Wireless Communications and Mobile Computing

Edge Intelligence for 6G-enabled Industrial Internet of Things

Lead Guest Editor: Zhaolong Ning
Guest Editors: Amr Tolba and Xiangjie Kong

Edge Intelligence for 6G-enabled Industrial Internet of Things

Wireless Communications and Mobile Computing

Edge Intelligence for 6G-enabled Industrial Internet of Things

Lead Guest Editor: Zhaolong Ning
Guest Editors: Amr Tolba and Xiangjie Kong

Contents

Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 8331642, 15 pages
https://doi.org/10.1155/2023/8331642

Research Article

Spatiotemporal Self-Attention-Based Network Traffic Prediction in IIoT

Xiaoteng Liu,[1,2] **Chuanhe Huang**,[1,2] **M. Wasim Abbas Ashraf**,[1,2] **Shidong Huang**,[1,2] **and Yirong Chen**[1,2]

[1]*School of Computer Science, Wuhan University, Wuhan 430072, China*
[2]*Hubei LuoJia Laboratory, Wuhan 430072, China*

Correspondence should be addressed to Chuanhe Huang; huangch@whu.edu.cn

Received 29 March 2022; Revised 9 May 2022; Accepted 14 June 2023; Published 14 July 2023

Academic Editor: Ashish Bagwari

The sixth-generation (6G) mobile communications are considered as a future network and very closed to the Industrial Internet of Things (IIoT) due to its low latency and high throughput. Massive nodes supported by 6G make up the complexity of the network. Moreover, the heterogeneous traffic brings difficulties to the network management. Long-term network traffic matrix (TM) prediction is a crucial technology for realizing network edge intelligence and dealing with the above issues. However, predicting long-term network traffic in heterogeneous IIoT is challenging. Due to the powerful feature extraction capability over long sequences, self-attention is widely applied in language inference tasks. Motivated by these observations, we propose a self-attention traffic matrix prediction (SATMP) model for long-term network TM prediction in IIoT scenarios. SATMP consists of three components: (a) a spatial–temporal encoding for obtaining the spatial–temporal features of network TM; (b) a learnable positional encoding for providing positional correlation to the traffic sequence; and (c) a self-attention module for capturing long-term dependence. These components work together to enhance long-term prediction performance in complex networks effectively. Extensive experiments on three publicly available datasets demonstrate that SATMP is feasible and accurate in IIoT long-term network TM prediction.

1. Introduction

The sixth-generation (6G) has superior features to previous generations of mobile communications, such as low latency communications, high throughput, and massive connection [1]. These advantages will bring new developments to Industrial Internet of Things (IIoT), especially the wide application of edge intelligence [2, 3]. Edge intelligence is a combination of edge computing and artificial intelligence, which deploys machine learning algorithms to edge devices and gets closer to data sources. Edge intelligence relies on edge devices to cooperate with other devices to complete tasks [4], shortening the data transmission delay. However, IIoT is characterized by large-scale heterogeneity and requires high fault tolerance. The application of edge intelligence will make computing, storage, and communication in IIoT more complex. These changes not only put forward the higher requirements for network quality of service (QoS), but also bring more challenges to network management:

(i) Data transfer security and stability: IIoT calculates, generates, stores large amount of production and user data. A secure and reliable network is needed to protect industrial data and user privacy [5, 6]. In addition, IIoT networks are densely connected and have expensive network equipment, which requires protection against industrial accidents caused by network latency and network attacks.

(ii) Efficient resource allocation: IIoT has a large number of sensor devices and the amount of data that the nodes need to collect and transmit increases significantly. The computing, storage, and communication capabilities of individual nodes are limited in IIoT. In this situation, deploying a complex algorithm framework may lead to higher network latency and energy consumption [7].

(iii) High heterogeneity of the network: 6G enables IIoT to have a large number of mobile devices and sensors

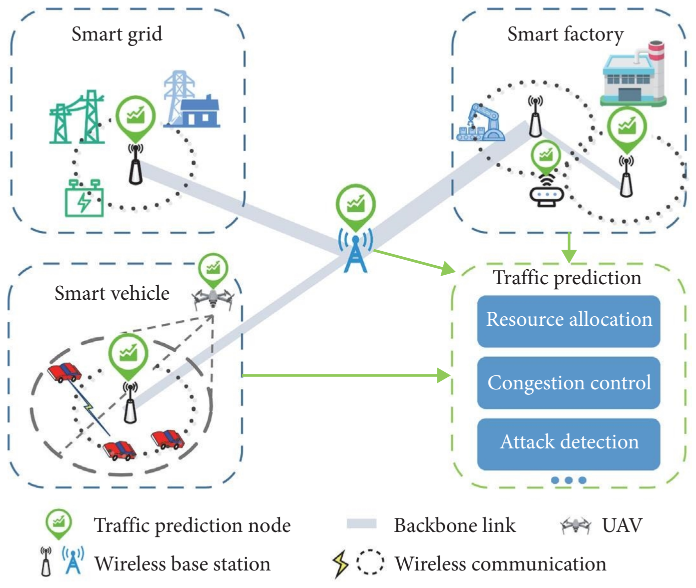


Figure 1: An illustration of edge network traffic prediciton in IIoT.

such as smart vehicles and unmanned aerial vehicles (UAVs) [8, 9], and so forth. Numerous mobile and fixed nodes are connected by wired and wireless networks [10]. They have different communication technologies and protocols, which require well-designed compatible protocols and network architectures [11].

Network traffic matrix (TM) prediction is one of the critical technologies of network edge intelligence and an effective method to deal with the above challenges. It uses the historical traffic of the network to predict the network traffic in the future. Accurate network traffic prediction results can provide reliable data support for intelligent network management. Its ability in load balancing, congestion control, and security warning can effectively improve the stability and operation efficiency of IIoT [12]. For example, TM prediction can provide early warning of abnormal traffic changes and guard against attacks such as flooding attacks. Besides, the predicted results help to allocate resources appropriately. According to the prediction result, some core nodes can release some computing resources during low flow periods. Traffic prediction can also help IIoT to automatically configure, supply, and test network equipment and routing algorithms and helps build better network structures. This paper focuses on the long-term prediction of the network TM in IIoT and building edge intelligence. The network TM contains the network traffic of each pair of origin–destination nodes within a sampling interval and represents the overall state of the IIoT network. Furthermore, compared with the short-term forecast, the long-term forecast reserves more time for network configuration and testing [13]. On the one hand, network managers can use the long-term prediction results to judge the long-term impact of the current operation on the future network. On the other hand, the long-term prediction results leave more time for network configuration and testing, which is more significant than the short-term prediction.

Figure 1 shows the edge network traffic prediction in IIoT. We list three scenarios for IIoT: smart cars, smart grids, and smart factories. Wireless base stations are connected through backbone links. Base stations on the edge provide wireless access to various nodes, such as smart vehicles, industrial sensors, industrial equipment, and electricity pylons. At the same time, UAV-assisted communication is available in some complex areas [14]. Wireless networks are also used to communicate between nodes. The heterogeneous IIoT scenario makes it difficult to predict network traffic. The proposed self-attention traffic matrix prediction (SATMP) can be deployed on edge computing nodes, such as wireless base stations, vehicles, UAVs, and so forth. Edge intelligence can quickly process computing tasks on edge without being limited to the network conditions of cloud computing [15]. IIoT can be better managed by referring to the predicted results. For example, primary routing nodes can reasonably allocate bandwidth and channels to improve resource utilization. Factories and grids can detect attacks based on changes in the network traffic pattern.

In essence, the prediction of network TM is a time-series prediction problem. However, the network TM series in IIoT is significantly different from other time series. First, network traffic in IIoT is stochastic and dynamic, which leads to more complex statistical characteristics. Existing studies have shown that network traffic has self-similarity, long-time dependence, heavy tail distribution [16], and other characteristics. Second, network traffic is affected by other factors besides time. For example, backbone networks in IIoT are affected by network topology and routing settings, while wireless communication is influenced by movement patterns and specific functions of mobile nodes. All the above factors bring difficulties to IIoT traffic prediction. In addition, long-term prediction is another challenge, which makes some models lose their predictive ability.

The attention mechanism originated from human vision. It assigns different weights to different input parts. The self-attention mechanism is a variant of the attention mechanism. The self-attention mechanism is first proposed by Lin et al. [17] for extracting an interpretable sense embedding. The self-attention mechanism reduces the dependence on external information and better captures the internal correlation of data. In recent years, the self-attention mechanism has made remarkable achievements in the natural language processing (NLP) field and achieved leading results in tasks such as text translation and language information [18, 19]. The self-attention module can calculate the data of different time steps in parallel through reasonable position encoding without attaching them to the recurrent neural network. It shortens the distance between any two inputs significantly to enhance the learning ability of long-term dependence.

Motivated by the above analysis, we utilize the self-attention mechanism for IIoT traffic prediction and propose a network TM prediction model based on spatiotemporal features. We first add spatial and temporal encoding to the data to effectively utilize the spatiotemporal information of network TM. Then, the self-attention mechanism is used to learn the spatiotemporal characteristics. In the encoder, we design a learnable positional encoding so that the model can perceive the order of input data. Meanwhile, positional encoding is used to ensure the parallelization of training. The main contributions of this paper are as follows:

(i) We propose SATMP, a novel model based on the self-attention mechanism for long-term network TM prediction in IIoT. The self-attention mechanism with spatiotemporal encoding is used to learn the complex associations of network traffic.

(ii) We design a learnable positional encoding scheme to provide position correlation for the long input sequence, enhancing model awareness of input order.

(iii) We evaluate the performance of SATMP on three publicly available datasets. The results show that SATMP is feasible and outperforms other methods in predicting accuracy.

The rest of this paper is organized as follows. Section 2 overviews the studies related to traffic prediction and describes their characteristics. In Section 3, we first provide a detailed definition of long-term IIoT traffic matrix prediction and then elaborate on the related challenges. The framework and details of the proposed algorithm are discussed in Section 4. In Section 5, we introduce the datasets we use, describe the experiment's specific details, and analyze the experimental results. Finally, we summarize the research of this paper and discuss our future work.

2. Related Work

Network traffic prediction is one of the hot topics in network characterization and measurements. It has attracted the wide attention of researchers, and the related literature is abundant in some specific contexts.

Many studies use statistical models to predict network traffic. For example, Moayedi and Masnadi-Shirazi [20] decompose network traffic into normal and abnormal parts and use the autoregressive integrated moving average model (ARIMA) to predict network traffic and detect anomalies. Wang et al. [21] propose a traffic flow modeling and prediction method based on an autoregressive moving average model (ARMA), which is easy to calculate. The autoregression-based model uses the linear combination of historical sequence data to generate predictions results. It cannot model the nonlinear features in the flow and requires the sequence to have a certain stability. Kim [22] uses integer-valued generalized autoregressive conditional heteroscedasticity (INGARCH) to capture the nonlinear characteristics of network traffic. Bayati et al. [23] use Gaussian process regression (GPR) to predict the flow and use self-similar covariance functions to improve the prediction accuracy. However, the above models usually do not take into account the spatial connection or interdependence of network nodes. The 6G-enabled IIoT tends to contain much more nodes and links, limiting the use of these statistics-based methods in more general problems.

At present, the mainstream methods pertain to machine learning and deep learning. Nikravesh et al. [24] compare the performance of support vector machines (SVM), multilayer perceptron with weight decay (MLPWD), and multilayer perceptron (MLP) in traffic prediction tasks. Jain and Prasad [25] use the XGBoost algorithm to predict the traffic of telecom network in peak hours. The machine learning algorithm can mine some complex network traffic patterns, but compared with deep learning, its feature extraction ability is relatively insufficient. Many researchers have used deep learning algorithms with stronger feature extraction ability, such as convolutional neural network (CNN) [26], deep belief network (DBN) [27], recurrent neural network (RNN) [28], long short-term memory networks (LSTM) [29], meta-learning method [30]. Some scholars specifically study traffic prediction methods in IIoT. For instance, Nie et al. [31, 32] design two prediction methods based on multitask learning and reinforcement learning, respectively, in complex and heterogeneous IIoT. Compared with linear and machine learning methods, the deep learning model is more complex and can better model the time dependence of traffic flow series. RNN is more suitable for processing time series with shared parameters in the time dimension. However, the recurrent structure of RNN is easy to cause gradient disappearance and gradient explosion when using too many units. LSTM adopts cell states and three gates: an input gate, an output gate, and a forget gate to alleviate the above problems. It could effectively learn the long-term correlation characteristics of the traffic. Gated Recurrent Unit (GRU) is a variant of LSTM that reduces the gates to two. LSTM and GRU have obtained better prediction results and have become a widely used traffic prediction model in recent years [33, 34].

Some researchers combine different models to improve performance. Compared with using these models alone, the hybrid model has improved the prediction effect. For example, Li et al. [35] propose a method combining wavelet transform and artificial neural network (ANN), which uses wavelet transform to decompose time-domain traffic and adds nonlinear prediction ability to the model with the help of ANN to reduce the prediction error. Similarly, Tian et al. [36] use the Mallat wavelet transform algorithm to decompose the network traffic, then use ARIMA and least-squares support vector machine (LSSVM) to predict different components, respectively. Zhang et al. [37] combine wavelet transform with LSTM and reduce the prediction error. Zhao et al. [38] and Tian et al. [39] decompose complex network data into low-frequency smooth sequence through empirical mode decomposition. Then they use LSTM and ARMA, respectively to predict, effectively improving the prediction performance.

The literature analysis shows that more and more researchers have preferred neural networks and their hybrid models in recent years. RNN, LSTM, and their variant networks can better model the network traffic sequence because of their solid time-dependent learning ability. Compared with linear and machine learning models, those models have a significant advantage in prediction accuracy.

In addition, the new research trend is the application of attention mechanisms in the network structure to enhance the model's attention to spatiotemporal correlation. For example, Feng et al. [40] propose a network traffic prediction model with attention mechanisms to capture long and complex dependencies. Zhao et al. [41] propose a spatial–temporal attention-CNN to effectively obtain the dynamic spatiotemporal correlations of cellular networks. In order to fully learn the characteristics of the IIoT long-term network TM sequence, we apply the self-attention mechanism to network TM prediction. We aim to improve the long-term prediction performance by using the powerful long-term feature extraction capability of the self-attention mechanism.

3. Problem Formulation

This section elaborates on the IIoT long-term TM problem and related challenges. We first define the long-term prediction problem of network TM in IIoT. Then, we analyze the challenges of the research problem from three aspects: multivariate, spatial–temporal correlation, and long-term dependence.

3.1. Long-Term IIoT TM Prediction. In order to facilitate the subsequent consideration of topological and spatial factors, we use a weighted directed graph to model the IIoT network. Considering an IIoT with N nodes, which are connected by H links, we establish a directed graph $G = (V, I)$, where V is the set of nodes, and I is the set of links. For the backbone network, if there is a link between node i and node j, then I has two edges (v_i, v_j) with different directions. The weight of each link represents the route weight of the link. For wireless networks, we think there is a direct connection between any two research nodes regardless of their link weight. The traffic in the IIoT network TM includes the traffic flow that each origin node sends to other destination nodes within a certain interval. The definitions covered in this paper are as follows:

Definition 1. The IIoT network traffic matrix $\boldsymbol{M}^t \in \mathbb{R}^{N*N}$ is:

$$\boldsymbol{M}^t = \begin{bmatrix} m_{1,1}^t & \cdots & m_{1,N}^t \\ \vdots & \ddots & \vdots \\ m_{N,1}^t & \cdots & m_{N,N}^t \end{bmatrix}, \quad (1)$$

where N is the number of nodes in the IIoT network, $m_{i,j}^t$ represents the traffic flow sent from node i to node j in the t sampling interval.

