize the delay. In straight-line road sections, a fuzzy logic prediction-based vehicle adaptive connectivity clustering routing algorithm is proposed to reduce the communication overhead during vehicle collaboration and ensure high network connectivity; at intersections, a probability-based minimum delay routing decision algorithm is proposed to reduce the information transmission delay. Experiments show that the proposed method effectively improves the efficiency of data-aware collection and transmission, and increases the reliability of transmission. With the explosive growth of video services, the problem of intelligent transmission of DASH-based video streams has become another research hotspot in mobile edge networks. Based on the edge container cloud architecture of vehicular networking, the resource constraints of microservices when deployed in the edge cloud platform were analyzed, and a multi-objective optimization model for microservice resource scheduling was established with the comprehensive performance objectives of shortest microservice invocation distance, highest resource utilization of physical machine clusters, and ensuring load balancing as much as possible.

1. Introduction

Before the emergence of container technology, microservices were mainly deployed on virtual machines directly on bare metal, which was difficult to operate and maintain. Container technology, on the other hand, is a lightweight virtualization technology that provides resource scheduling and isolation for microservices at the container engine layer, reducing concerns about inconsistencies across platforms between development, testing, and production environments [1]. When deploying specific functional modules in different locations, how to utilize limited resources in a more balanced and efficient manner and further improve application service quality and user experience is a challenge in the initialization process of mobile IoT slices. And because containers reduce the hardware system virtualization layer, they can use the hardware resources of the actual physical host directly and therefore make fuller use of hardware resources. In summary, lightweight, and fast start/stop containers are well suited for edge workloads and are a good vehicle for Telematics microservices. However, as the granularity of containers is smaller, the number of

containers that can be started by one physical machine is also larger, and the resource management for containers is more complex [2]. Common container scheduling tools such as Kubernetes and Docker Swarm Kit only provide some simple resource scheduling policies, which cannot fully utilize the performance of physical machines, so it is crucial to design a reasonable resource scheduling policy for microservice containers [3]. Vehicle networking refers to a system network that carries out wireless communication and information exchange among vehicles, roads, pedestrians, and the Internet on the basis of in-vehicle network, inter-vehicle network, and in-vehicle mobile Internet according to agreed communication protocols and data exchange standards. Traditional vehicle networking refers to the electronic label loaded on the vehicle through the wireless radio frequency and other recognition technology, effectively realized in the information network platform to the vehicle attributes and static, dynamic information extraction, and utilization, according to the different function demands to the vehicle operation status to provide effective supervision and comprehensive service system. With the rapid development of the Telematics industry and technology, the traditional definition can no longer cover all its contents.

The rapid development and widespread use of the Internet of Everything have led to a shift in the role of edge devices, from the role of a single consumer of data to the role of a consumer and producer of data, with edge devices becoming more intelligent and capable of autonomous deep learning, predictive analysis, and intelligent data processing of data at the edge of the network. Big data processing is slowly entering the era of edge computing with the Internet of Everything at its core from the cloud computing era [4]. Cloud computing relies on powerful resource provisioning in data centers to centrally process big data, compared to edge computing which relies on numerous edge devices at the edge of the network to process massive amounts of data, reduce the occupation of network resources, enhance real-time communication capabilities, and complete data processing and execution services with extremely low latency [5]. With the growth of the Internet of Everything, latency-sensitive and compute-intensive application services are increasing, and cloud computing solutions cannot meet the latency requirements of these application services. For example, autonomous driving, self-driving cars generate 4 TB of data per day, which is demanding in terms of computational latency. The accuracy of multiedge collaborative mobile IoT slicing can reach 85%, while the other two comparison models are 72% and 65%, respectively. WAN transmission brings latency uncertainty, and autonomous driving data needs to be processed at the edge of the network to ensure low latency, but the computational resources on the edge side of the network are far inferior to cloud computing, and the data are processed through deep neural networks under resource-constrained conditions [6].

With the continuous improvement of relevant standards and the increasing number of smart vehicles, it is foreseeable that more vehicles will be connected to the network through relevant protocols in the future. Along with the increasing number of vehicles, road hazards have become an issue that is faced in the development of Telematics. This makes it increasingly important to study the transmission strategy of vehicle safety services. In the process of vehicle communication based on IEEE 802.11P and LTE-V protocols, channel congestion, channel interference, shadow fading, and intelligent computational processing are the main factors affecting the performance of vehicle communication. It is important to study how to schedule the computational and communication resources in vehicular networking to improve the communication performance of vehicle safety services.

To sum up, the continuous development of the Internet of Vehicles business and the continuous increase in the number of connected vehicles have brought great challenges to the existing Internet of Vehicles solutions. In order to improve driving safety and travel efficiency, the problem of limited computing power of a single vehicle can be solved by offloading tasks to the MEC server for execution. On the one hand, different IoV services have different requirements for latency, bandwidth, and computing power. How to manage and allocate communication and computing resources to meet the needs of various services is a key issue in-vehicle edge computing networks. On the other hand, due to the distributed deployment of MEC servers, the communication and computing resources on edge nodes are relatively limited. However, the unloading requests of vehicles are usually random and sudden, and an unreasonable resource allocation scheme will cause problems such as increased delay, unstable network services, and poor service quality. Therefore, it is of great significance to study communication and computing resource allocation methods for task offloading in-vehicle edge computing networks.

This paper proposes a distributed end-edge collaboration algorithm for the edge network of intelligent networked vehicles. According to the characteristics of high reliability and low delay content transmission of the Internet of Vehicles, a limited block length mechanism is introduced. At the same time, the compression coding power consumption of the vehicle video information source is introduced, and the vehicle energy consumption model is established. According to the video quality requirements of the vehicle video information source, by adjusting the video coding rate, the information source transmission rate, and the selection of vehicle multipath routing, a fully distributed optimization algorithm is proposed to improve the utilization of network resources and ensure a single Equity in energy consumption of vehicles. This paper proposes a distributed edge-end collaborative algorithm based on the subgradient algorithm, which realizes the resource allocation strategy by adjusting the video coding rate, the information source transmission rate, and the vehicle multipath routing decision. The farther the terminal is from the communication node, the greater the data transmission delay. Therefore, this paper uses dynamic communication nodes to solve the delay and energy consumption problems faced by mobile terminals. The algorithm can be deployed and executed in each ICV and only needs to exchange a small amount of information with its neighbouring nodes.

2. Related Works

As transferring large amounts of data to the cloud not only takes up limited backhaul bandwidth resources, it also generates large transmission latency and poses security risks of data leakage. In response to these problems, edge computing was born [7]. Edge computing is a service that deploys data processing and storage capabilities from the cloud as close to the endpoint as possible, storing, and analyzing data at the edge of the network, solving many of the challenges that exist when transferring data to the cloud center. As smart chips continue to develop, the processing power of terminals is gradually increasing, so that some simple data processing can be done locally at the terminal [8]. Of course, edge intelligence and terminal intelligence also have problems that need to be solved, such as the uneven distribution of edge devices and the uneven data storage and processing capabilities; not only is the energy consumption of terminal devices relatively high during processing, the terminal itself has limited endurance, and the life span of the terminal is also a problem that cannot be ignored when processing large amounts of data [9].

A heuristic algorithm based on three scenarios is proposed for the task scheduling problem of edge servers in multiserver multiuser mobile edge computing systems. Experimental results show that the algorithm can significantly reduce the average task execution delay. An efficient lightweight offloading scheme is proposed for the multi-user edge system [10]. The results show that this offloading scheme can effectively reduce the execution time of end-to-end tasks and improve the resource utilization of the edge server [11]. The battery size of end devices is typically very limited due to device size, etc. [12].