Definition 2. The network traffic vector $\boldsymbol{X}^t \in \mathbb{R}^{N^2}$ is a transformation of $\boldsymbol{M}^t$. $\boldsymbol{X}^t$ is made up of m_k^t, where k is calculated by $k = (i-1) \times N + j$. $\boldsymbol{X}^t$ is:

$$\boldsymbol{X}^t = \left\{ m_1^t, m_2^t, \dots, m_{N^2}^t \right\}. \quad (2)$$

Assuming that the number of the obtained traffic is T, all the obtained traffic can be donated as $\boldsymbol{TM} = \{\boldsymbol{X}^1, \boldsymbol{X}^2, \dots, \boldsymbol{X}^T\} \in \mathbb{R}^{N^2 \times T}$. Then, the prediction task in this paper is using the historical flow $\boldsymbol{S} = \{\boldsymbol{X}^{t-\omega+1}, \boldsymbol{X}^{t-\omega+2}, \dots, \boldsymbol{X}^t\}$ to predict the flow $\boldsymbol{Y} = \{\boldsymbol{X}^{t+1}, \boldsymbol{X}^{t+2}, \dots, \boldsymbol{X}^{t+l}\}$, where ω is the historical length and l is the traffic length to be predicted. The formula is as follows:

$$\boldsymbol{Y} = f(\boldsymbol{X}^{t-\omega+1}, \boldsymbol{X}^{t-\omega+2}, \dots, \boldsymbol{X}^t; \boldsymbol{R}; \boldsymbol{P}), \quad (3)$$

where $\boldsymbol{R}$ is the spatial correlation of the network and $\boldsymbol{P}$ is the timestamp. We predict longer lengths than previous work [33, 42] in our work. The goal is to build a model that achieves low error between predicted value $\widehat{\boldsymbol{Y}}$ and ground truth $\boldsymbol{Y}$.

3.2. Challenges. In addition to the challenges in short-term time-series prediction, such as multiple factors influence, prediction lag, high randomness, and so forth; the long-term prediction of IIoT TM faces more challenges due to the complexity of network traffic itself and the influence of potential factors:

(i) Multiple dimensions: the network TM represents all the origin–destination traffic in the IIoT network. Assuming an IIoT has N nodes, there are N^2 traffic flows in the network TM. Compared with single-dimensional time series, multiple dimensional traffic prediction requires higher prediction performance and hardware consumption. Models must have solid predictive power to simultaneously capture deeper features and predict all dimensions. In addition, nodes in IIoT have limited power and computing capacity [43], so the computational time and space complexity of the prediction model should not be high.

(ii) Complex temporal and spatial associations: some IIoT, such as smart vehicles and smart logistics, are designed to provide services for human beings. Their traffic is closely related to human activities, so there is a temporal correlation between future traffic and historical traffic. However, the temporal correlation pattern has different dimensions, such as quarter, month, and week. In other cases, IIoT traffic has a high suddenness in fine granularity, which brings difficulties to learning association mode. In addition, the change in network traffic is affected by spatial association, such as other network nodes, network topology, routing algorithm, link bandwidth, and so forth. For example, many backbone networks use the Open Shortest Path First (OSPF) gateway protocol, which generally uses the shortest path algorithm, for example, Dijkstra, to construct a routing table. Network packets choose the route to the destination node according to the routing table, which affects the traffic changes of different nodes and links.

(iii) Long-term dependencies: it is difficult to capture the long-term dependencies. The prediction target of many previous studies is the network traffic situation at the next interval [32, 44], while the purpose of our study is to predict the change of network traffic in a long period (such as the next 48 hr). Long-term prediction requires a more vital ability to model long-term dependencies. Many existing models have limited ability to capture long-term dependencies. For instance, RNN and LSTM have the advantage of processing time-series data. However, their performance will decrease in long-term prediction tasks. For example, Zhou et al. [45] prove that as the prediction length increases, the inference time of LSTM becomes longer, and the prediction error increases.

In view of the above challenges, we designed a more efficient prediction scheme of self-attention mechanism for IIoT. We use the parallelism capability of the self-attention mechanism to improve the processing capability of multidimensional data. In addition, we design spatiotemporal encoding to provide characteristics of IIoT network TM sequences. Last but not least, we design learnable position encoding to provide time correlations for long input sequences. The framework and details of the model are explained in Section 4.

4. System Model

In this section, we first introduce the main framework of the proposed model. Then we discuss the various modules in our proposed work, including spatiotemporal encoding, learnable positional encoding, self-attention module, feedforward layer, and output layer. The main notations covered in this article and their descriptions are summarized in Table 1.

Table 1: Key notations.

Notation	Description
G	Graph of a network
N	Numbers of nodes
T	Numbers of network TM
$\boldsymbol{T}^t$	The t_{th} traffic vector
$\boldsymbol{R}$	Spatial encoding
$\boldsymbol{P}$	Timestamp encoding
d	Dimensions of traffic vector
ω	Length of input sequence
l	Length of target sequence
$\boldsymbol{E}$	Sum of different encodings
$\boldsymbol{W}_*$	Weight of fully connected layer

4.1. Main Framework. Figure 2 shows the main framework of the prediction model with the spatiotemporal self-attention mechanism proposed in this paper. The model's input consists of three parts: network TM, traffic flow timestamp, and network topology and route weight. We first add spatial and temporal encodings to the preprocessed flow matrix sequence. Then the sequence is input into the stacked encoders to learn the temporal and spatial characteristics and dependent correlations. Every encoder consists of learnable positional encoding, a multihead self-attention module, and a feedforward neural network. The positional encoding provides position relations for input sequences. The temporal and spatial dependence of different sequences is learned by the self-attention modules. The feedforward neural networks provide nonlinear transformations for each encoder. Residual connection is used between the three parts to prevent network degradation and enhance network stability. Finally, the fully connected layer is used to output the prediction results.

4.2. Spatiotemporal Encoding. In urban traffic flow prediction, the spatial dependencies between road segments are of great importance, which has a significant influence on the change of traffic flow [46]. Unlike the transportation system, the relative position of nodes in the network is not essential, while the interactive relationship between nodes or regions has a more significant impact on network traffic. In the backbone network scenario, we utilize the network topology and routing protocol to build spatial encoding. The network topology and route weight are used to calculate the shortest path matrix of the network. The transformed routing matrix is added into the model as spatial routing encoding to enhance the model's perception of spatial dependence. For graph G, we first use the shortest path algorithm corresponding to the network, such as Dijkstra, to calculate the routing distance of each pair of origin–destination nodes according to the weight of I. The routing matrix is donated by $\boldsymbol{R}$. Then we flatten $\boldsymbol{R}$ in the way just like we map the TM. Since the path selection strategy takes a shorter path, we invert every element in $\boldsymbol{R}$. Then we normalized $\boldsymbol{R}$ to prevent interference with the original data. In the wireless network scenario, we build the spatial encoding of network TM by utilizing the functional area type of telecommunication region. First, we use Google Map to find the points of interest (PoIs) of each study region. Next, we determine the functional area types of all study regions by considering the PoIs and CORINE Land Cover (CLC) map [47]. CLC provides information on land cover and land cover change across Europe. Based on satellite images, the land is divided into urban fabric, industrial or commercial fabric, green urban areas, and so forth. The dataset of wireless cellular network used in this paper was collected from 2013 to 2014, so the

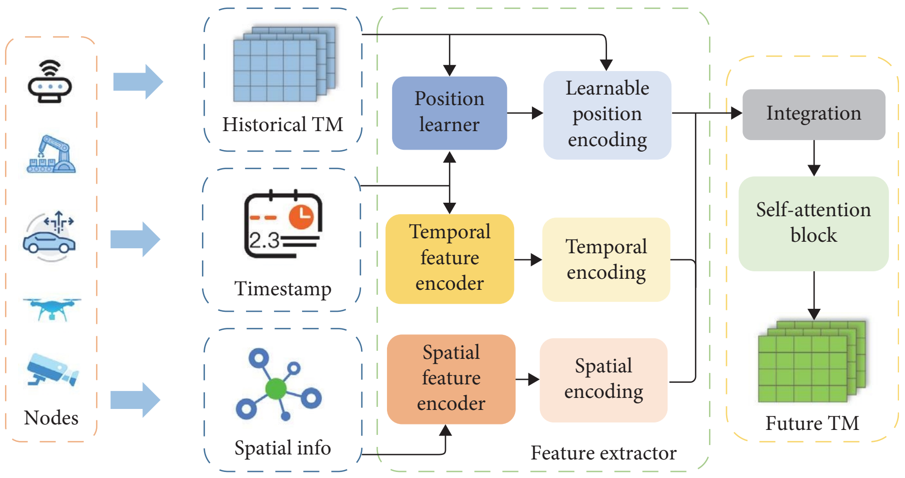


Figure 2: Main framework of spatio-temporal self-attention network TM prediction model.

Table 2: The functional areas of wireless dataset.

Functional areas	Related PoIs
Residential area	Living quarters and villages
Business district	Hotel, supermarket, and club
Industrial area	Factory and garage
Suburb	Farmhouse and farmland
Education area	High school and college
Scenic spot	Parkland

closest CLC2012 was selected. Due to some land changes from 2012 to 2014, some areas may be in an overlapping position of the two lands. In order to determine the functional types of such lands, we combine Google Maps from February 2014 to collect PoIs in the vaguely delineated areas. To the end, the selected functional areas of the wireless dataset are shown in Table 2. Then, we list the functional area pairs formed by source and destination region and number them. Finally, we constructed the spatial encoding of functional area $\boldsymbol{R} = \{F_{11}, F_{12}, \ldots, F_{NN}\}$, where F_{ij} represents the number of functional area pairs from region i to region j. We also normalize $\boldsymbol{R}$ as we do for the backbone network dataset.

Network traffic series are highly correlated with time changes. Statistical models, such as ARIMA, can learn some regression features of previous time series. However, the features it learned do not correspond to time. For example, there is usually a peak in network traffic between 9 and 10 am, which corresponds to people's working hours. We add easily available timestamp information encoding as features to network TM so that the model can learn the influence of different times on traffic changes. In other words, we enhance the model's ability to perceive different timestamps rather than just learning the periodic changes of traffic as time steps move backward. One-hot is a commonly used encoding method for the discrete temporal feature. However, one-hot encoding is sparse. When considering multidimensional time (month, day, and week), concatenating one-hot encoding yields large dimensions. Using word embedding encoding ensures that the time encoding dimension of each level is

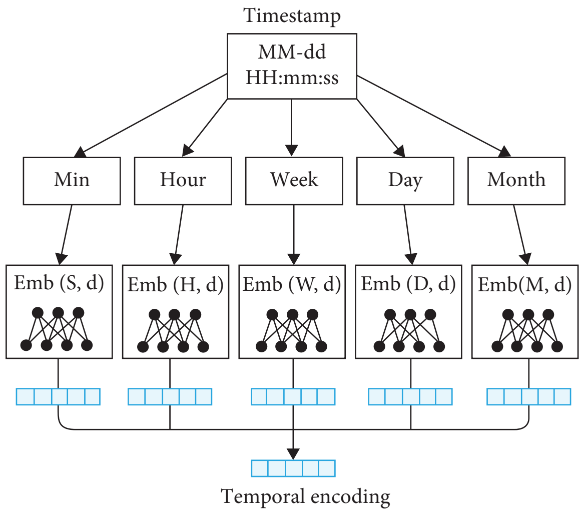


Figure 3: The encoding of timestamp.

the same as the input data. For example, one component $\boldsymbol{X}^t$ of the input sequence represents the network TM sampled in time t. It has d dimensions, which means there is d origin–destination traffic flow in the network. Assuming that its timestamp is $\boldsymbol{P}$, for example, "2021-10-22 15:25". The encoding of the timestamp is shown in Figure 3. We first decompose $\boldsymbol{P}$ in different granularity, for example, year, month, day, hour, minute, and the day of the week. Then the timestamps are input into different fully connected layers with different embedding dimensions. Then we map them into d-dimension embeddings. Finally, the time embeddings are added together as the final timestamp encoding. We design different granularity timestamp encodings to provide different span time markers for subsequent self-attention modules. We expect the self-attention module to learn different traffic patterns through these markers, such as weekly and daily patterns.

4.3. Learnable Positional Encoding. Figure 4 shows the training process of RNN and self-attention model, where $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_t$

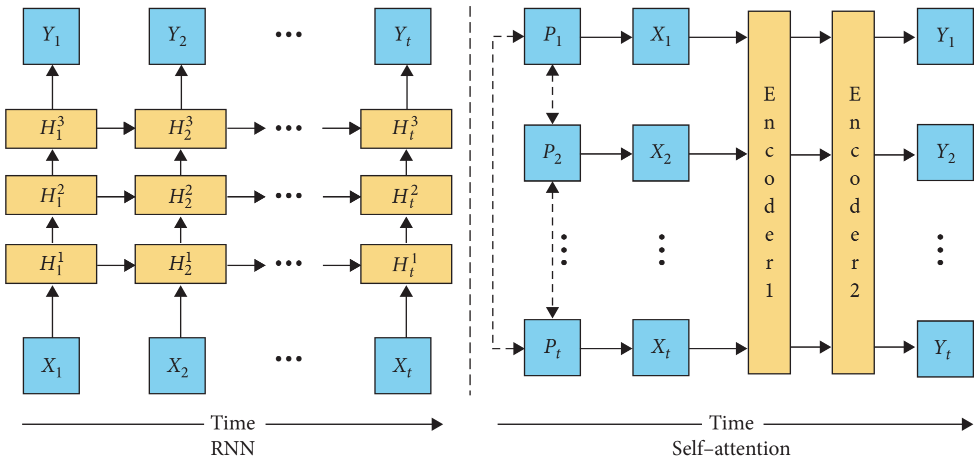


Figure 4: The training process of RNN and self-attention model.

is the input sequence, H_i^j is the j_{th} hidden layer of time step i, and $\boldsymbol{Y}_1$, $\boldsymbol{Y}_2$,…, $\boldsymbol{Y}_t$ is the output. In the training process of RNN, LSTM, and other recurrent neural networks, since the solution of the current system state requires the results of the previous time step, the input sequence will be fed into the network in time order. This kind of model can distinguish the before and after time relation of the input sequence. However, in the self-attention mechanism used in this paper, the encoder receives timing sequence data of different time steps simultaneously and calculates their similarity. It leads to the model's failure to capture the input sequence's time association.

Adding positional encoding (shown as P_i in Figure 4) to the encoder is an effective solution to the above problem, which provides the position relation of sequence for the encoder to ensure the normal training of the model. In addition, positional encoding could ensure the parallelization of training and speed up the training of the model. Vaswani et al. [19] use fixed positional encoding consisting of sine and cosine functions. The common data processing method is the sliding window method in the time-series prediction problem. If the data are not moved out of the window, then the same data remain in the following input. The difference is that it is shifted forward by one unit. Fixed position encoding is weak in learning relative position relationships, so we use another position encoding: learnable position encoding. We add different positional encodings for the input sequence in different encoders. First, we construct the tensor $\boldsymbol{L} = \{L_1, L_2, \ldots, L_d\}$, where d is the dimension of the input sequence. We compare several initialization methods, such as normal distribution initialization, xavier initialization, and uniform distribution initialization. Different initialize methods have little effect on the results. We chose the uniform distribution and the positional encodings are initialized within $[-1, 1]$. The learnable positional encoding is trained along with the whole structure during training. As shown in Figure 4, the added positional encoding plays a role in providing location relations. Multilayer encoders contain multiple levels of position encoding, and we expect to improve the model's ability to learn position relationships at different levels in this way.

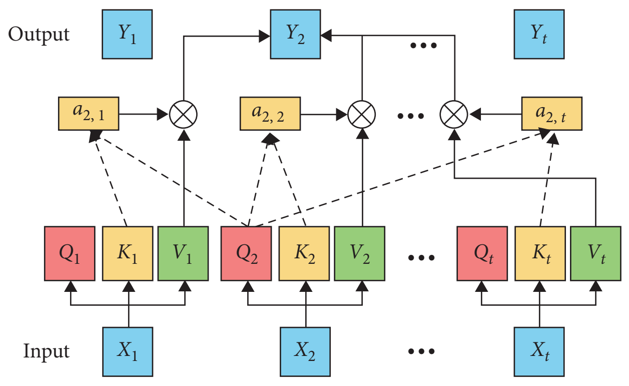


Figure 5: The calculation process of self-attention mechanism.

4.4. Self-Attention Module. According to the previous analysis, the temporal and spatial correlation between the input and target sequences is significant in network TM prediction. When the input sequence and the prediction sequence are long, it becomes difficult for the model to learn the dependencies of the sequence. The self-attention mechanism is widely used in NLP tasks. In translation tasks, the self-attention mechanism can calculate the similarity between other words and the current word in an input sentence, assigning different attention weights to different words. In this way, self-attention could learn the dependency relationship in the sentence. Furthermore, the self-attention mechanism does not use recurrent structures to capture features. It uses matrix multiplication to calculate the similarity of two words. This approach shortens the distance between two words to capture longer dependencies. Inspired by this idea, we transfer the self-attention mechanism to the network TM prediction problem. We treat the network TM at the moment as a word and use the self-attention mechanism to calculate the dependency of the long-term input sequence.

The self-attention mechanism essentially reflects how much each token pays attention to other tokens. Figure 5 shows the calculation process of the self-attention mechanism. Suppose $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_t$ is the input sequence encoded by space–time and position. $\boldsymbol{E}$ is formulated as the sum of all the encodings:

$$\boldsymbol{E} = \boldsymbol{R} + \boldsymbol{P} + \boldsymbol{L}, \tag{4}$$

where $\boldsymbol{R}$ is the spatial encoding, $\boldsymbol{P}$ is the timestamp encoding, and L is the positional encoding.

First, query, key, and value matrices are generated for each token through different fully connected layers, which are donated by $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$, respectively. In the case of $\boldsymbol{Q}$, the equation is as follows:

$$\boldsymbol{Q} = (\boldsymbol{X} + \boldsymbol{E})\boldsymbol{W}_Q, \tag{5}$$

where $\boldsymbol{W}_Q$ is the weight of the fully connected layer that calculates $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ are also calculated by the same formula, using weights $\boldsymbol{W}_K$ and $\boldsymbol{W}_V$, respectively.

Take $\boldsymbol{X}_2$ as an example, multiply $\boldsymbol{Q}_2$ by $\boldsymbol{K}_i$ of each time step to calculate the attention score, and then we can get the attention of $\boldsymbol{X}_2$ to the input of other time steps. The output of this layer can be expressed as:

$$\boldsymbol{Y}_i = \sum_j a_{i,j}\boldsymbol{V}_j, \tag{6}$$

where i and j are the serial numbers of tokens, and $a_{i,j}$ is the attention weight of the $\boldsymbol{X}_i$ to $\boldsymbol{X}_j$. That is, the output is the weighted sum of each value matrix. This method can effectively capture the dependencies in a long sequence. The specific calculation process of attention score is shown in Equation (7), where d is the dimension of network TM. The dot product of the matrices is used to calculate the attention score of the two matrices. After scaling, the attention score is multiplied by $\boldsymbol{V}$ as a weight to obtain the weighted matching result.

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}}\right)\boldsymbol{V}. \tag{7}$$

We can understand how the self-attention mechanism captures the spatiotemporal dependence by analyzing the computational process. Assuming that $\boldsymbol{W}_Q$ and $\boldsymbol{W}_K$ are the weights of the fully connected layer forming query matrix and key matrix, respectively, and $\boldsymbol{X}$ is the input sequence. According to Equations (5) and (7), the calculation process of attention with $\boldsymbol{A}_i$ and $\boldsymbol{A}_j$ is as follows:

$$\begin{aligned}\text{Attention}_{i,j} &= \boldsymbol{Q}_i\boldsymbol{K}_j^T \\ &= \underbrace{\boldsymbol{X}_i\boldsymbol{W}_Q\boldsymbol{W}_k^T\boldsymbol{X}_j^T}_{(a)} + \underbrace{\boldsymbol{X}_i\boldsymbol{W}_Q\boldsymbol{W}_k^T\boldsymbol{E}_j^T}_{(b)}, \\ &+ \underbrace{\boldsymbol{E}_i\boldsymbol{W}_q\boldsymbol{W}_k^T\boldsymbol{X}_j^T}_{(c)} + \underbrace{\boldsymbol{E}_i\boldsymbol{W}_Q\boldsymbol{W}_k^T\boldsymbol{E}_j}_{(d)}\end{aligned} \tag{8}$$

where (d) in this equation contains the encodings of two sequences: $\boldsymbol{E}_i$ and $\boldsymbol{E}_j$, which reflects the model's learning of the spatiotemporal dependence of the two sequences.

Multihead attention modules are combined with several self-attention modules. Assuming that attention modules use h heads, note that h can divide d, which is the dimensions of network TM embedding. The multihead attention module divides the d dimensions of $\boldsymbol{Q}$, $\boldsymbol{K}$ and $\boldsymbol{V}$ into h parts first, and uses the Equation (7) to calculate the attention score, respectively. Finally, the module contacts each result as its final output. Multihead attention can play a role in ensemble learning, prevent overfitting and help to extract multiple features.

In order to optimize the temporal and spatial complexity of the model, we use the self-attention mechanism with linear complexity proposed in [48]. The following equation shows its calculation process:

$$\boldsymbol{E}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \rho_{\text{row}}(\boldsymbol{Q})(\rho_{\text{col}}(\boldsymbol{K})^T\boldsymbol{V}), \tag{9}$$

where ρ_{row} and ρ_{row} denote applying the softmax function along each row or column of matrix, respectively. This mechanism has $O(dn + d^2)$ memory and $O(d^2n)$ computational complexities, where n is the input length, d is the dimension of $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$. d is usually much less than n in the long-term prediction, so the memory and computational complexity can be approximated as $O(n)$. Compared with the original attention whose memory and computational complexity is $O(n^2)$, the computational complexity is significantly reduced.

4.5. Feedforward and Output Layer. The subsequent structure in the encoder is the feedforward layer, which contains two fully connected layers. The activation function of the first layer is rectified linear unit (ReLU), which can provide nonlinear transformation. The feedforward layer can be described as the following equation:

$$Z(\boldsymbol{X}) = ReLU(\boldsymbol{X}\boldsymbol{W}_1 + b_1)\boldsymbol{W}_2 + b_2, \tag{10}$$

where $\boldsymbol{X}$ is the output of multihead attention module, and $\boldsymbol{W}$, b are the weight and bias of the fully connected layer, respectively.

In the language translation task, word2vec or other methods are needed to encode each word first. Since there is no complex embedding of traffic TM, we only use the encoder structure. We use the linear layer to produce the prediction results in the output module. ReLU is used as its activation function.

4.6. Training Algorithm. The proposed SATMP is trained by minimizing the mean square error (MSE) of the predicted value $\widehat{\boldsymbol{Y}}$ and the ground truth $\boldsymbol{Y}$. The loss function can be defined by:

$$\text{Loss} = \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{Y}_i - \widehat{\boldsymbol{Y}}_i\right)^2, \tag{11}$$

where N is the number of network nodes, and $\widehat{\boldsymbol{Y}}$, $\boldsymbol{Y}$ represents the predicted value and the ground truth, respectively.

The training process of SATMP is described in Algorithm 1.

```
Input: Historical network traffic $\boldsymbol{X}$ with a time span of $T$.
    Spatial encoding $\boldsymbol{R}$.
    Temporal encoding $\boldsymbol{P}$.
Output: SATMP model
1: $t = 1$
2: for $t = 1$ to $T - 2\omega + 1$ do
3:     $e = t + \omega - 1$
4:     obtain the input sequence $L = (\boldsymbol{X}^t, \boldsymbol{X}^{t+1}, \cdots, \boldsymbol{X}^e)$
5:     obtain the target sequence $Y = (\boldsymbol{X}^{e+1}, \boldsymbol{X}^{e+2}, \cdots, \boldsymbol{X}^{e+w})$.
6:     build the training instance $(\{\boldsymbol{L}, \boldsymbol{R}, \boldsymbol{P}\}, Y)$ by Equations (4).
7: end for
8: Initialize the trainable parameters in SATMP
9: Update the parameters in SATMP using backpropagation
   algorithm with loss function Loss as Equation (11).
```

Algorithm 1: SATMP Training Algorithm.

5. Evaluation

5.1. Datasets and Preprocessing. We use three publicly available and well-known traffic datasets to evaluate the performance of the proposed algorithm. These three datasets are summarized in Table 3. Abilene dataset [49] is sampled from the Abilene Network and consists of 12 nodes and 15 links. The network traffic matrix formed by it has 144 dimensions. Abilene records the network TM from March 1th, 2004 to September 10th, 2004 at 5 min sampling intervals. Geant carries research traffic from the European National Research and Education Networks (NRENs) [50]. It contains more network nodes and links and has 23 nodes linked by 38 links. The TM of Geant has 529 dimensions. The data are taken in 15 min steps starting on May 4th 2005 at 15:00 and ending on August 31st 2005. Due to the discontinuity of the dataset sampling, we select continuous data for the experiment instead of using all the data. Specifically, we use Abilene data from May 1st 00:00 to May 28th 23:55, which contains 8,064 TM. For Geant, we select data from June 1st 00:00 to June 28th 16:30, which contain 2658 TM. The MItoMI telecommunications dataset provides the directional interaction strengths between different areas of Milan from November 1st 2013 to January 1st 2014 [51]. The dataset divides Milan into 100×100 grids and records the interaction at a sampling interval of 10 min. Since the communication intensity of most regions is 0 most of the time, we selected 20 active regions for study. We get a total of 8,640 TM with 400 dimensions.

The way we preprocess the data is as follows. First, we perform a unit conversion on the dataset. The original values are converted into $MBit/s$ values on backbone network datasets. Then we normalize the TM by dividing its maximum value. We also convert the timestamp to the "Europe/Rome" timezone on the MItoMI dataset. Finally, we use a sliding window to build the training dataset, as is shown in Figure 6. Assuming that the dataset has a total of T TM, the sliding window size is ω, and the predicted window size is l. The values of ω and l are determined by actual needs, and the relationship between them in our study is $\omega = l$. Then we roll the sliding window with stride = 1. According to the above assumptions, the length of the dataset generated by our preprocess is $T - \omega - l + 1$. We divide it into the training set, validation set, and testing set according to the ratio of 6 : 1 : 3.

Table 3: Summary of used datasets.

Attribute	Abilene	Geant	MItoMI
Network type	Backbone	Backbone	Wireless
Number of nodes/regions	12	23	10,000
Number of links	15	36	-
Sampling interval/min	5	15	10
Time span/mon	6	4	2

5.2. Experimental Details. We use Pytorch to build our model. The loss function we choose is MSELoss. The AdamW algorithm with decay learning rate is utilized to optimize the model. In addition, we use dropout and gradient clipping to avoid overfitting. The hyperparameter settings of the proposed model are shown in Table 4.

Prophet [25], LSTM [52], and GRU [53] are selected to compare with the proposed framework. Prophet is an additive model that can effectively capture the periodicity of time series. LSTM model effectively alleviates the problems of RNN gradient disappearance and explosion by adding gated structure and cell state. GRU simplifies the structure of LSTM and achieves better results on some tasks.

We use the above models to train and test their performance in predicting flow sequences of different lengths. The predicted lengths of time we choose are 24, 48, 72, and 96 hr. For the methods used for comparison, we tune their hyperparameters with a validation set for better results.

We evaluate the performance of network TM prediction based on two metrics: MSE (Equation (11)) and mean absolute error (MAE), which are defined as:

$$\mathrm{MAE} = \frac{1}{N} \sum_{i=1}^{N} \left| \boldsymbol{Y}_i - \widehat{\boldsymbol{Y}}_i \right|, \qquad (12)$$

where N is the number of network nodes, and $\widehat{\boldsymbol{Y}}$, $\boldsymbol{Y}$ represents the predicted value and the ground truth respectively.

5.3. Results and Analysis. The prediction performance of all methods is summarized in Table 5. The best result for each prediction task is highlighted in bold in the table. Note that the dataset changes due to the different predicted lengths, so we only make horizontal comparisons and do not explore the performance changes of each model as the sequence length increases. Our proposed SATMP significantly reduces the prediction error. Take the traffic prediction for 72 hr as an example, the MSE of SATMP is 68%, 48%, and 50% lower than that of Prophet, LSTM, and GRU on the Abilene. On MItoMI, the MSE of SATMP is 61%, 27%, and 24% lower than that of methods for comparison.

According to the experimental results, the MSE and MAE of Prophet are very high. Prophet can model the periodic features of historical data. However, it cannot use additional information to capture the spatiotemporal association of

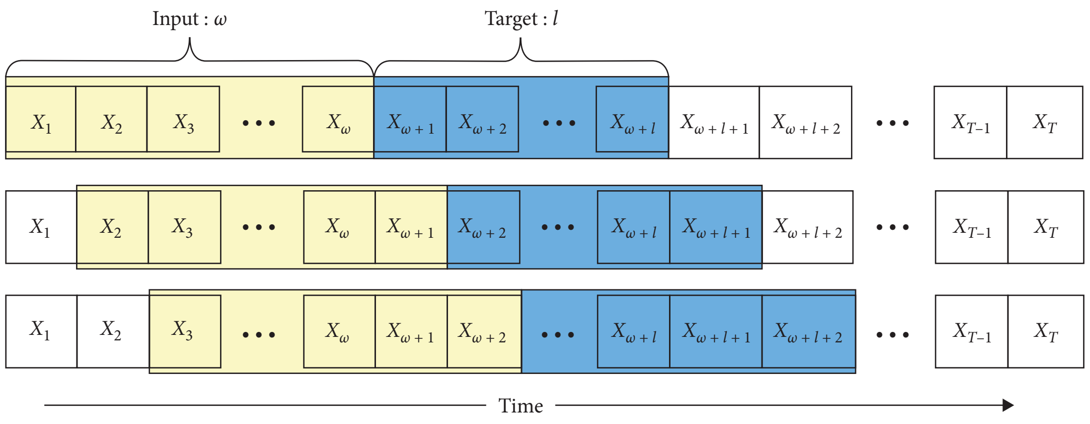


Figure 6: Sliding window for building training dataset.

Table 4: Hyperparameter settings.