Abreha et al. proposed an analytical framework that models downlink traffic in a drive-through vehicle networking scenario via a multidimensional Markov process. It can be speculated that the computing load brought by the number of tasks at this time is not high for the edge servers in the network. So, the effect is not obvious. As the number of tasks increases, the task completion rate varies greatly. When the number of tasks is 60, the lowest task completion rate is 79.1% and the highest is 94.8%. There is a 15.7% gap between the lowest and highest. The arrival of packets in the RSU buffer is constructed as a Poisson process, and the transit time is exponentially distributed [13]. Considering the state space explosion problem associated with multidimensional Markov processes, this paper uses an iterative per-duration technique to compute the stationary distribution of Markov chains [14]. Sar-dianos et al. studied the hybrid data dissemination problem, i.e., optimally determining the time and destination of data transmission vehicles, and whether the vehicles obtain the required data directly from the edge of nearby vehicles [15]. The authors proposed a new data propagation algorithm, called the hybrid data propagation offline algorithm, which prioritizes finding the most beneficial vehicle-to-vehicle broadcast, and then selected the feasible vehicle-to-base station propagation method [16]. Shakir et al. studied how to deploy drop box optimally by considering the trade-off between delivery delay and drop box deployment cost [17]. To address this issue, first, provide a theoretical framework to accurately estimate delivery delay; then, based on the dimension based on the idea of enlargement and dynamic programming [18]. In terms of content uploading, Guan et al. proposed to deploy dedicated access points (APs) at bus stops to facilitate video uploading to study the video uploading problem of mobile buses and proposed a water injection placement algorithm that aims to balance the distribution [19]. The aggregate bandwidth of each bus is analyzed by establishing a queuing model to analyze the upload delay of video content, and a machine learning model is further used to incorporate the impact of bus routes into the queuing model. Based on the different application conditions, it is a great challenge to meet the requirements of low delay, huge amount of calculation, high efficiency, high reliability, and meticulous precision. For example, each car has different requirements for communication; there are self-driving cars and ordinary vehicles, which need to be treated differently.

In an edge computing environment, there are two aspects of energy consumption by the end device when performing task offloading [20]. One is the computational energy consumed when the task is computed locally, and the other is the transmission energy consumed when the task is uploaded to the edge server and the results are received back from the edge server. Therefore, offloading strategies can be designed to reduce energy consumption by means such as adjusting CPU frequency and offloading intensive tasks to the server [21].

The joint proposes a layered, modular edge computing architecture that runs on the cloud, fog, and edge devices and provides containerized services and microservices. The proposed architecture has three main layers: the sensing layer, the intermediary layer, and the enterprise layer. The perception layer is the underlying layer that performs sensing and operations (edge computing); the intermediary layer represents intermediate devices and operations (gateways, fog computing); and the upper layer, called the enterprise layer, represents the cloud and operations such as long-term global storage. The proposed architecture ensures that the data are collected and analyzed in the most efficient and logical place between the source and the cloud, balancing the load and pushing the computation and intelligence to the appropriate layer. It is also necessary to allocate the microservice containers with call dependencies to the same physical host, so that the cross-server calls of the microservice container are as few as possible, to reduce the response time of the service. For the container scheduling problem under the microservice architecture, a container scheduling strategy based on an improved particle swarm algorithm that effectively reduces network calls to fast physical hosts in container-based microservice clusters is proposed, considering the invocation relationship conditions between microservices.

3. Analysis of the Distributed Allocation Algorithm for Collaborative Resource Allocation in the Cloud-Edge-Vehicle Side of Telematics

3.1. Collaborative Resource Design for Cloud-Edge-Vehicle Telematics. The user terminal in Telematics is a vehicle, and

because vehicles travel fast on the road with many vehicles and complex road conditions, smart driving vehicles under Telematics have high quality of service (QoS) requirements for various applications [22]. For this reason, when performing microservice deployment under Telematics, various aspects are considered including the dynamic characteristics of the vehicle (e.g., whether it is moving or not), the type of big data problems (e.g., speed, accuracy), and commutationally complex data analysis. The deployment architecture of microservices in the Telematics system is derived from the previous section on Telematics system architecture: cloud-side-end microservice layered deployment architecture, as shown in Figure 1.

This architecture consists of three parts: the vehicle and road test terminal, the edge cloud platform, and the central cloud. Unlike traditional cloud computing centers, the edge cloud layer deploys the cloud infrastructure near the service road section, which is not as powerful as traditional large cloud data centers but is closer to the specific service area and can effectively improve the quality of service (QoS) of Telematics applications. Vehicle and road information sensing through sensing technology, the On-Board Unit (OBU) enables vehicle-to-vehicle (V2V), vehicle-to-road, and vehicle-to-cloud communications. Microservices with functions such as onboard data fusion calculations, location positioning, road condition sensing, periodic or event data sending and receiving, and supporting autonomous driving fusion decisions are therefore deployed to the onboard and roadside terminals. The number of users accessing the Internet of Vehicles service will suddenly increase, resulting in the overload of some specific microservice resources. When the QoS of the application is reduced, the specific microservice container instance will be dynamically added.

The edge cloud platform of Telematics is a data processing center, through its strong computing capacity, it can realize the processing of massive real-time data of Telematics; it is an application software deployment platform, providing traffic-based cloud services to multiple users, to realize the interoperability of resources of different traffic systems; it is a resource management platform, through virtualization technology, realizing the unified management and elastic expansion of computing resources, thus increasing system stability, reducing costs, and saving energy consumption.

At the vehicle end, the vehicle unit and sensors are mainly used to collect the vehicle and environmental information; at the tube end, the wireless communication network is mainly responsible for the return transmission of information collected by the vehicle unit and roadside units, and the distribution of control information; at the edge of the vehicle network, the cloud end, using its strong computing and data processing capabilities, will process the collected data and information and calculate the integrated output for the application services required by the user. Accordingly, it is important to consider having enough CPU processing power available, memory, disk space available, and bandwidth resources to meet the hardware requirements when deploying microservices [16]. In the Telematics Edge Cloud, container-based microservice resource scheduling means that according to the resource requirements of the microservice, the cloud computing center allocates the corresponding container resources to it, and then the scheduling system deploys it to the physical machine to run. The operation logic of microservice scheduling is divided into four steps: (1) Grab the task in the task executor by annotation and report it to the task registration center. (2) The task orchestration center obtains data from the task registration center to schedule and save it into persistent storage. (3) The task scheduling center obtains scheduling information from persistent storage. (4) The task scheduling center accesses the task executor according to the scheduling logic. It has high precision and robustness, and is more suitable for solving problems such as missing data. The essence is that the scheduling system schedules the set of containers for deploying microservices to run on the set of physical machines according to the scheduling policy. The containers can be configured together with the resources required by the microservice programs, making full use of the hardware resources of the Telematics Edge Cloud platform, and making the physical machine clusters in the Edge Cloud as load balanced as possible while meeting the resource requirements of the microservices.

In addition, microservices are generally responsible for a single service function. When providing services to Telematics users, multiple microservices are often required to work together to meet user requirements, so there is a dependency relationship between microservices to invoke and be invoked. Each microservice is deployed into a container, and there is a one-to-one correspondence between the microservice and the container, thus creating a call dependency between each container. Therefore, in the scheduling and resource dispatching of containers, it is not only necessary to consider maximizing server resource utilization and load balancing between servers but also to allocate microservice containers with invocation dependencies to the same physical host as far as possible, so that the cross-server invocations of microservice containers are as few as possible to reduce the response time of the service, as shown in Figure 2.

The initial deployment of microservices means that the application estimates the number of resources it needs based on the actual number of users it serves daily while ensuring that there is a certain number of resources left over; then it applies for resources to the edge cloud based on its estimated resource characteristics; finally, the Telematics Edge Cloud platform schedules and deploys it to a designated physical host to run according to the microservice resource scheduling policy.