Hyperparameter	Abilene	Geant	MItoMI
Batch size	32	32	32
Dropout	0.1	0.1	0.1
Encoder layer	8	6	8
Attention head	12	23	10
Attention dimension	2,048	2,048	2,048
Epoch	150	150	150
Learning rate	0.001	0.001	0.0001
Optimizer	AdamW	AdamW	AdamW
Parameter clip	True	True	True

Table 5: Prediction performance of all methods on three datasets.

Methods		SATMP		Prophet		LSTM		GRU	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Abilene	24	**0.0114**	**0.0692**	0.0284	0.1125	**0.0114**	0.0724	0.0146	0.0809
	48	**0.0118**	**0.0705**	0.0279	0.1113	0.0271	0.1089	0.0235	0.1063
	72	**0.0095**	**0.0609**	0.0295	0.1062	0.0181	0.0872	0.0189	0.0963
	96	**0.0081**	**0.0538**	0.0329	0.1133	0.0165	0.0799	0.0165	0.0846
Geant	24	**0.0033**	**0.0228**	0.0046	0.0384	0.0041	0.0241	0.0042	0.0242
	48	**0.0028**	**0.0222**	0.0042	0.0367	0.0034	**0.0218**	0.0039	0.0260
	72	**0.0025**	**0.0201**	0.0056	0.0445	0.0032	0.0212	0.0043	0.0302
	96	**0.0023**	**0.0204**	0.0045	0.0402	0.0031	0.0226	0.0033	0.0238
MItoMI	24	**0.0067**	**0.0364**	0.0298	0.0577	0.0094	0.0496	0.0089	0.0492
	48	**0.0070**	**0.0352**	0.0224	0.0501	0.0097	0.0509	0.0094	0.0512
	72	**0.0069**	**0.0354**	0.0177	0.0474	0.0094	0.0495	0.0091	0.0498
	96	**0.0066**	**0.0330**	0.0222	0.0510	0.0089	0.0477	0.0090	0.0489

The values in bold are the values with the lowest error (MSE, MAE, respectively) among the four methods.

network TM, which leads to a high error. LSTM and GRU have similar prediction performance, which proves the effectiveness of their time-series modeling. Obviously, SATMP significantly reduces MSE and MAE. By utilizing spatiotemporal encodings and self-attention mechanisms, SATMP can better capture the potential features of the network TM.

Figure 7 shows the 48 hr prediction result of SATMP, Prophet, LSTM, and GRU on the Abilene dataset. The blue curves represent the real flow, while the curves in other colors represent the predicted results of other methods. It is evident that Prophet failed in long-term prediction. It can only predict a general trend. All the predicted results are basically the same and deviate greatly from the real value. SATMP, LSTM, and GRU have similar prediction performance before the second 18:00. However, after the second 18:00, the results of LSTM and GRU begin to show large errors, which indicates that the accuracy of these two models will decline when the prediction time is very long. The results

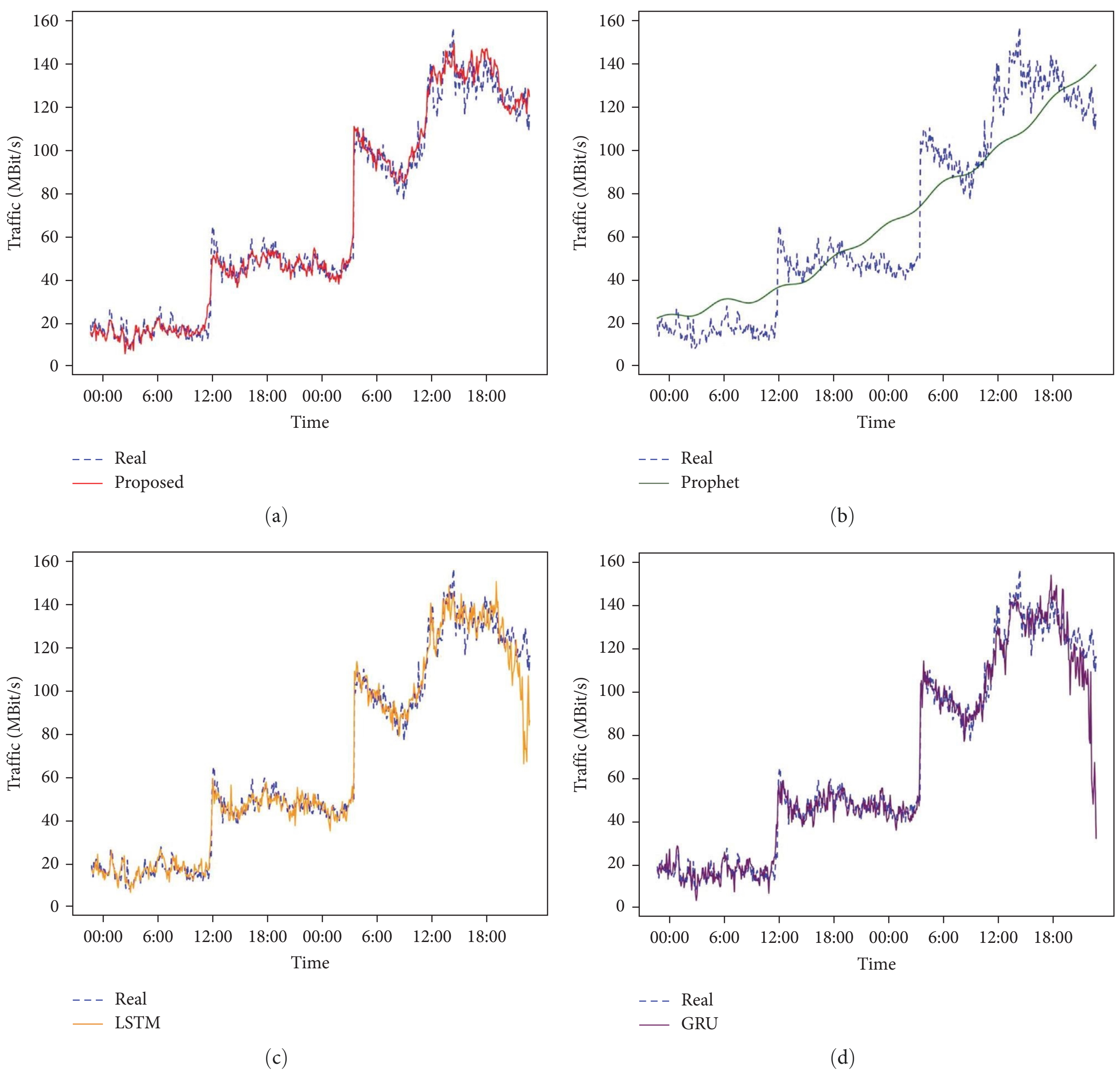


Figure 7: The prediction (48 hr) of SATMP, Prophet, LSTM and GRU on the Abilene dataset.

demonstrate that SATMP has a more potent ability to extract long-term correlation. Spatiotemporal encoding can effectively provide spatiotemporal information to the self-attention module and enhance the prediction performance.

Figure 8 shows the prediction of SATMP on Abilene (48 hr with 576 time steps). It can be found that the traffic of the backbone network conforms to certain rules, but there is a lot of fluctuation. The prediction result confirms that SATMP can better predict the fluctuation of irregular traffic. It can cope with high burst traffic scenarios as shown in Figures 8(b) and 8(c). The proposed algorithm makes full use of the spatiotemporal characteristics and uses the self-attention mechanism to capture its long-term spatiotemporal dependence, so as to learn the properties of network traffic more comprehensively. The aforementioned result shows that SATMP has the advantages of high accuracy and long-term prediction in complex networks.

In addition, we compare the hardware requirements of LSTM with the proposed algorithm. The sampling interval is 5 min on the Abilene dataset. 24 hr traffic data contains 288 network TM, and 96 hr traffic data contains 1,152 network TM. Such a long sequence imposes massive memory consumption of graphics cards on LSTM. Take the prediction of 96 hr as an example, LSTM occupies 14 GB of video memory on average, sometimes more than 24 GB. The average occupied memory of SATMP is 9 GB, saving a lot of computation power. Besides, SATMP eliminates the recurrent structure and can compute all inputs in parallel. With the same number of parameters, it has a faster training speed. As discussed above, the model proposed in this

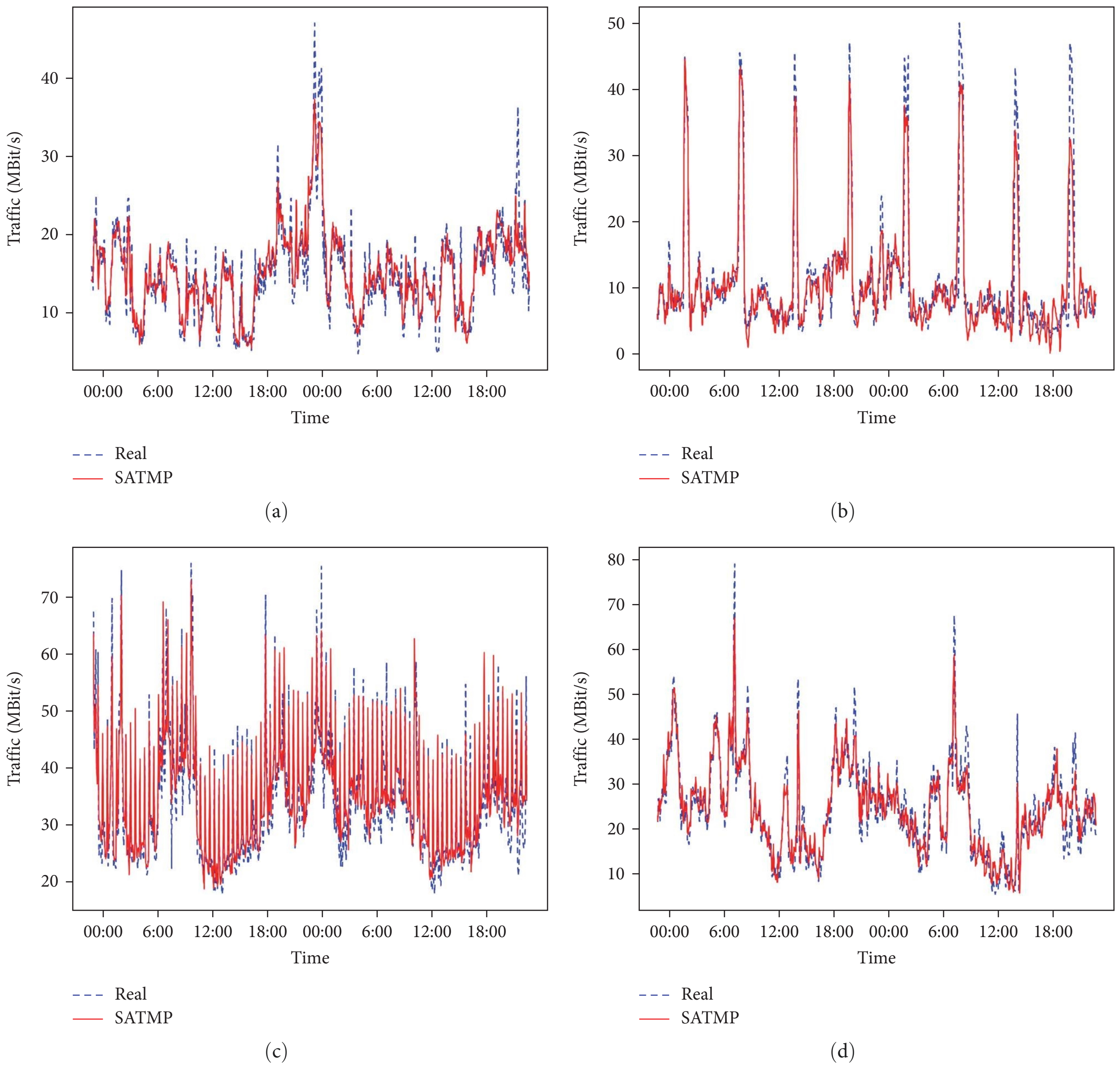


Figure 8: The prediction (48 hr) of SATMP on Abilene.

paper has a great advantage in consuming computational power.

In the highly dynamic environment of IIoT, the network changes over time. For example, the addition and departure of nodes will affect the traffic of the entire network. Therefore, our algorithm needs to be retrained after the network changes greatly. Fortunately, the training process of our algorithm is parallel, which has higher training efficiency and can realize model iteration faster.

5.4. Ablation Study. To test the effectiveness of each module in SATMP, an ablation experiment is designed in this paper. We design two ablation models and test them on Abilene. Model 1 deletes the spatiotemporal feature extraction module of proposed framework. Model 2 replaces the learnable positional encoding with fixed position code. At the same time, if the removed modules involve neural networks, we use the fully connected layer instead to ensure that the number of parameters is roughly unchanged.

Figure 9 shows the results of ablation study. The result confirms that the spatiotemporal feature extraction module provides more additional features to the model, which can significantly reduce the prediction error. Compared with fixed positional encoding, learnable positional encoding has better ability to describe the temporal correlation of traffic sequence, which is helpful to predict network traffic matrix more accurately.

6. Conclusion

This paper investigates the problem of long-term prediction of network TM in large-scale IIoT networks. The 6G-enabled IIoT will contain many heterogeneous networks, making traffic

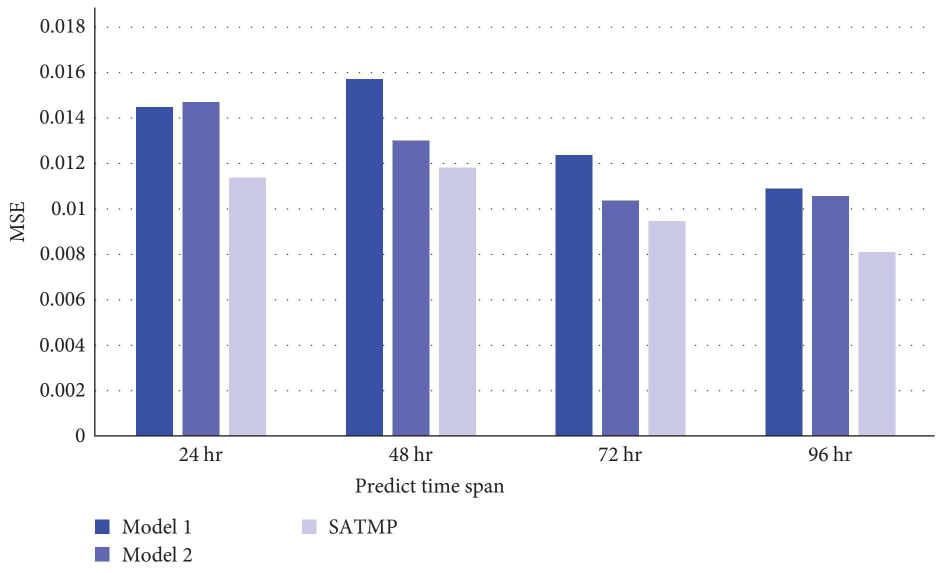


FIGURE 9: Ablation study of SATMP.

prediction difficult. We provide a novel method by applying the self-attention mechanism to resolve this issue. The self-attention mechanism can reduce the distance of time-series dependence. Inspired by that, we apply the mechanism to the long-term prediction of IIoT network TM and propose SATMP, a self-attention prediction model combining spatio-temporal encoding. We show the effectiveness of SATMP for long-term TM prediction with a detailed analysis and evaluation on three backbone and wireless network datasets. SATMP's accurate long-term prediction results enable IIoT networks to implement effective resource allocation, congestion control, and attack detection. In addition, SATMP supports parallel computing and can be deployed on edge IIoT nodes for edge intelligence.

The data in 6G-enabled IIoT need to be computed securely and quickly. Several learning frameworks have been proposed to address this characteristic. For instance, federal learning is considered a key technology for the future of IIoT, which supports the collaborative training of nodes while protecting data privacy. We plan to combine self-attention mechanisms with Federal learning for further IIoT network research in future work.

Data Availability

The datasets used in this study can be downloaded from https://www.cs.utexas.edu/~yzhang/rese-arch/AbileneTM, https://totem.info.ucl.ac.be/dataset.html, and https://dataverse.harvard.edu/da-taverse/bigdatachallenge.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Science Foundation of China (no. 61772385).

References

[1] D. C. Nguyen, M. Ding, P. N. Pathirana et al., "6G internet of things: a comprehensive survey," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 359–383, 2022.

[2] Z. Ning, Y. Yang, X. Wang et al., "Dynamic computation offloading and server deployment for UAV-enabled multi-access edge computing," *IEEE Transactions on Mobile Computing*, vol. 22, no. 5, pp. 2628–2644, 2023.

[3] Y. Gong, H. Yao, J. Wang, M. Li, and S. Guo, "Edge intelligence-driven joint offloading and resource allocation for future 6G industrial internet of things," *IEEE Transactions on Network Science and Engineering*, 2022.

[4] X. Kong, S. Tong, H. Gao et al., "Mobile edge cooperation optimization for wearable internet of things: a network representation-based framework," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 5050–5058, 2021.

[5] Z. Ning, S. Sun, X. Wang et al., "Intelligent resource allocation in mobile blockchain for privacy and security transactions: a deep reinforcement learning based approach," *Science China Information Sciences*, vol. 64, Article ID 162303, 2021.

[6] X. Kong, K. Wang, M. Hou et al., "A federated learning-based license plate recognition scheme for 5G-enabled internet of vehicles," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 12, pp. 8523–8530, 2021.

[7] A. Mukherjee, P. Goswami, M. A. Khan, L. Manman, L. Yang, and P. Pillai, "Energy-efficient resource allocation strategy in massive IoT for industrial 6G applications," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5194–5201, 2021.

[8] Z. Ning, P. Dong, M. Wen et al., "5G-enabled UAV-to-community offloading: joint trajectory design and task scheduling," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 11, pp. 3306–3320, 2021.

[9] B. Barakat, A. Taha, R. Samson et al., "6G opportunities arising from internet of things use cases: a review paper," *Future Internet*, vol. 13, no. 6, Article ID 159, 2021.

[10] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman, and D. O. Wu, "Edge computing in industrial internet of things: architecture, advances and challenges," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2462–2488, 2020.

[11] W. Z. Khan, M. H. Rehman, H. M. Zangoti, M. K. Afzal, N. Armi, and K. Salah, "Industrial internet of things: recent advances, enabling technologies and open challenges," *Computers & Electrical Engineering*, vol. 81, Article ID 106522, 2020.

[12] Y. Wang, R. Forbes, U. Elzur et al., "From design to practice: ETSI ENI reference architecture and instantiation for network management and orchestration using artificial intelligence," *IEEE Communications Standards Magazine*, vol. 4, no. 3, pp. 38–45, 2020.

[13] C. Zhang and P. Patras, "Long-term mobile traffic forecasting using deep spatio-temporal neural networks," in *Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 231–240, Association for Computing Machinery, New York, NY, USA, 2018.

[14] X. Wang, Z. Ning, S. Guo, M. Wen, L. Guo, and H. V. Poor, "Dynamic UAV deployment for differentiated services: a multi-agent imitation learning based approach," *IEEE Transactions on Mobile Computing*, vol. 22, no. 4, pp. 2131–2146, 2023.

[15] X. Kong, K. Wang, S. Wang et al., "Real-time mask identification for COVID-19: an edge-computing-based deep learning framework," *IEEE Internet of Things Journal*, vol. 8, no. 21, pp. 15929–15938, 2021.

[16] M. Alasmar, R. Clegg, N. Zakhleniuk, and G. Parisis, "Internet traffic volumes are not gaussian—they are log-normal: an 18-year longitudinal study with implications for modelling and prediction," *IEEE/ACM Transactions on Networking*, vol. 29, no. 3, pp. 1266–1279, 2021.

[17] Z. Lin, M. Feng, C. N. dos Santos et al., "A structured self-attentive sentence embedding," 2017.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," 2019.

[19] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," 2017.

[20] H. Z. Moayedi and M. Masnadi-Shirazi, "Arima model for network traffic prediction and anomaly detection," in *2008 International Symposium on Information Technology*, pp. 1–6, IEEE, Kuala Lumpur, Malaysia, 2008.

[21] Y. Wang, D. Jiang, L. Huo, and Y. Zhao, "A new traffic prediction algorithm to software defined networking," *Mobile Networks and Applications*, vol. 26, pp. 716–725, 2021.

[22] M. Kim, "Network traffic prediction based on INGARCH model," *Wireless Networks*, vol. 26, pp. 6189–6202, 2020.

[23] A. Bayati, V. Asghari, K. Nguyen, and M. Cheriet, "Gaussian process regression based traffic modeling and prediction in high-speed networks," in *2016 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7, IEEE, Washington, DC, USA, 2016.

[24] A. Y. Nikravesh, S. A. Ajila, C.-H. Lung, and W. Ding, "Mobile network traffic prediction using MLP, MLPWD, and SVM," in *2016 IEEE International Congress on Big Data (BigData Congress)*, pp. 402–409, IEEE, 2016.

[25] G. Jain and R. R. Prasad, "Machine learning, prophet and XGBoost algorithm: analysis of traffic forecasting in telecom networks with time series data," in *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pp. 893–897, IEEE, 2020.

[26] W.-C. Chien and Y.-M. Huang, "A lightweight model with spatial–temporal correlation for cellular traffic prediction in internet of things," *The Journal of Supercomputing*, vol. 77, pp. 10023–10039, 2021.

[27] L. Nie, D. Jiang, S. Yu, and H. Song, "Network traffic prediction based on deep belief network in wireless mesh backbone networks," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–5, IEEE, March 2017.

[28] M. Li, Y. Wang, Z. Wang, and H. Zheng, "A deep learning method based on an attention mechanism for wireless network traffic prediction," *Ad Hoc Networks*, vol. 107, Article ID 102258, 2020.

[29] S. Nihale, S. Sharma, L. Parashar, and U. Singh, "Network traffic prediction using long short-term memory," in *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, pp. 338–343, IEEE, Coimbatore, India, 2020.

[30] Q. He, A. Moayyedi, G. Dán, G. P. Koudouridis, and P. Tengkvist, "A meta-learning scheme for adaptive short-term network traffic prediction," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 10, pp. 2271–2283, 2020.

[31] L. Nie, X. Wang, S. Wang et al., "Network traffic prediction in industrial internet of things backbone networks: a multitask learning mechanism," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 10, pp. 7123–7132, 2021.

[32] L. Nie, Z. Ning, M. S. Obaidat et al., "A reinforcement learning-based network traffic prediction mechanism in intelligent internet of things," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 3, pp. 2169–2180, 2021.

[33] S. Troia, R. Alvizu, Y. Zhou, G. Maier, and A. Pattavina, "Deep learning-based traffic prediction for network optimization," in *2018 20th International Conference on Transparent Optical Networks (ICTON)*, pp. 1–4, IEEE, July 2018.

[34] N. Li, L. Hu, Z.-L. Deng, T. Su, and J.-W. Liu, "Research on GRU neural network satellite traffic prediction based on transfer learning," *Wireless Personal Communications*, vol. 118, pp. 815–827, 2021.

[35] Y. Li, H. Liu, W. Yang, D. Hu, and W. Xu, "Inter-data-center network traffic prediction with elephant flows," in *NOMS 2016–2016 IEEE/IFIP Network Operations and Management Symposium*, pp. 206–213, IEEE, Istanbul, Turkey, 2016.

[36] Z. Tian, "Network traffic prediction method based on wavelet transform and multiple models fusion," *International Journal of Communication Systems*, vol. 33, no. 11, Article ID e4415, 2020.

[37] L. Zhang, H. Zhang, Q. Tang et al., "LNTP: an end-to-end online prediction model for network traffic," *IEEE Network*, vol. 35, no. 1, pp. 226–233, 2021.

[38] W. Zhao, H. Yang, J. Li, L. Shang, L. Hu, and Q. Fu, "Network traffic prediction in network security based on EMD and LSTM," in *Proceedings of the 9th International Conference on Computer Engineering and Networks*, Q. Liu, X. Liu, L. Li, H. Zhou, and H.-H. Zhao, Eds., vol. 1143 of *Advances in Intelligent Systems and Computing*, pp. 509–518, Springer, Singapore, 2021.

[39] M. Tian, C. Sun, and S. Wu, "An EMD and ARMA-based network traffic prediction approach in SDN-based internet of vehicles," *Wireless Networks*, 2021.

[40] J. Feng, X. Chen, R. Gao, M. Zeng, and Y. Li, "DeepTP: an end-to-end neural network for mobile cellular traffic prediction," *IEEE Network*, vol. 32, no. 6, pp. 108–115, 2018.

[41] N. Zhao, Z. Ye, Y. Pei, Y.-C. Liang, and D. Niyato, "Spatial-temporal attention-convolution network for citywide cellular traffic prediction," *IEEE Communications Letters*, vol. 24, no. 11, pp. 2532–2536, 2020.

[42] A. Azzouni and G. Pujolle, "NeuTM: a neural network-based framework for traffic matrix prediction in SDN," in *NOMS 2018—2018 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–5, IEEE, April 2018.

[43] Z. Ning, P. Dong, X. Wang et al., "Distributed and dynamic service placement in pervasive edge computing networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 6, pp. 1277–1292, 2021.

[44] D. Andreoletti, S. Troia, F. Musumeci, S. Giordano, G. Maier, and M. Tornatore, "Network traffic prediction based on diffusion convolutional recurrent neural networks," in *IEEE INFOCOM 2019—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 246–251, IEEE, April 2019.

[45] H. Zhou, S. Zhang, J. Peng et al., "Informer: beyond efficient transformer for long sequence time-series forecasting," 2021.

[46] L. N. N. Do, H. L. Vu, B. Q. Vo, Z. Liu, and D. Phung, "An effective spatial-temporal attention based neural network for traffic flow prediction," *Transportation Research Part C: Emerging Technologies*, vol. 108, pp. 12–28, 2019.

[47] G. Büttner, "CORINE land cover and land cover change products," in *Land Use and Land Cover Mapping in Europe*, I. Manakos and M. Braun, Eds., pp. 55–74, Springer, Dordrecht, 2014.

[48] Z. Shen, M. Zhang, H. Zhao, S. Yi, and H. Li, "Efficient attention: attention with linear complexities," 2020.

[49] Y. Zhang, "Abilene dataset," 2004, https://www.cs.utexas.edu/~yzhang/research/AbileneTM/.

[50] S. Uhlig, B. Quoitin, J. Lepropre, and S. Balon, "Providing public intradomain traffic matrices to the research community," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 1, pp. 83–86, 2006.

[51] G. Barlacchi, M. De Nadai, R. Larcher et al., "A multi-source dataset of urban life in the city of Milan and the Province of Trentino," *Scientific Data*, vol. 2, Article ID 150055, 2015.

[52] A. Azzouni and G. Pujolle, "A long short-term memory recurrent neural network framework for network traffic matrix prediction," 2017.

[53] R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Applying deep learning approaches for network traffic prediction," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 2353–2358, IEEE, September 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 9391939, 13 pages
https://doi.org/10.1155/2022/9391939

Research Article

Intelligent Dynamic Spectrum Allocation in MEC-Enabled Cognitive Networks: A Multiagent Reinforcement Learning Approach

Chan Lei, Haitao Zhao, Li Zhou, Jiao Zhang, Haijun Wang, and Haitao Chen

College of Electronic Science and Technology, National University of Defense Technology, Changsha 410073, China

Correspondence should be addressed to Haitao Zhao; haitaozhao@nudt.edu.cn

Received 22 June 2022; Revised 17 August 2022; Accepted 25 August 2022; Published 25 September 2022

Academic Editor: Amr Tolba

Making effective use of scarce spectrum resources, along with efficient computational performance, is one of the key challenges for future wireless networks. To tackle this issue, in this paper, we focus on the intelligent dynamic spectrum allocation (DSA) in a mobile edge computing (MEC) enabled cognitive network. And our objective is to optimize the spectrum utilization and load balance among idle channels. Since users can only acquire part of environment information in a decentralized way, we model such a problem as decentralized partially observed Markov decision process (Dec-POMDP) and design the corresponding evaluating metric to encourage users sense and access spectrum properly. Then, we propose a QMIX-based DSA method with centralized training decentralized execution (CTDE) structure to tackle it. In the training phase, the users offload the computational tasks to the MEC server to obtain the optimal distributed DSA strategies, through which the users select the optimal channel locally in the execution phase. Simulation results show that, using the proposed algorithm, users can independently capture spectrum holes, and hence improve the spectrum utilization while balancing the load on available channels.

1. Introduction

With the rapid development of the future beyond 5G/6G wireless communication, more and more emerging applications like virtual reality (VR), augmented reality (AR), and interactive game are springing up, requiring low-latency, high-reliability and powerful computational capability [1]. Due to size, weight, and power (SWaP) constraints, devices are recognized to have limited computational capability to fully support these applications. Mobile edge computing (MEC) has a great potential to overcome this issue, through which the users can offload the computational tasks to MEC servers [2–4]. By this way, many services can be provided, like communication, caching, and computing [5]. Meanwhile, it can also potentially boost intelligence for users to achieve efficient spectrum access with low coordination overhead. Confronting the time-varying and complex wireless environment, it is challenging for users to access the spectrum adaptively with robustness guarantee. The MEC-enabled intelligent dynamic spectrum allocation (DSA) can play a significant role in this case.

Many efforts have been devoted on the DSA in the traditional way, e.g., blind rendezvous in D2D [6] and cognitive radio networks (CRNs) [7], cross-layer perspective [8, 9] and randomized rounding algorithm [10] in CRNs, and bipartite graph theory in wireless LANs [11]. Since these works require lots of statistical knowledge, which is difficult to obtain in a dynamic network, the intelligent approaches have been adopted for DSA further. In [12], genetic algorithm (GA) is adopted by the central node to complete DSA for each secondary user (SU) in a CRN. Benefit from the fitting properties of deep neural networks and the interaction with the environment, several works focus on the deep reinforcement learning (DRL) structure [13]. Specially, in [14, 15], the central controller is employed to evaluate and allocate channels to multiuser through DRL, while the users report the channel state after having an access. And the maximum channel utilization and minimum collision are expected to be optimized with no prior information for multiuser. However, there is still much room for improvement compared with the optimal scheme, due to the heavy signalling overhead and overdependence on the central node. To tackle this problem, there are increasing works that focus on intelligent DSA in a distributed way. In [16], the bioinspired solution is employed for users to adjust and optimize the cluster size distributedly in cognitive internet of things (IoT), which aims to achieve efficient spectrum allocation, flexible connection, and minimum network access delay. In [17], a heuristic method is applied for users to sense an unoccupied spectrum and build an optimum route, so as to complete dynamic spectrum allocation and power control. The authors in [18] have modeled the multiuser multichannel allocation as an undirected graph, and a greedy algorithm is designed to form the load balanced cluster. The authors in [19, 20] introduce the game theory for SUs to allocate spectrum, and the SUs learn access strategies competitively by maximizing their respective revenue. The authors in [21] have investigated the problem of multiuser enabled frequency division multiple access in radio network, and a bargaining approach is used to allocate subcarrier and power for multiuser by get a Nash equilibrium, so as to achieve the tradeoff between throughput and power consumption. Even though these works make a good breakthrough in the intelligent DSA, there are still some shortcomings. For one thing, these methods are just feasible for the simple case with small strategy space, otherwise, the long decision time would occur. Obviously, it is impractical for the low-latency scenarios. For another, the interaction should be addressed in the dynamic environment furtherly.