The initial deployment of microservices means that the application estimates the number of resources it needs based on the actual number of users it serves daily while ensuring that there is a certain number of resources left over; then it applies for resources to the edge cloud based on its estimated resource characteristics; finally, the Telematics Edge Cloud platform schedules and deploys it to a designated physical host to run according to the microservice resource scheduling policy. The transmission time of each batch is

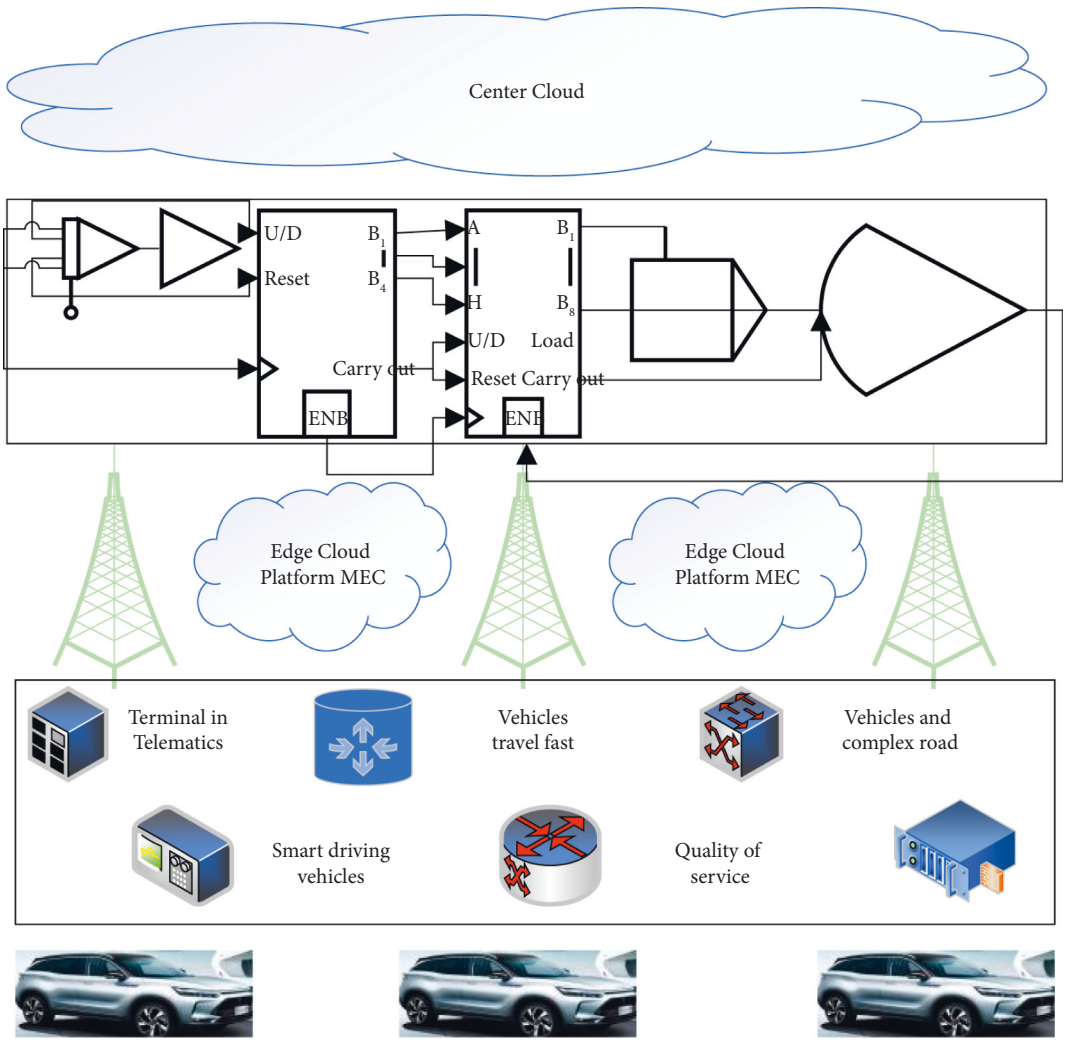

Figure 1: The layered deployment architecture of Telematics microservices.

minimized and the cost function is minimized. At the same time, the above algorithm also considers the upper limit of the bandwidth of each task, which can maximize the use of bandwidth resources. The initial deployment of microservices can ensure the number of resources required for their daily operation and meet their QoS requirements. The dynamic scaling of microservices refers to the dynamic addition of specific microservice container instances when unexpected conditions are encountered, such as a sudden increase in the number of vehicles in the service area of the Telematics Edge Cloud Platform, which results in the overloading of some specific microservice resources and a reduction in the QoS of the application, and then the Edge Cloud Platform deploys the added microservice instances to the specified physical hosts to run following the microservice resource scheduling policy to meet their resource requirements.

Whether it is the initial deployment of microservices or dynamic expansion, the edge cloud microservice resource scheduling policy is required to reasonably schedule microservice containers to deploy and run on the specified physical hosts. In the following, the microservice resource scheduling problem on the Telematics edge cloud platform is modeled according to its characteristics [23]. In the microservice resource scheduling problem on the edge cloud, it is crucial to make the most efficient use of resources in the resource scheduling process due to the limited computing resources compared to traditional cloud computing centers and the high user requirements for latency. In addition, as there are dependencies between microservices, the dependency between containers is also an important factor in the resource scheduling process, thus ensuring the responsiveness of the edge cloud to tasks, improving the utilization of cloud resources, and reducing the energy consumption of the edge cloud center.

3.2. Distributed Allocation Algorithm Design. In the system proposed in this paper, each buyer has a computationally intensive service to perform, but due to its computing resources and capacity constraints needs to migrate part of the service to a suitable service vehicle in the vicinity, and pay the final chosen service vehicle a certain amount of money. Each buyer is represented by a 7-tuple as follows:

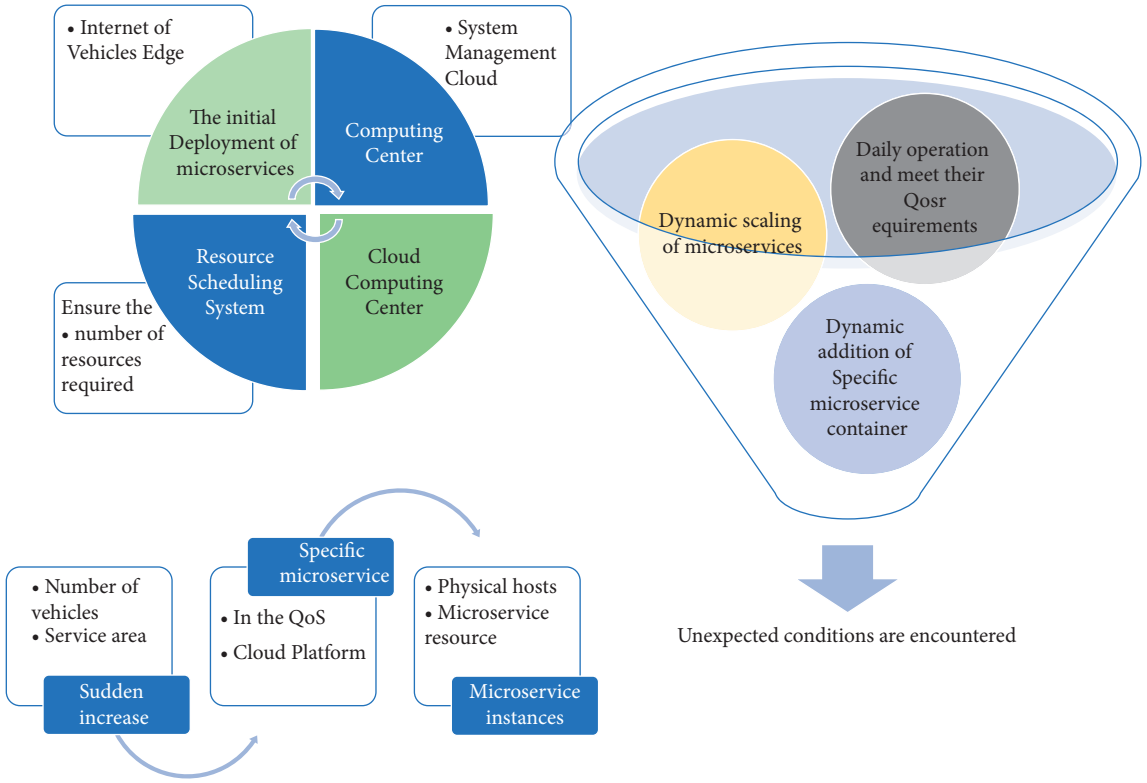

Figure 2: Schematic of cloud microservice resource scheduling in the telematics edge.

$$SR = \{\Gamma(t), f^T, v, \theta, \lambda^2, R\}, \tag{1}$$

where $\Gamma(t) = \{x, y\}$ denotes the geographic coordinates of the buyer at the time t, f^T is the buyer's local computing power (CPU cycle/s), v and θ are the buyer's speed (km/h) and direction of travel, respectively. d and $\lambda \in [0, 1]$ denote the data volume (bits) of the buyer's computationally intensive business, i.e., and the business migration rate, respectively. The data volume size (bit) of the migrated part of the business; R is the communication radius of the vehicle. Therefore, when scheduling resources for microservice containers, one of the optimization goals is to occupy the least number of physical hosts on the premise of meeting their needs, so that the resources of physical hosts can be effectively utilized.

The vehicle cloud consists of a set of computing services providing vehicles within the buyer's one-hop V2V communication range, where each member has more computing power and free resources compared to the buyer, represented by the following 7-tuple:

$$SP = \{\Gamma(t), f^S, v, \theta, \lambda^2, R^2\}, \tag{2}$$

$$P_{tot} = \sum_{k=1}^{K} \sum_{m=1}^{M} \sum_{l=1}^{L} \left(\xi P_{k,m,l}^2 - P_c\right). \tag{3}$$

The interference limit in this section considers the interference between the RSU sender pair and the receiver side of the vehicle node V, and the interference between the sender side of the K -relay forwarding and the receiver side of the vehicle node V. (3) is the interference between the K sender and the receiver V in different regions, where k, m, and$G$ denote the channel gain between the k-th relay and the m-th node on the l-th subcarrier.

$$I_{SBS} = \sum_{m=1}^{M} \sum_{l=1}^{L} P'_{k,m,l} V_{|l-m|} G^2_{k,m,l}. \tag{4}$$

In the actual process of vehicular network traffic flow data acquisition, it is often accompanied by loss communication such as sensor failure or transmission distortion, which inevitably results in the occurrence of missing, lost, or abnormal data, and may even lead to a high percentage of data loss, resulting in unreliable transmitted data [18]. Previous studies have shown that the higher-order tensor can tap higher-level data correlation, make full use of data dimensional information, improve the accuracy of data recovery, have higher accuracy and robustness, and is more suitable for solving problems such as missing data.

An analytical model of mobile IoT slices was established, abstracting different slices as different layers in a multilayer graph. The RSU can timely broadcast the vehicle density information and the priority information of the road condition warning message to the roadside cluster head vehicles in a timely manner. The aim is to solve the problems of how to deploy slices efficiently and flexibly, dynamically, and controllably allocate resources and optimize performance within and between slices according to the needs of different applications, and how to achieve highly reliable and low

latency edge computing slices in the application scenario where IoT terminals are constantly on the move, to maximize resource utilization and optimize the performance of mobile IoT slicing services, as shown in Figure 3.

In the subsequent study, we found that using only the trained neural network to predict the test set could not achieve better results, probably because the network decision space was too large and the training set could not cover all the decided cases. To solve the above problem, a genetic algorithm was used to perform a range search after the neural network decision to obtain a better decision result.

To solve (5), the task bandwidth allocation algorithm is designed because the core objective is to minimize the transmission completion time of each batch, i.e., to minimize the transmission time consumed by the last task to finish transmission within each batch, so with a certain bandwidth of the base station, the bandwidth is first allocated in equal proportion to the data size of the task, and if the bandwidth allocated to user i exceeds its bandwidth limit Bi, then if the bandwidth allocated to user i exceeds its bandwidth limit Bi, then the user's upper bandwidth limit is allocated, and then the remaining tasks are reallocated proportionally according to the above process [19]. In this way, the tasks within a batch can be transferred as simultaneously as possible, minimizing the transfer time of each batch and thus the cost function, while the algorithm also considers the bandwidth limit of each task, allowing for maximum utilization of bandwidth resources. Once the task transfer is complete, computational processing can begin, using a simple single-core processor to process incoming tasks serially, following the first-come, first-served principle, until all tasks are finally processed.

$$W_{ij} = WR_{ij}^2 RV_{ij}^2. \tag{5}$$

Since the adjacency matrix of each node of the graph random wandering model is a Markov matrix, the wandering probability of each node is a specific value in the adjacency matrix. Assuming that there are M components to be deployed, the matrix will reach a new state after M steps of wandering. The idea of routing switching is integrated, and the probability-based routing decision is made with the goal of minimizing the delay. Then, according to the final state of the matrix, the specific deployment probability of each node can be obtained, and the system needs to allocate more available resources to the node with the highest probability.

In the process of slicing resource management, this paper proposes to achieve this through a multimodel collaborative learning scheme, i.e., using Generative Adversarial Networks (GAN) and Deep Reinforcement Learning (DRL) to address resource demand prediction within slices and dynamic resource allocation between multiple slices, respectively, i.e., through multimodel collaboration to perform dynamic resource allocation for different slices to achieve efficient use of limited resources while providing slicing services for more applications.

$$\begin{aligned}\max \min V(d,G) = {} & E_{x\sim P_{\text{data}(x)}}[\log D(x) \\ & - E_{Z\sim P_Z(Z)}[1+\log D(g)]].\end{aligned} \tag{6}$$

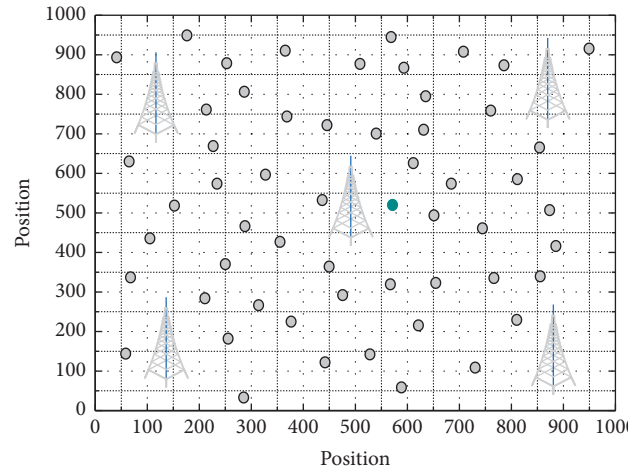

Figure 3: Network environment.

When a user sends a service request, a chain of invocations will be formed in the server to handle the user's demand. All microservices in the chain of invocations collaborate to meet the user's needs. Therefore, it is necessary to invoke the containers deployed in each physical host. The invocation of containers in the same physical host consumes significantly less time than the invocation across physical hosts. Therefore, in the process of container deployment and resource scheduling, the number of container calls across physical hosts should be minimized, so that the time for container calls across physical hosts can be reduced and the network resources wasted.

Therefore, when scheduling resources for microservice containers, one of the optimization goals is to minimize the number of physical hosts occupied while satisfying their requirements so that the resources of the physical hosts can be used effectively. In this paper, we use the defined parameter Z to denote the total number of physical hosts occupied by the deployment of microservices-equipped containers, while the combined resource utilization of the physical host population activated for the deployment of microservices in the entire edge cloud is expressed by the parameter U, as shown in the defined formula in (7).

$$Z = \sum_{i=1}^{n} P_i, \tag{7}$$

$$U = \left(\frac{\sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{l=1}^{s} p_{ij}^2 \times k_{ij}^2 \times r_{j,l}}{\sum_{m=1}^{M} \sum_{l=1}^{L} P'_{k,m,l} V_{|l-m|} G_{k,m,l}^2} \right). \tag{8}$$

The value of takes the range (0, 1), and the fewer physical hosts occupied by the same number of microservices, the higher the resource utilization of the physical machine cluster. The RSU, a network node located in the middle of a roadside or intersection, can communicate with the traffic management center and roadside vehicles to obtain timely information on the global traffic situation [24]. The centralized control mechanism allows the RSU to play an important regulatory role in congestion control and wireless channel control of in-vehicle communication. In the proposed algorithm, the RSU