To tackle the dilemmas above, a powerful method, mutliagent reinforcement learning (MARL), has gained increasing interest recently. MARL is an extension of reinforcement learning (RL) [22], which is suitable for distributed learning and processing. With the aid of this method, users can interact with environment and obtain their DSA strategies as agents. Particularly, in [23, 24], the double deep Q-network (DQN) algorithm and multiagent Q-learning are applied to active users, who compete to access multichannel independently to achieve minimum collisions. In [25], the SUs in CRN are aiming to learn proper spectrum access strategies autonomously and the DRL method, along with echo state network (ESN) is adopted, through which interference can be alleviated. And in [26], a competitive spectrum access scheme for multiuser is proposed, and the performance of DSA and avoidance is simply analyzed by Q-learning method. While in [27], multiuser DSA is modeled as a multiarmed bandit problem, in which the users are supposed to access proper spectrum distributedly. However, since there is no negotiation that users analyze information locally and allocate the spectrum resources by way of competition in these works, the spectrum access can be regarded as an ALOHA-like process, which is still confronted with great challenges in highly dynamic environments. Moreover, for constraints of computing and battery capacity, there is still a huge gap to fill before deployment in real world.

Inspired by MEC technology, which could extend the computational capacity and process the task for DSA at edge cloud platform [28–30], the centralized training decentralized execution (CTDE) scheme of MARL is addressed to achieve efficient DSA [31–33]. By this way, the distributed users offload the DSA task to the cloud platform in the training phase, so as to learn the action-taking strategies, and then work in a fully distributed manner in the practical implementation phase. Specially, the authors in [31] have investigated a noncooperative DSA in CRN. In [32], users are expected to transmit on idle channel and cooperate to maximize the sum rate, and the double DQN algorithm is utilized, resulting in a certain gap with the optimal performance. Considering the linear relationship between the global and individual utility, the authors in [33] propose a QMIX-based DSA algorithm, which brings the excellent strategies for users, and the maximum successful transmission and minimum collision are realized. However, in order to avoid collision, the users in [32, 33] are allowed to be silent. That is, it is assumed that only part of users could participate in the DSA at the same time, which is unfair to all users distributed in the network. Moreover, the works in [18, 24, 27] take the perfect spectrum detection into consideration, where the selection for idle spectrum is always guaranteed. In reality, influenced by the dynamic environment and limited hardware condition, the detection ability is usually partial [34] and imperfect [35].

In this paper, we investigate the intelligent DSA for multiuser MEC networks. We consider that the users can only sense part of information, and the ability of detection is assumed to be imperfect. Motivated by the monotonicity of the considered problem, as well as the powerful computation ability, the QMIX-based DSA algorithm with CTDE structure is employed. To the best of our knowledge, this work has not been researched yet. And we highlight the main contributions of our work as follows:

(1) We focus on a MEC-enabled cognitive network deploying multiple SUs who attempt to access the dynamic spectrum without perfect sensing capabilities. In this scenario, SUs are supposed to be intelligent to achieve their common task autonomously. Meanwhile, in order to make up for the users' limited computational capability and energy reserve, the MEC server, which is computationally powerful and long-lived, is employed at BS. This is practically significant, since the traditional dependence on a central controller is released. And the users can adapt to the environment independently and timely with lower overhead, so that it can be further extended to the latency-sensitive applications

(2) We formulate a distributed DSA problem to improve both the idle channel utilization and load balance. And the problem is modeled as a decentralized partial observation Markov decision process (Dec-POMDP). Then we propose a CTDE enabled DSA algorithm, whose characteristic is consistent with that of the modeled problem. This algorithm can handle the environment dynamics and users' partial observation with low-complexity for practical implementation. Specially, different from these online searching methods, such as POMCP [36], DESPOT [37], and HyP-DESPOT [38], there are two phases

in our proposed algorithm, i.e., offline training and online execution. In the offline phase, the task of DSA is offloaded to the MEC. With the aid of MEC, the SUs adaptively adjust their DSA strategies, so that their network models can be well trained finally. While in the online phase, each SU executes action locally based on the trained model with no central controller and the coordination among SUs

(3) We present simulations to demonstrate the effectiveness and feasibility of our proposed DSA algorithm in the different settings under dynamic environment. We observed that the optimal network utility is always realized after limited training, while the sensing accuracy is also improved. The SUs can effectively overcome their imperfect sensing characteristic and capture the idle channels. Based on this, the expected optimal DSA task could be completed in a fully distributed manner

The rest of this paper is structured as follows: the system model is provided in Section 2. In Section 3, the problem is formulated as a Dec-POMDP to maximize the global utility. Then in Section 4, QMIX-based algorithm is proposed to obtain the optimal DSA policy. Numerical results are provided in Section 5, and the conclusion of the paper is presented in Section 6.

2. System Model

As shown in Figure 1, we consider a MEC-enabled cognitive network consisting of one primary user (PU), N SUs, with the set denote by $\mathrm{SN} = \{\mathrm{SU}_1, \mathrm{SU}_2, \cdots, \mathrm{SU}_N\}$, and one cognitive BS with MEC server. There are M orthogonal authorized channels, denoted as $CH = \{\mathrm{ch}_1, \mathrm{ch}_2, \cdots, \mathrm{ch}_M\}$. The channels' state switches between idle and occupied according to the communication behavior of the PU. However, the channel state and switching pattern is unknown for SUs. We assume that there are $K(1 < K < M)$ idle channels, which are feasible for SUs distributed in the network to utilize opportunistically. In this paper, the SUs are supposed to capture the PUs occupation mode and learn to sense and access channels autonomously, so as to achieve the efficient DSA. Due to the limited computational ability and battery life, SUs offload their DSA tasks to the MEC server for computation and analysis. Thereafter, the MEC server distributes the DSA strategies to each SU for the online learning to realize the DSA. In the whole process, no information interaction is required among SUs.

We assume that all the SUs are slot-synchronized, and only part of primary channels can be sensed by each of them, one of which should be accessed by each SU further. Here, the energy detection mechanism is employed [39]. In practice, there are imperfect detections, which may cause the wrong judgement inevitably. And also, since the environment is unstable and channel states are time-varying as mentioned above, the SUs interact with environment to learn how to sense and access a particular channel.

Specially, as depicted in Figure 2, the whole DSA procedure for all SUs can be illustrated as follows: the PU occupies one or more channels at each time slot, and the state of primary channels may change at each time slot. Firstly, each SU senses channels independently to judge whether they are occupied by the PU. Then, the SUs attempt to access one of the sensed channels and send the request signals to the BS, where the MEC server is employed to finish the computation and analysis so that the distributed DSA strategies for each users can be produced. In this way, SUs can learn the switching patterns of the channel states and decide which channel should be sensed in the next time, by analyzing their current DSA scheme and the corresponding feedback received from the BS.

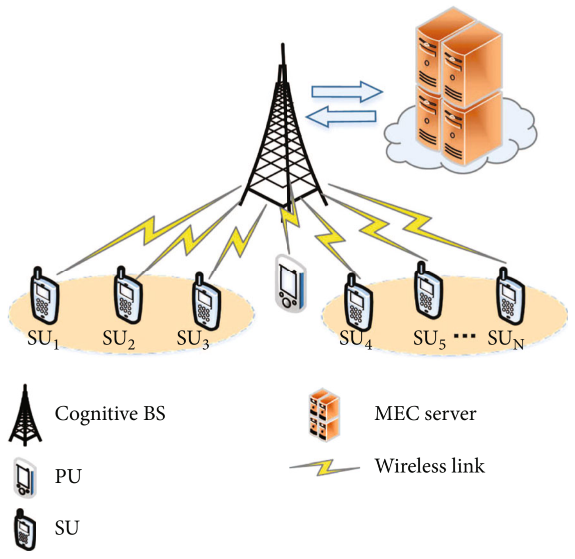


Figure 1: System model of the MEC-enabled cognitive network.

3. Problem Definition with Dec-POMDP

Note that all SUs aim to achieve DSA, in which the objective including full idle channel utilization, and load balance is considered. For each SU, it can only sense part of primary channels, then judges the occupation state of the channel that it accesses without prior coordination among SUs. That is, the multiple SUs distributed in the network can only obtain partial environment information. Therefore, the problem can be modeled as Dec-POMDP, which can be formulated as a tuple $\langle N, S, O, A, P, r\rangle$. The definitions of the tuple elements are listed as follows, and some of the key symbols are summarized in Table 1.

N is the number of SUs who are regarded as multiple agents in the interactive environment.

"S is the global channel state space, which reflects the true state of M orthogonal authorized channels in the communication environment." At time slot t, the channel state space is defined as $S^t \triangleq \{s_1^t, \cdots, s_M^t\}$, where the state of ch_m, $m \in [1, M]$ is given by

$$s_m^t = \begin{cases} -1, \text{if } ch_m, \text{is busy at slot } t, \\ 1, \text{if } ch_m, \text{is idle at slot } t. \end{cases} \tag{1}$$

O is the partial observation space, and it represents the sensed channels for all agents. At each time slot t, agents

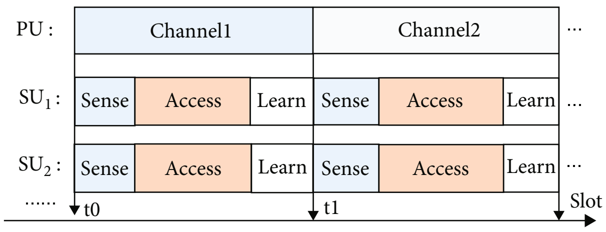


Figure 2: The time-block structure for the DSA procedure.

Table 1: Glossary of key symbols.

Symbol	Description
N	Number of agent
M	Number of orthogonal authorized channel
S	Global channel state space
S^t	Channel state space at slot t
s_m^t	State of channel m at slot t
O	Observation space
O^t	Observation space at slot t for all agents
o_n^t	Observation for agent n at slot t
$o_{n,m}^t$	Observation of channel m for agent n at slot t
A	Action space for all agents
A^t	Action profile for all agents at slot t
a_n^t	Actions for agent n at slot t
$a_{n,m}^t$	Action for agent n who chooses channel m at slot t
r	The set of immediate reward for all agents
r_n^t	Immediate reward for agent n at slot t
c_m^t	Number of agents allocated on the same channel m at slot t
r_{tot}^t	Total reward for all agents at slot t
τ_n	Observation-action history of agent n
R_{tot}	Global reward of all agents in the finite time slots
γ	Discount factor

observe some representation of environment state $O^t \triangleq \{o_1^t, \cdots, o_N^t\}$ from the state S^t. Particularly, the agent may not obtain full and perfect knowledge of the channel states, i.e., for agent $n \in [1, N]$, $o_n^t \neq S^t$. And we define the observation of the agent $n \in [1, N]$ for $ch_m (m \in [1, M])$ as $o_n^t = \{o_{n,m}^t | m \in [1, M]\}$, where

$$o_{n,m}^t = \begin{cases} -1, \text{if } ch_m, \text{is sensed busy at slot } t, \\ 1, \text{if } ch_m, \text{is sensed idle at slot } t. \end{cases} \tag{2}$$

A is the action space for all agents. The action profile for all agents at slot t is formulated as $A^t \triangleq \{a_1^t, \cdots, a_N^t\}$. For the agent $n \in [1, N]$ who chooses the ch_m, $m \in [1, M]$, we define $a_n^t = \{a_{n,m}^t | m \in [1, M]\}$, where

$$a_{n,m}^t = \begin{cases} -1, \text{if the chosen } ch_m, \text{is busy at slot } t, \\ 1, \text{if the chosen } ch_m, \text{is idle at slot } t. \end{cases} \tag{3}$$

P is the state transition matrix, reflecting the transition of channel occupation from state S^t to S^{t+1}.

$r = \{r_1, \cdots, r_N\}$ is the set of immediate reward for all agents after accessing the sensed channels, which encourages agents to learn an optimal DSA strategy. Here, the agents are supposed to independently sense and access a truly vacant and proper channel and obtain a reward according to the feedback from the BS.

The key point of the reward for each agent is to make perfect use of the idle channels in the space as fair as possible. Since the MEC server at BS will collect the channel state and the number of agents who request for the same channel, we design the immediate reward of the n-th agent at slot t as

$$r_n^t = \begin{cases} f\left(\dfrac{c_m^t}{N}\right), s_m^t = 1, a_n^t = ch_m, \left|a_n^t \cup a_{\bar{n}}^t\right| = c_m^t, \\ -1, s_m^t = -1. \end{cases} \tag{4}$$

where the function $f((c_m^t)/(N))$ is defined in the case when the channel available is chosen. With regard to the ratio of the number of agents allocated on the same channel ch_m, $m \in [1, M]$ to the total number of agents, $((c_m^t)/(N))$, it guides the agent to access idle channel properly. And $\bar{n}$ denotes the agents except for agent n, $|\cdot|$ measures the number of agents on ch_m. On the contrary, when a busy channel is wrongly chosen by agent n, a negative reward -1 is occurred and an error identification is informed.

Specifically, $f(\cdot)$ denotes a piecewise function bounded on $1/K$, which is set to guarantee that all the available channels can be utilized fairly. The function is formulated as

$$f\left(\frac{c_m^t}{N}\right) = \begin{cases} (K-1)\dfrac{c_m^t}{N}, 0 < \dfrac{c_m^t}{N} < \dfrac{1}{K}, \\ \dfrac{1}{K}\left(\dfrac{N}{c_m^t} - 1\right), \dfrac{1}{K} < \dfrac{c_m^t}{N} < 1. \end{cases} \tag{5}$$

When $((c_m^t)/(N))$ increases, the reward increases at first and achieves the maximum value at the boundary $1/K$, then decreases rapidly. It signifies that the agent n will get a small reward whether there are too many or too few agents on the selected ch_m. Whereas a balance scheme is explored for all agents in the cognitive network under the limited channels.