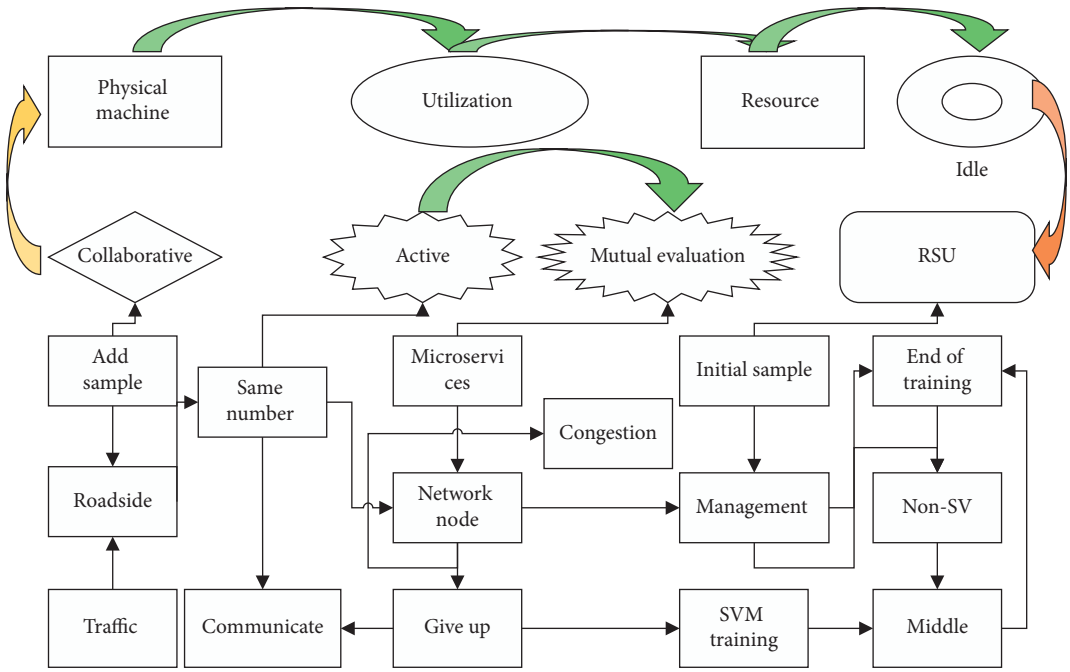

Figure 4: Channel state division diagram.

can broadcast accurate information on the density of vehicles and the priority of roadside alert messages to the roadside cluster headers on time by reusing the same communication resources. By combining the vehicle density information and the priority of the alert message, the vehicle node can calculate the information interference factor.

According to the CBR value of the cluster head vehicle, the power adjustment algorithm classifies the channel into three states, when the CBR is less than 0.5, the channel is in an idle state when the CBR is greater than 0.5 and less than 0.8, the channel is an inactive state, and when the CBR is greater than 0.8, the channel is in a congested state. The channel state transfer diagram is shown in Figure 4.

Resource management for containers is even more complicated. Commonly used container orchestration tools such as Kubernetes and Docker SwarmKit only provide some simple resource scheduling strategies and cannot fully utilize the performance of physical machines. Therefore, it is very important to design a reasonable resource scheduling strategy for microservice containers. When the channel is congested, by appropriately reducing the vehicle signal power, the conflict of vehicles competing for channel resources can be reduced, thus improving the efficiency of message delivery. When the channel is idle, the vehicle signal power is increased to improve the delivery success rate and the coverage of the alert message. When the channel state is active, the power of the vehicles in the cluster is not adjusted.

4. Analysis of Results

4.1. Performance Results of the Cloud-Edge-Vehicle-Side IoT Collaborative Resource Platform. When an IoT application starts running, the slicing system will perform business identification and related resource requirement analysis and use this as the basis for scalable deployment of functional modules. During the deployment process, depending on the type of service and demand, certain functional modules may be deployed at the core, at the edge, or even at the bearer network between the terminal and the core or other locations, enabling the flexible and scalable deployment of each functional module. These virtualized platforms for deploying different functional modules have different available resources, and the links connecting them have different transmission performances. Therefore, it is a challenge for the initialization process of mobile IoT slices to make more balanced and efficient use of limited resources when deploying specific functional modules at various locations and to further improve the quality of service and user experience of applications.

The latency in this experiment refers to the entire time that the data are sent from the terminal to the distributed computing platform until the calculation results are returned to the terminal. The goal of the multiedge collaborative mobile IoT slicing is to reduce the latency while maximizing recognition accuracy, and the experimental results confirm that the architecture achieves this goal. In contrast, edge computing relies on numerous edge devices at the edge of the network to process massive data, reduce the occupation of network resources, enhance real-time communication capabilities, and complete data processing and service execution with extremely low latency. There are three main reasons for this: firstly, the constant switching of communication nodes as the terminal moves; secondly, the multiedge collaborative slicing model that uses distributed deep learning from multiple edge nodes; and thirdly, the efficient task allocation and optimal transmission path selection method implemented through graph neural networks, as shown in Figure 5.

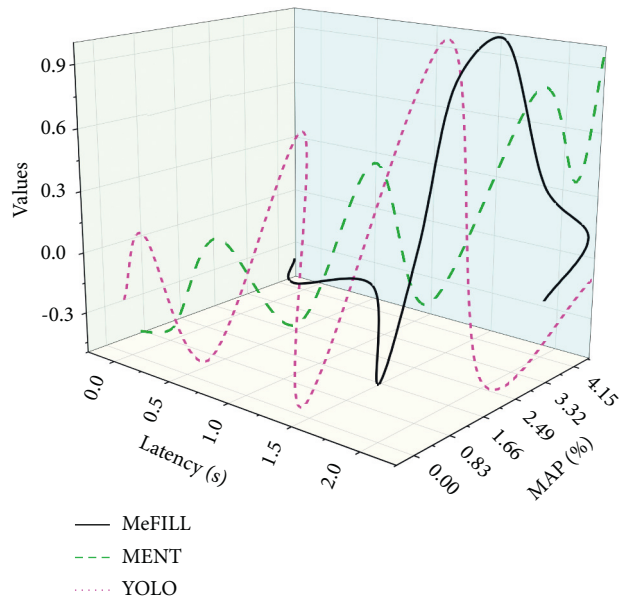

Figure 5: Comparison between time delay and accuracy.

In Figure 5, the comparison between latency and average correctness for the three architectures mentioned above is depicted. As can be seen from the figure, if higher accuracy is required, a larger latency is used. In the case of object recognition experiments, higher accuracy is used when the latency requirement is first guaranteed. For example, the requirement that the time delay must be less than 1.5 seconds enables the multiedge collaborative mobile IoT slice to achieve 85% accuracy, compared to 72% and 65% for the other two comparison models, respectively, thus showing that the multiedge collaborative mobile IoT slice has higher accuracy for the same time delay requirement, and similarly, the multiedge collaborative mobile IoT slice has lower latency for the same accuracy requirement.

As shown in Figure 6, both the delay and energy consumption in dynamic communication node mode is essentially constant, but both the delay and energy consumption in fixed communication node mode gradually increase as the terminal moves away from the communication node. This is because when the terminal is further away from the communication node, the terminal needs to increase its transmitting power to transmit data to a greater distance so that the communication node can receive it properly, which inevitably results in higher energy consumption. Similarly, the further away the terminal is from the communication node, the greater the data transmission delay. Therefore, this paper uses dynamic communication nodes to solve the problems of latency and energy consumption faced by mobile terminals. In some application scenarios of the Internet of Vehicles, such as autonomous driving, the latency requirement even needs to be lower than 10 ms. This makes the research on the transmission strategy of IoV security services more important.

In this network environment, using the CB-SIC solution (combined CB and SIC technology), all task nodes gain a total of 232-time slices to transmit data to the base station if they transmit with incremental CB power. When transmitting with fixed CB transmission power, a total of 219-time slices of data are transmitted to the base station. With the CB-only scheme (using only CB technology), only one task node in a time slice can transmit data to the base station. The interference avoidance scheme is like the CB-only scheme in this respect. The SIC-only scheme (using only SIC technology) has better data throughput than the CB-only and interference avoidance schemes but is still not comparable to the CB-SIC scheme.

By adjusting the number of tasks and idle nodes, 24 different network environments were obtained. In these network environments, the experimental results of the CB-SIC scheme, the CB-only scheme, the SIC-only scheme, and the interference avoidance scheme are compared. On-Board Units (OBUs) enable vehicle-to-vehicle (V2V), vehicle-to-road, and vehicle-to-cloud communications. The amount of data transmitted by each node in all-time slices is then calculated based on the transmission rate of the nodes, and the average data throughput in each network environment are shown in Figure 7.

As can be seen from the figure, the throughput obtained by the CB-SIC scheme is significantly improved compared to the other three schemes. Regardless of the number of task nodes and idle nodes, the CB-SIC solution consistently achieves more than twice the data throughput of the CB-only solution. The throughput can be further increased by increasing the CB transmission power, and the CB and SIC technologies increase the data throughput of the entire wireless network.

The task nodes need to transmit data directly to the base station by using CB technology. The base station, in turn, uses SIC technology to receive multiple signals simultaneously. Therefore, the container-based microservice architecture is adopted, and the application software functions are disassembled into microservices with smaller granularity for deployment, to achieve high reliability, flexibility, and high performance under limited resource conditions. In addition, the application scenarios of the edge cloud of the Internet of Vehicles are fixed. In this chapter, the structure of the network system is first given and a mathematical model is developed by analyzing it. The CB technique also generates a power gain, which further increases the number of signals that can be received and decoded at the same time by the base station using the SIC technique. In these ways, the amount of data transmitted to the base station in a fixed period is maximized. Simulation results show that the CB-SIC scheme, which combines CB and SIC technology, can significantly increase throughput compared to the CB-only scheme, the SIC-only scheme, and the interference avoidance scheme.

4.2. Performance of the Cooperative Resource Distributed Allocation Algorithm. In both the CB-SIC PRO and CB-SIC FIFO schemes, a suitability threshold is used to select an edge server for each task. Therefore, changes in the experimental results were observed by increasing the fitness thresholds in both schemes. To further demonstrate the

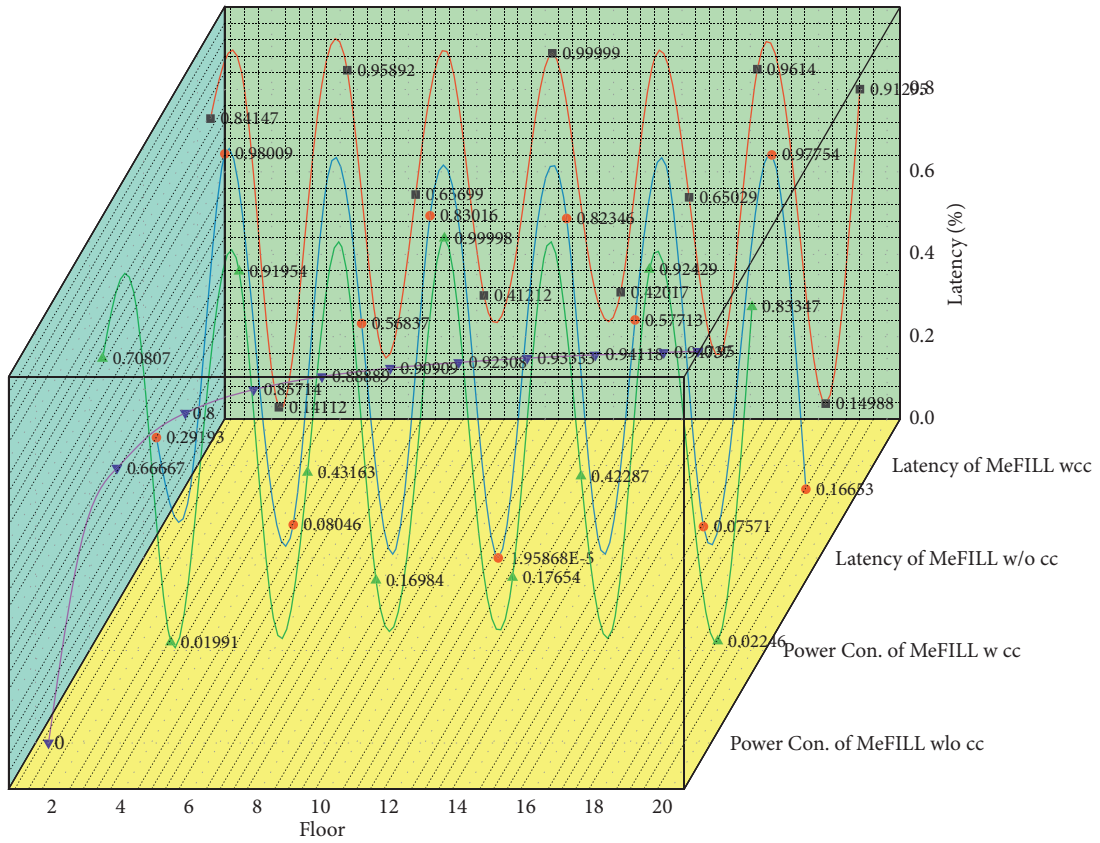

Figure 6: Comparison of terminal latency and energy consumption.

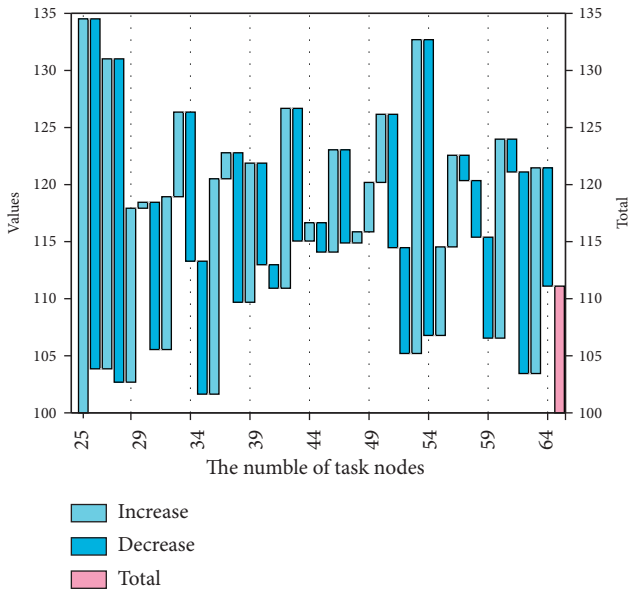

Figure 7: Schematic comparison of data throughput in different network environments.

effect of threshold α on task completion rate and task completion latency in different network environments, the number of randomly generated tasks in the network was set to 40, 50, 60, and 70.

From the function in the optimization objective, it can be analyzed that the closer the user is to the base station, the lower the bandwidth cost to achieve the same code rate. Therefore, the algorithm adopts the shortest distance access principle, which can greatly reduce the bandwidth consumption of the base station. The step-by-step training of the DNN network relies on its self-learning ability to finally obtain a better training model, and then make decisions on the incoming tasks, and output the best access strategy selected, as shown in Table 1.

It can be seen from Table 1 that the MBRA algorithm has nearly 50% of the data errors within 20%, followed by nearly 98% of the data errors within 40%, which is significantly higher than the other three heuristic algorithms, and the overall effect is better. It can also be seen from Table 1 that the average accuracy of the MBRA algorithm in the entire training process is 84.13%, the total time consumption for processing 20,000 pieces of data is 1471 s, and the time consumption for a single task decision is only 74 ms. The decision-making time of the heuristic algorithm is short, but the overall error is relatively large, so it cannot achieve a good decision-making effect.

The main purpose of the cooperative network is to make use of the high mobility and processing ability of the intelligent terminals carried by users, to make the intelligent terminals act as relays randomly, and to establish the communication between the sensor nodes and the infrastructure in a cooperative way, so that the data can be

Table 1: Time complexity and solution average error rate of MBRA and comparison algorithms.