Based on the immediate rewards from all agents, the total reward in one slot can be written as

$$r_{\text{tot}}^t = \sum_{n=1}^{N} r_n^t(\tau_n, a_n), \tag{6}$$

where τ_n denotes the observation-action history of agent n. Actually, each agent in the network is supposed to action toward the whole optimal DSA. We call it a cooperative game, which is a special type of exact potential game

(EPG). Based on this theory, the monotonicity of r_{tot}^t conforms to that of r_n^t [40]. Thus we have

$$\underset{\boldsymbol{a}}{\operatorname{argmax}} r_{\text{tot}}^t\left(S^t, \boldsymbol{a}\right) = \begin{pmatrix} \underset{a_1}{\operatorname{argmax}} r_1^t(\tau_1, a_1) \\ \underset{a_2}{\operatorname{argmax}} r_2^t(\tau_2, a_2) \\ \vdots \\ \underset{a_N}{\operatorname{argmax}} r_N^t(\tau_N, a_N) \end{pmatrix}. \tag{7}$$

In the finite time slots, the global reward of all agents can be obtained as

$$R_{\text{tot}} = \sum_{t=1}^{T} \sum_{n=1}^{N} \gamma^{t-1} r_n^t(\tau_n, a_n), \tag{8}$$

where γ is the discount factor, reflecting the influence of the agents' action at the current time slot on the long-term return. And equation (8) can be simply transformed into

$$R_{\text{tot}} = \sum_{t=1}^{T} \gamma^{t-1} \sum_{n=1}^{N} r_n^t(\tau_n, a_n), \tag{9}$$

which can be furtherly integrated as

$$R_{\text{tot}} = \sum_{t=1}^{T} \gamma^{t-1} r_{\text{tot}}^t. \tag{10}$$

The ultimate goal of each agent is to obtain their own optimal spectrum sense and DSA strategies $\pi^* \triangleq (\pi_1^*, \cdots, \pi_N^*)$, so as to maximize the expected cumulative reward of the whole network. The corresponding problem can be formulated as

$$\text{P1}: \pi^* = \underset{\pi:\{a_1,\cdots,a_N\}}{\operatorname{argmax}} E(R_{\text{tot}}), \tag{11}$$

where $E(\cdot)$ denotes the expectation.

From the above defined problem, the agents are expected to possess their excellent abilities of independent perception and decision to maximize a global cumulative reward in a cooperative manner. It is challenging since there is not even a central node to control the whole allocation in practical scenario and no direct information exchange beforehand among agents.

4. QMIX Algorithm for DSA

4.1. Algorithm Description. We consider the QMIX algorithm [41] with the CTDE structure to solve the DSA problem. There are two phases for the DSA of all the agents, i.e., offline training and online execution, respectively. For one thing, in the offline phase, agents perceive environmental information and offload the DSA task to the MEC server, who is responsible to train and issue the distributed DSA strategies by computing and analyzing the received data. For another, in the online phase, the MEC server keeps silence, and each agent executes action autonomously by the learnt strategy.

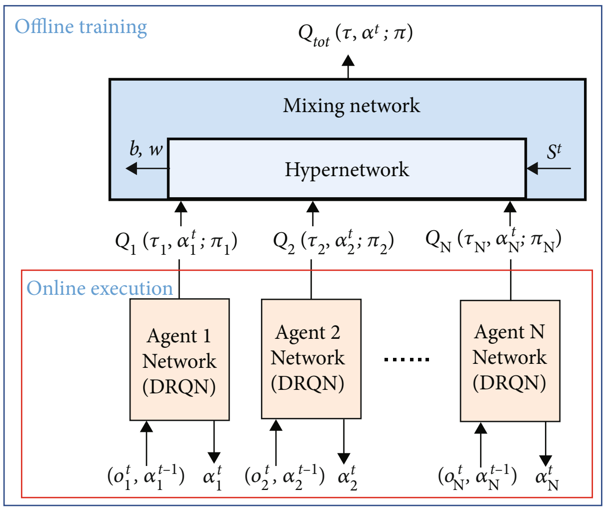

Offline training
$Q_{tot}(\tau, \alpha^t; \pi)$
Mixing network
b, w
Hypernetwork
S^t
$Q_1(\tau_1, \alpha_1^t; \pi_1)$
$Q_2(\tau_2, \alpha_2^t; \pi_2)$
$Q_N(\tau_N, \alpha_N^t; \pi_N)$
Online execution
Agent 1 Network (DRQN)
Agent 2 Network (DRQN)
Agent N Network (DRQN)
(o_1^t, α_1^{t-1}) α_1^t
(o_2^t, α_2^{t-1}) α_2^t
(o_N^t, α_N^{t-1}) α_N^t

FIGURE 3: The structure of QMIX algorithm.

As shown in Figure 3, there are N local agent networks for SUs and one mixing network deployed at the central controller. And the agent networks are constructed by deep recurrent Q-network (DRQN), catering for the agents' partial observation. Here, as the agents considered in the network are homogeneous and also for the system stability, all the DRQNs are equipped with the same network structure and parameters. For any agent $n \in [1, N]$, with the current observation o_n^t and the previous action a_n^{t-1}, the local action value function Q_n is obtained, which enables the agent to choose action a_n^t. Then all the agents' value functions $\{Q_n(\tau_n, a_n^t; \pi_n) | n = 1, 2, \cdots, N\}$ are injected into the mixing network. Note that a hypernetwork is embedded in the mixing network, which makes full use of the global channel state S^t to improve the convergence speed and output the parameters, e.g., bias and nonnegative weight (b, w) for the mixing network. Finally, by the nonlinear map model of the mixing network, the joint action value function $Q_{\text{tot}}(\tau, \boldsymbol{a}^t; \pi)$ is produced.

The advantage of this method is that the monotonicity of Q_{tot} and Q_n can remain the same, i.e.,

$$\frac{\partial Q_{\text{tot}}}{\partial Q_n} \geq 0, \forall n \in [1, N], \tag{12}$$

which well coincides with the property of the problem. Therefore, the relationship between Q_{tot} and Q_n can be furtherly written as

$$\underset{a}{\operatorname{argmax}} Q_{\text{tot}}\left(\boldsymbol{\tau}, \boldsymbol{a}^t; \boldsymbol{\pi}\right) = \begin{pmatrix} \underset{a_1^t}{\operatorname{argmax}} Q_1\left(\tau_1, a_1^t; \pi_1\right) \\ \underset{a_2^t}{\operatorname{argmax}} Q_2\left(\tau_2, a_2^t; \pi_2\right) \\ \vdots \\ \underset{a_N^t}{\operatorname{argmax}} Q_N\left(\tau_N, a_N^t; \pi_N\right) \end{pmatrix}. \tag{13}$$

```
1: Initialization:
    The network environment and experience replay buffer $\mathscr{R}$; the parameters for hypernetwork and all of the agent networks $\pi$;
2: Setting:
    The target-network parameters $\bar{\pi} = \pi$, the learning rate $\alpha$, the discount factor $\gamma$, the batch size $l$, maximum training epoch, episode, slot: Epo, Epi, T, maximum train step $L$;
3: [Centralized Training Phase]:
4:  while $epoch \le Epo$ do
5:    for $episode = 1, \cdots,$ Epi do
6:      for $t = 1, \cdots,$ T do
7:        for each agent $n$ do
8:        Get observation $o_n^t$, action $a_n^t$, reward $r_n^t$;
9:        end for
10:         Get the next observation $o^{t+1}$;
11:         Store the $o^t, a^t, r^t, o^{t+1}$ to the observation-action history;
12:         end for
13:         Store the episode data to the replay buffer $\mathscr{R}$;
14:       end for
15:       for $train \le L$ in each epoch do
16:         Sample a batch of $l$ episodes' experience from $\mathscr{R}$;
17:       for each slot in each sampled episode do
18:           Get $Q_t$ and $Q_{t+1}$ from the evaluate-network and the target-network, respectively;
19:       end for
20:       Calculate the loss function by (14), and update the evaluate-network parameters $\pi = \pi - \alpha \nabla_\pi L(\pi)$;
21:       Update the target-network parameters $\bar{\pi} = \pi$;
22:     end for
23:     Save DRQN and QMIX network models;
24: end while
25: [Decentralized Executing Phase]:
26: Setting: $loadmodel$ = TRUE;
27: Input: The channel state;
28: Output: The agents' observations and actions.
```

Algorithm 1: The proposed QMIX-based DSA algorithm.

By learning the optimal joint value function Q_{tot}, we can obtain the agents' local distributed strategies indirectly. And the update criteria for Q_{tot} is to minimize the loss function $L(\pi)$, which is given by

$$L(\pi) = \sum_{i=1}^{l} \left[y_{\text{tot}}^i - Q_{\text{tot}}\left(\tau, \boldsymbol{a}, S^t; \pi\right) \right]^2, \tag{14}$$

where y_{tot}^i is expressed as

$$y_{\text{tot}}^i = R_{tot}^i + \gamma \max_{\boldsymbol{a}'} \bar{Q}\left(\tau', \boldsymbol{a}', S^{t\prime}; \bar{\pi}\right), \tag{15}$$

where $\bar{Q}(\tau', \boldsymbol{a}', S^{t\prime}; \bar{\pi})$ denotes the target network, supplying for the stable training.

Specifically, the process of the proposed QMIX-based DSA algorithm is listed as Algorithm 1.

4.2. Computational Complexity Analysis. In the proposed QMIX-based DSA algorithm, DRQN is adopted for each agent, which can well handle the Dec-POMDP problem. Besides, some simple activation function, e.g., ReLU and ELU are employed in the algorithm. The operation mainly involves matrix multiplication and addition. In particular, for the DRQN, let us assume that there are V layers, and the number of neural units is $u_\nu (\nu \in [1, V])$ in ν-th layer, and Z is the size of input layer. Then, the number of multiplications through DRQN can be presented as $W = Z \cdot \nu_1 + \sum_{\nu=1}^{V-1} u_\nu \cdot u_{\nu+1}$. For each agent, the computational complexity of one sample is $O(W)$. Note that the offline training is

Table 2: Simulation parameters.

Parameters	Values
Number of PUs	1
Number of SUs	9
Number of MEC servers	1
Number of authorized channels	4
Discount factor γ	0.99
Learning rate α	$5 * 10^{-4}$
Replay buffer capacity	100 episodes of experience data
Batch size l	16 episodes of experience data
Updating step	40 steps
Training epoch Epo	2000 epochs
Training episode Epi	100 episodes
Training slot T	20 slots
Exploration probability ε	0.4 to 0.02

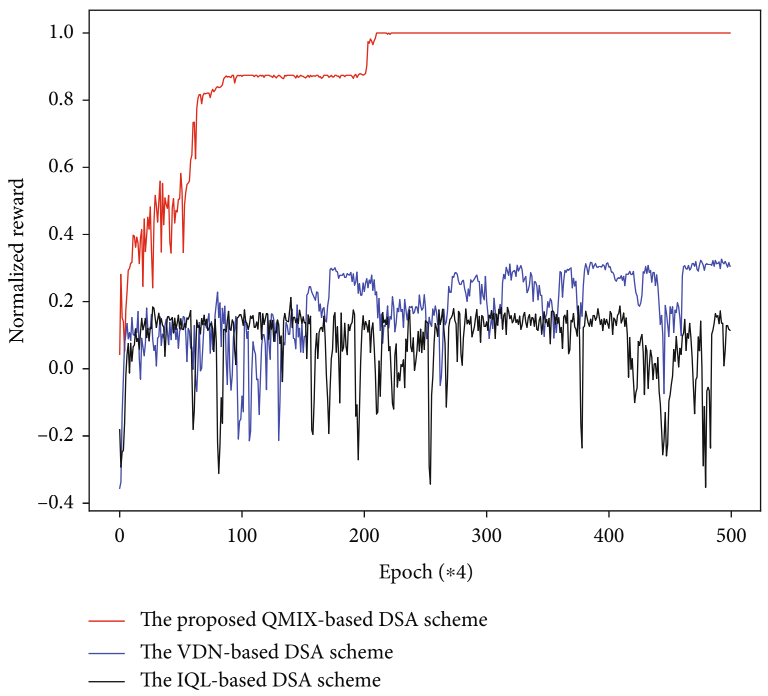


Figure 4: Normalized reward versus training epoch under different schemes.

parallelly worked at the edge server in the training phase. The training complexity of one batch of l episodes under T training slots is $O(lTW)$. And the whole computational complexity is $O(IlTW)$ until the algorithm converges over I iterations. Further, the computational complexity of the execution phase is $O(W)$, since each SU acts locally at each time step. Due to its monotonicity, the complexity increases linearly with the increase of input scale, which greatly improves the efficiency of the algorithm. Therefore, less computational resource is required in practice.

5. Simulation Results

In this section, we provide the simulation parameter setting, and then evaluate the performance of the proposed QMIX-based DSA scheme and the rationality of the defined problem via simulation.

Since there is a mixing network, a hypernetwork and N agent networks in the proposed algorithm, the corresponding network parameters setting are illustrated as follows: the mixing network, which brings the global action value from local action values, has one hidden layer of 32 neurons, and the nonlinear function ELU is employed as the activation function. For the hypernetwork, it consists of one hidden layer which has 64 neurons with ReLU as the activation function. Each agent network is with one recurrent layer employing a GRU with 64-dimension hidden state. Unless otherwise specified, other simulation parameters are summarized in Table 2.

In particularly, for hyperparameters, the replay buffer is capacity-limited which can store 100 sets of data, and the oldest data will be removed when the buffer is full. The batch size l for sampling is 16 episodes. The target networks of the DRQN are updated every 40 training steps. In the whole process, it is considered that there are 2000 training epochs, each epoch has 100 episodes, in which 20 time slots are regarded as one episode. In the training phase, to encourage the agent to explore the environment, the "explore and exploit" mechanism is employed by agents to choose actions [42]. The exploration probability ε decays from 0.4 to 0.02 over 400 steps. Then, in order to timely evaluate the quality of the training performance, the distributed execution is conducted every four training epochs, where we set $\varepsilon = 0$, and the agents make decisions only by the local models. For the environment setting, we assume that the channels change in periodic mode and each SU can only sense one channel.

To verify the advantages of the proposed scheme, in Figure 4, we compared our proposed scheme with two other schemes: (1) the IQL-based DSA scheme [43] and (2) the VDN-based DSA scheme [44]. We take nine SUs and four channels with one channel unavailable to compare the performance of these schemes. The abscissa is the training epoch, and the ordinate is the normalized reward. The simulation results show that, the performance curves of IQL and VDN based schemes show relatively large fluctuation, which are far from the best effect. It can be seen that the maximum value under IQL is only 0.17, while VDN is better than IQL reaching just 0.32. The reasons for this result can be explained as follows: for IQL-based scheme, each agent operates independently in the whole learning process, which is not conducive to the stability and convergence. And for VDN-based scheme, it does not use global state information during central training, and a simple weighted summation method is used to decompose the joint value function to update the agents' strategies, causing a bad training effect. In addition, due to the powerful fitting ability and the integration of global environment information, an excellent DSA effect is achieved in our proposed QMIX-based scheme. Therefore, the DSA performance of our proposed QMIX-based DSA scheme outperform other two schemes.

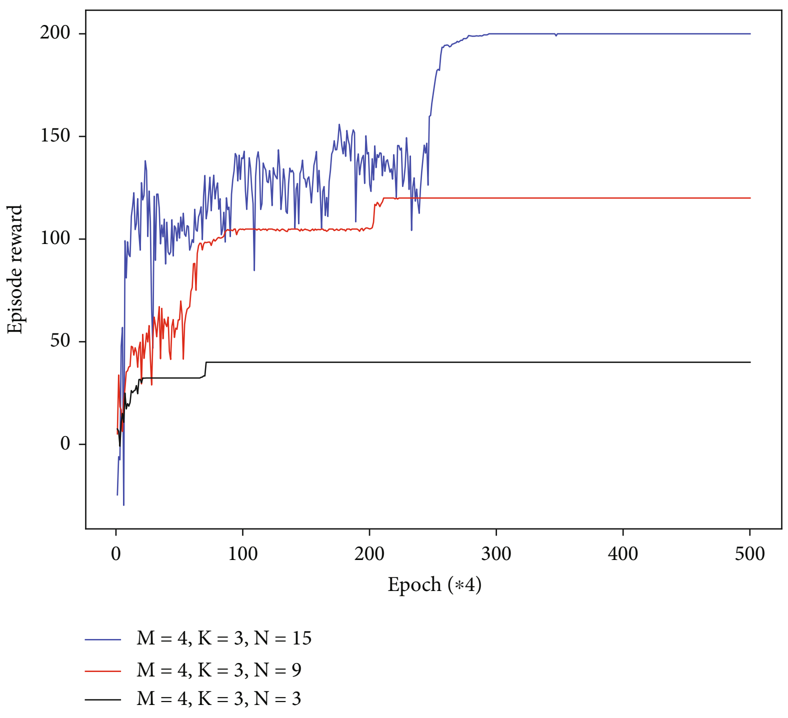


Figure 5: Episode reward versus training epoch under different number of SUs.