Algorithm	Solving time (s/data)	Average error rate (%)
Base station access decision algorithm MBRA	0.074	15.87
Random access RAS	0.00452	54.19
Access to NDAS at the closest distance	0.0029	29.4
Equal distribution of access to EDS	0.00481	53.97
Brute-force algorithm VA	5.262	0

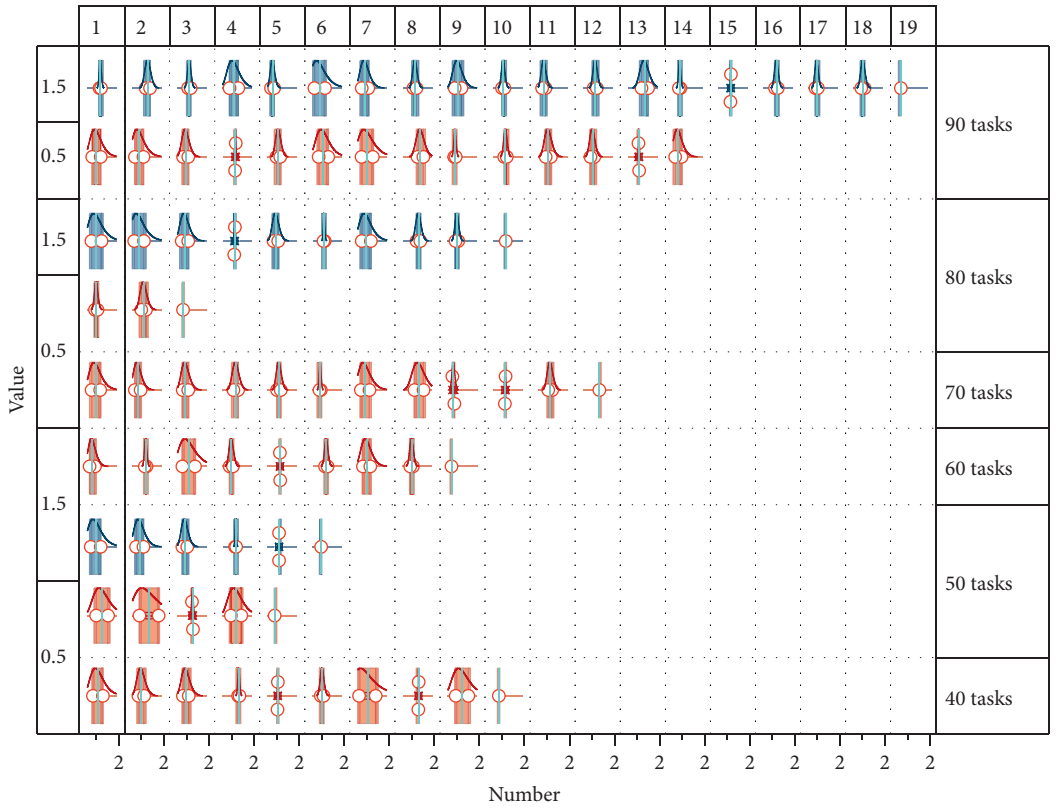

Figure 8: Diagram showing the effect of threshold α on the task completion rate of the CB-SIC PRO solution.

efficiently aggregated to the core network. Therefore, in the vehicle networking scenario, these ICVs need to consume more energy to relay the data sent by other sensor nodes. The algorithm in this paper considers the edge-end collaboration mechanism, and the ICVs closer to the base station assist the ICVs farther away from the base station to transmit data by means of data relay, thus resulting in more energy consumption.

Under the edge-end coordination mechanism of the algorithm in this paper, some ICVs that are far away from the base station do not communicate directly with the base station but use other ICVs to transmit data through multipath and multihop routing. Therefore, this algorithm can also reduce the number of links for vehicle-to-base station direct communication, thereby saving communication bandwidth resources.

Figure 8 illustrates the effect of progressively increasing suitability thresholds on the task completion rates. When the number of tasks is 40, the task completion rate increases slowly at first, and then gradually starts to decrease once it reaches 100%, and finally remains constant. When $\alpha = 0$, the task completion rate is 98.2%. When $\alpha = 14$, the task completion rate reaches 100%. At $\alpha = 22$, this starts to decrease and eventually stays at 98.2%. The task completion rate of the CB-SIC PRO solution remains at a high level for a task count of 40, and it can be assumed that the computational load from the task count at this point is not high for the edge servers in the network. Therefore, the impact is not significant. As the number of tasks increases, the task completion rate varies considerably. At a task count of 60, the task completion rate ranged from a low of 79.1% to a high of 94.8%. There is a 15.7% difference between the minimum and maximum.

The experimental results in Figure 9 show that the degree of load imbalance in the data center increases as the size of the microservice containers to be deployed increases. Then, the scheduling system deploys it to the physical machine to run. Its essence is that the scheduling system schedules the container set for deploying microservices to run on the physical machine set according to the scheduling policy, and

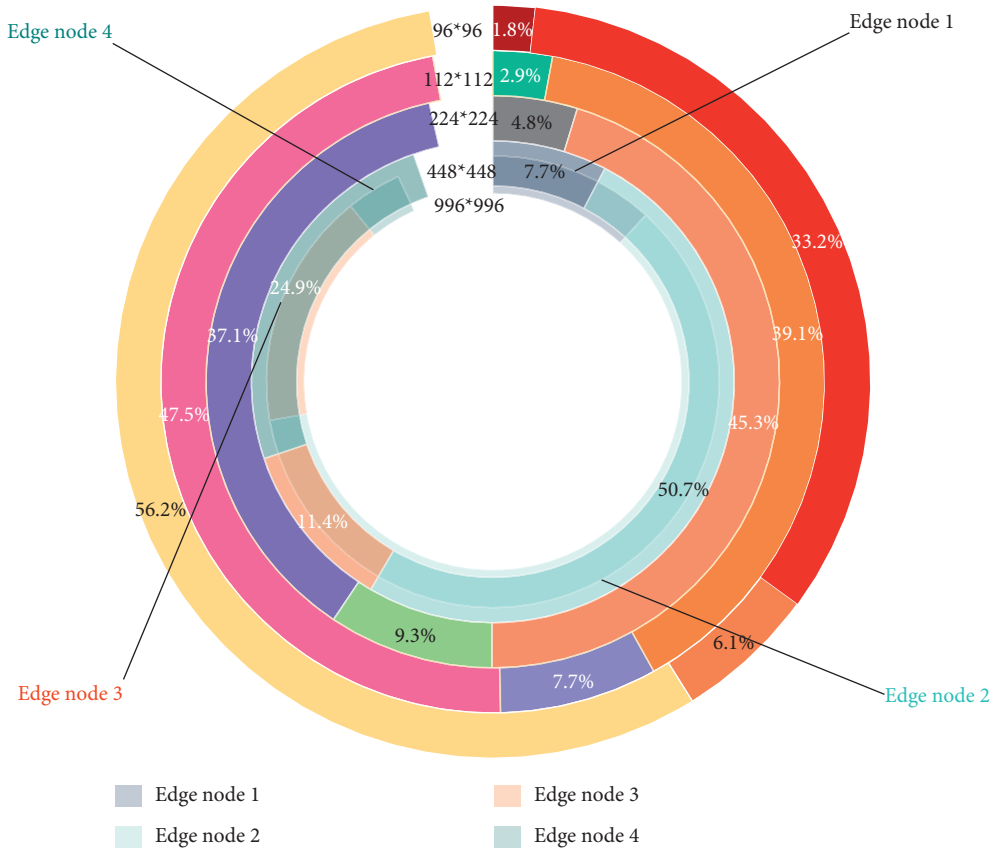

Figure 9: VGG16 model and the effect of feature map size on edge clustering.

the container can be configured with the resources required by the microservice program. For the same experimental conditions, the standard deviation of the data center load imbalance of the MFGA algorithm is better than the other three algorithms and stays in the lower range, achieving good load balancing.

From the experimental results in the previous section, communication time consumption has a more obvious impact on the overall time consumption, and the amount of data determines the communication time consumption. Here, we investigate the impact of the size of the input feature map of the deep network model on the computation time and acceleration ratio of the edge clusters, with five VM VMs-1 used as edge devices, one as the edge gateway and the remaining four as edge nodes, the number of data frames is 1, and the network bandwidth is set to 1000 Mbps.

Under the mechanism that the ICV communicates directly with the base station, the total energy consumption of video compression and communication varies greatly for each ICV, that is, the total energy consumption of video compression and communication of ICVs closer to the base station is higher than that of the ICV. This is because the ICVs that are closer to the base station have better signal-to-noise ratios of transmission channels and can support higher data transmission rates with less communication energy consumption.

One limitation of the value iteration method is that it requires a finite and minimal number of states, which makes solving the system of equations almost impossible when, as in this paper, the state space is growing exponentially. The policy iteration algorithm includes a process of policy estimation, which requires scanning all states several times, a complexity that seriously affects the efficiency of the policy iteration algorithm. Both value iteration and policy iteration require a known state transfer probability to compute the optimal policy, which is difficult to implement in real-world usage scenarios.

5. Conclusion

The Internet of Vehicles, as an application of the Internet of Things in the field of intelligent transportation, is an important development direction for future driving technology. In an IoT environment, it is impossible to realize autonomous driving without the collaborative cooperation of cloud, edge, and end. As a data processing center and application software deployment platform close to the end service terminal in the Telematics system, the edge cloud platform will carry most Telematics applications. As it is deployed at the edge of the network, it can greatly shorten the response time of Telematics applications, reduce bandwidth costs, and improve service quality. The edge cloud platform has relatively limited resources compared to traditional cloud data centers, and its resource scheduling algorithm directly affects the performance of Telematics applications that are not on it. Therefore, it is important how

to make limited use of edge cloud hardware resources in the Telematics edge cloud platform and ensure the reliability and high performance of Telematics applications. In this strategy, firstly, the clustering algorithm is used to cluster the vehicles on the road, secondly, it is introduced how to evaluate the channel busy status of the clustered vehicles using the RSU of the roadside node, then the vehicles are classified into different channels states according to different channel busy degrees, and the corresponding power adjustment strategies are carried out by the vehicles in different channel states to improve the communication performance of the alarm messages transmitted on the road. Finally, the performance of the proposed algorithm is effectively verified in simulation experiments.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Consent

Informed consent was obtained from all individual participants included in the study references.

Consent

The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Conflicts of Interest

The authors declare no conflict of interest.

Acknowledgments

This work was supported by 2019 Cross Science Research Project of Nanyang Institute of Technology, Grant No. 201913502, Research on Intelligent Mining and Recommendation of Zhang Zhongjing Prescription Based on Deep Neural Network, and Henan Science and Technology Plan Project, Grant No. 222102210134, Research on Key Technologies of Cloud Security Desktop Based on Kunpeng Architecture.

References

[1] Y. Ding, M. Jin, S. Li, and D. Feng, "Smart logistics based on the internet of things technology: an overview," *International Journal of Logistics Research and Applications*, vol. 24, no. 4, pp. 323–345, 2021.

[2] K. Kardaras, G. I. Lambrou, and D. Koutsouris, "Telematics healthcare through digital terrestrial television networks: applications and perspectives," *International Journal of Sensors, Wireless Communications & Control*, vol. 11, no. 5, pp. 560–576, 2021.

[3] H. Tavolinejad, M. R. Malekpour, N. Rezaei et al., "Evaluation of the effect of fixed speed cameras on speeding behavior among Iranian taxi drivers through telematics monitoring," *Traffic Injury Prevention*, vol. 22, no. 7, pp. 559–563, 2021.

[4] D. Ivanov, C. S. Tang, A. Dolgui, D. Battini, and A. Das, "Researchers' perspectives on Industry 4.0: multi-disciplinary analysis and opportunities for operations management," *International Journal of Production Research*, vol. 59, no. 7, pp. 2055–2078, 2021.

[5] O. R. Sánchez, C. A. Collazos Ordonez, M. A. Redondo, and I. Ibert Bittencourt Santana Pinto, "Homogeneous group formation in collaborative learning scenarios: an approach based on personality traits and genetic algorithms," *IEEE Transactions on Learning Technologies*, vol. 14, no. 4, pp. 486–499, 2021.

[6] S. Jung, J. Kim, M. Levorato, C. Cordeiro, and J. H. Kim, "Infrastructure-assisted on-driving experience sharing for millimeter-wave connected vehicles," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 8, pp. 7307–7321, 2021.

[7] K. Yue, Y. Zhang, Y. Chen et al., "A survey of decentralizing applications via blockchain: the 5g and beyond perspective," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2191–2217, 2021.

[8] C. H. Lai and J. S. Fu, "Exploring the linkage between offline collaboration networks and online representational network diversity on social media," *Communication Monographs*, vol. 88, no. 1, pp. 88–110, 2021.

[9] Q. Yu, M. Wang, H. Zhou, J. Ni, J. Chen, and S. Cespedes, "Guest editorial special issue on cybertwin-driven 6G: architectures, methods, and applications," *IEEE Internet of Things Journal*, vol. 8, no. 22, pp. 16191–16194, 2021.

[10] G. Fodor, J. Vinogradova, P. Hammarberg et al., "5G new radio for automotive, rail, and air transport," *IEEE Communications Magazine*, vol. 59, no. 7, pp. 22–28, 2021.

[11] Z. H. Ali and H. A. Ali, "Towards sustainable smart IoT applications architectural elements and design: opportunities, challenges, and open directions," *The Journal of Supercomputing*, vol. 77, no. 6, pp. 5668–5725, 2021.

[12] C. Na, D. Lee, J. Hwang, and C. Lee, "Strategic groups emerged by selecting R&D collaboration partners and firms' efficiency," *Asian Journal of Technology Innovation*, vol. 29, no. 1, pp. 109–133, 2021.

[13] H. G. Abreha, C. J. Bernardos, A. D. L. Oliva, L. Cominardi, and A. Azcorra, "Monitoring in fog computing: state-of-the-art and research challenges," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 36, no. 2, pp. 114–130, 2021.

[14] K. Yang, C. Hu, Y. Qin, Y. Huang, and X. Tang, "Potential and challenges to improve vehicle energy efficiency via V2X: literature review," *International Journal of Vehicle Performance*, vol. 7, no. 3/4, pp. 244–265, 2021.

[15] C. Sardianos, I. Varlamis, C. Chronis et al., "The emergence of explainability of intelligent systems: delivering explainable and personalized recommendations for energy efficiency," *International Journal of Intelligent Systems*, vol. 36, no. 2, pp. 656–680, 2021.

[16] C. Englund, E. E. Aksoy, F. Alonso-Fernandez, M. D. Cooney, S. Pashami, and B. Astrand, "AI perspectives in Smart Cities and Communities to enable road vehicle automation and smart traffic control," *Smart Cities*, vol. 4, no. 2, pp. 783–802, 2021.

[17] A. Rajesh and S. Shaffath Hussain Shakir, "Investigations on scheduling algorithms in LTE-advanced networks with carrier aggregation," *International Journal of Advanced Intelligence Paradigms*, vol. 18, no. 2, pp. 1–62, 2021.

[18] B. Bhushan, C. Sahoo, P. Sinha, and A. Khamparia, "Unification of Blockchain and Internet of Things (BIoT): requirements, working model, challenges and future directions," *Wireless Networks*, vol. 27, no. 1, pp. 55–90, 2021.

[19] S. Guan, J. Wang, C. Jiang, R. Duan, Y. Ren, and T. Q. S. Quek, "MagicNet: the maritime giant cellular network," *IEEE Communications Magazine*, vol. 59, no. 3, pp. 117–123, 2021.

[20] Y. Zhang, K. C. S. Lee, and D. Adams, "Visualizing research in educational technology leadership using CiteSpace," *Int. Online J. Educ. Leadership*, vol. 5, pp. 61–77, 2021.

[21] J. Moeyersons, S. Kerkhove, T. Wauters, F. De Turck, and B. Volckaert, "Towards cloud-based unobtrusive monitoring in remote multi-vendor environments. Software: practice and Experience," *Software: Practice and Experience*, vol. 52, no. 2, pp. 427–442, 2022.

[22] B. Chan, "Sidecar learning vs LibWizard: a comparison of two split-screen tutorial platforms," *Journal of Web Librarianship*, vol. 15, no. 2, pp. 90–103, 2021.

[23] C. Simon, M. Maliosz, M. Máté, D. Balla, and K. Torma, "Sidecar based resource estimation method for virtualized environments," *INFOCOMMUNICATIONS JOURNAL*, vol. 12, no. 2, pp. 4–11, 2020.

[24] P. Jamshidi, C. Pahl, N. C. Mendonça, J. Lewis, and S. Tilkov, "Microservices: the journey so far and challenges ahead," *IEEE Software*, vol. 35, no. 3, pp. 24–35, 2018.