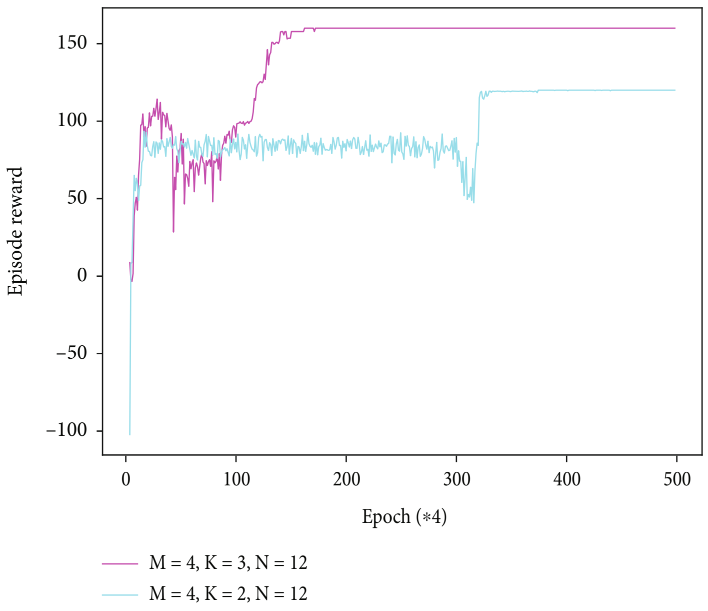


Figure 6: Episode reward versus training epoch under different channels available.

Figure 5 displays the sum of rewards obtained by all users in an episode versus training epoch in the case of different number of SUs. There are four channels considered in the network, where there are always three channels available for SUs that changes periodically over time. It can be seen intuitively that under three different settings, as the number of training epoch increases, the total episode reward increases gradually, and finally reaches to the maximum value within limited training. Note that the negative reward happens at the beginning of the training, which can be explained that some SUs select the nonidle channels, since the agents' network models are still rough at that moment, although four epochs' training is done. Specially, when there are 15 SUs, the initial episode reward is about -30, and then a longest time is experienced for convergence. This is because the more number of SUs means the larger calculation dimension and the slower learning speed, which makes it more difficult for SUs to learn and analyze the environment. That is, through the proposed method, firstly, the SUs have learned to capture the spectrum holes. On this basis, the load balance is realized, so as to obtain the

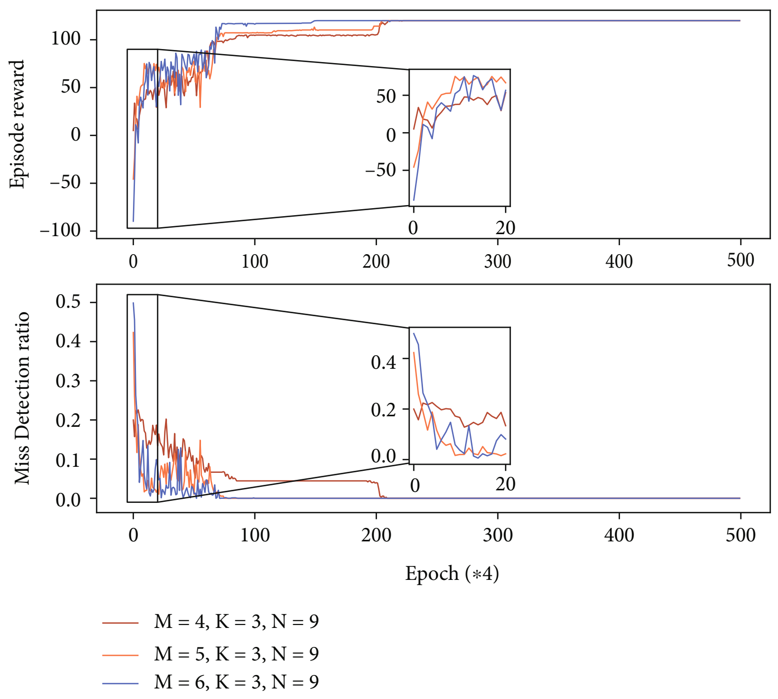


FIGURE 7: Episode reward and miss detection ratio versus training epoch under different number of authorized channels.

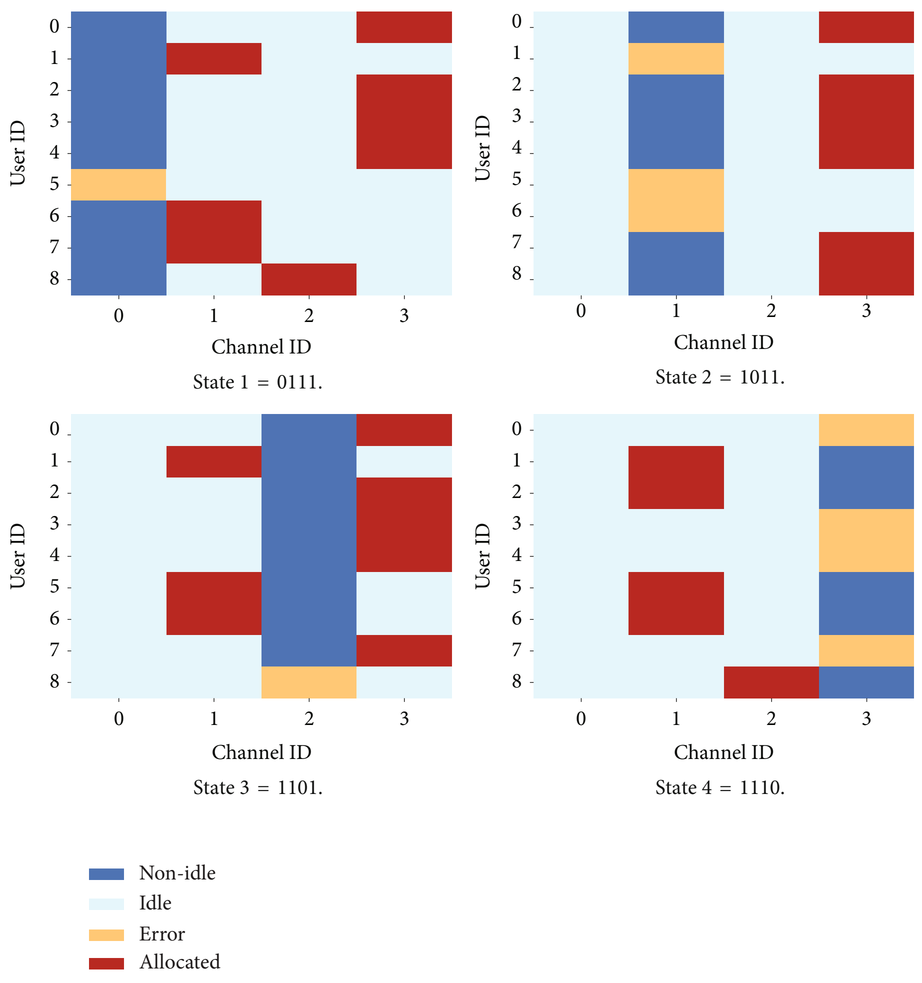


FIGURE 8: The distributed execution after the initial training.

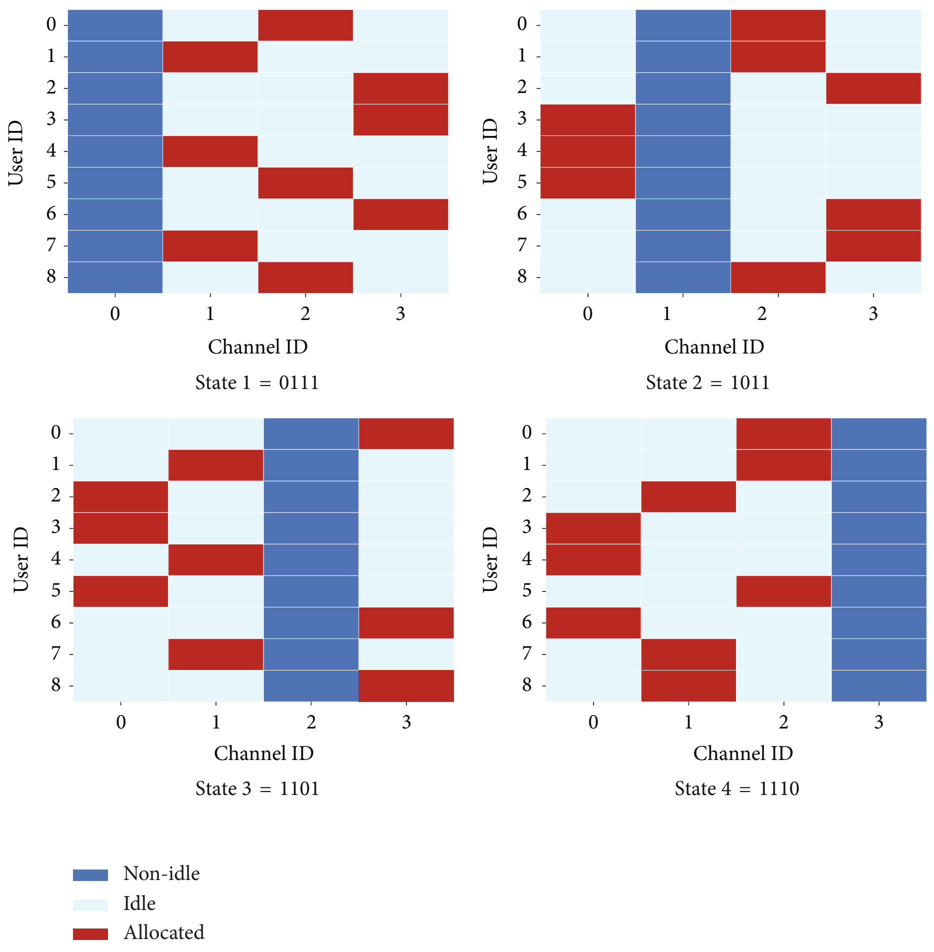


Figure 9: The distributed execution after the final training.

optimal DSA. Besides, we can observe that, when the networks are well trained, more episode reward value of the system can be achieved with more users. It is related to the definition of global reward in cooperative MARL environment, which integrates all SUs' rewards.

In Figure 6, the behavior of the episode rewards under different number of idle channels is plotted. To facilitate the comparison, we set the idle channels $K = 2, 3$ for $M = 4$, and 12 SUs are participating in the DSA. Likewise, we can observe that the curves fluctuate but overall increase and then converge to the maximum value as the training epoch increases. As for the situation of the fewer channels available, it can be seen that a slower convergence speed is acquired, which means the fewer optional channels requires more time for multiple SUs to make a "trial and error" until the model is well-done, so as to achieve the optimal DSA. We can also observe that at the beginning of the training, a quite low negative value, about -100, is caused when $K = 2$, i.e., there are two unavailable channels. It can be explained that the more number of unavailable channels, the greater probability that the SUs make a wrong decision to choose unavailable channels. It also reflects the huge influence of the number of unavailable channels on network learning effect. What is more, it is also shown that the more available channels, the larger total episode reward is obtained, which reveals that the more available channels brings the better balance among these channels.

Figure 7 evaluates the performance with respect to number of authorized channels, which are set as four, five, and six, and the PU occupies one, two, or three channels correspondingly to ensure the same channels available for SUs. Considering that the SUs' sense accuracy is also an important indicator for the DSA, the miss detection ratio is evaluated as well. From the presented reward curves, we can observe that in three different cases, the episode rewards are increasing gradually. Since there is little difference among the set of available channels and SUs, the curves are entangled with each other, and the values are relatively close in the whole process. Finally, they all converge at the same value, which is also the optimal value after the network model is fully trained.

Meanwhile, the miss detection ratio behaves an opposite trend. This intuitively shows that the agents' detection ability is indeed weak as they experience little learning, which makes it easy for them to make the error decision. And we find that, when more channels are occupied by the PU, the initial miss detection ratio is greater, which is consistent with the reward at the starting stage. Then, with the increase of training time, the SUs achieve a perfect detection (miss detection ratio = 0). At this time, compared with reward curves, it can be seen that the reward value has not reached the optimal value, but converges after more training. This shows that after overcoming the imperfect detection, it takes some time to further realize load balance for SUs on the available channels.

To be more intuitive, Figures 8 and 9 present the effects of the distributed execution in four consecutive channel states under the initial and final training phases, respectively. Here, the initial phase is the first four training epochs, while the final phase is the last epoch when the models are fully trained. We assume that nine SUs independently make a choice from four authorized channels in different channel states for instance.

We can see from Figure 8 that there are many wrong channel detections and selections in the fourth tested states. Due to the fact that the SUs have no more knowledge of the channel variation characteristics in the initial training phase, they access channels almost randomly. Specially, in the state2, although there are three idle channels, the SUs gather in the fourth channel. This reduces the channel utilization and also causes congestion of other idle channels. Whereas, we can see from Figure 9 that the result of DSA is well done after the final training. Not only no one selects the occupied channel but also the SUs on each available channels are balanced. The goal of the DSA in the considered scenario is achieved, and the effectiveness of proposed algorithm is demonstrated. Besides, we have counted the real execution time under one of the channel states, and we find that it takes only about 26.59 ms for all SUs' DSA. It highlights the low computational complexity of the proposed algorithm furtherly. Then, since all SUs' collected spectrum data has no error after training, the availability, accuracy, and timeliness of the acquired spectrum data can be guaranteed.

6. Conclusion

In this paper, we have studied the distributed DSA strategies for multiple cognitive users in MEC-enabled network, where the spectrum environment is time-varying, and the users make decisions with imperfect spectrum sensing. The DSA task, including capturing the spectrum hole and achieving the load balance on channels available, is investigated. We modeled the problem as Dec-POMDP, and a QMIX-based DSA algorithm is proposed, which allows users offload their task to the MEC server to train the network models. We evaluated the system reward and the miss detection ratio of the DSA by the proposed algorithm. The results showed the rationality of the model and the effectiveness of the proposed algorithm.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by National Natural Science Foundation of China under Grant Nos. 62171449, 61931020 and 62001483.

References

[1] P. Mach and Z. Becvar, "Mobile edge computing: a survey on architecture and computation offloading," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[2] U. Drolia, R. Martins, J. Tan et al., "The case for mobile edge-clouds," in *2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing*, pp. 209–215, Vietri sul Mare, Italy, 2013.

[3] Z. Ning, Y. Yang, X. Wang et al., "Dynamic computation offloading and server deployment for UAV-enabled multi-access edge computing," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[4] X. Tang, Z. Wen, and J. E. A. Chen, "Joint optimization task offloading strategy for mobile edge computing," in *2021 IEEE 2nd international conference on information technology, Big Data and Artificial Intelligence (ICIBA)*, pp. 515–518, Chongqing, China, 2021.

[5] X. Wang, Z. Ning, S. Guo, M. Wen, L. Guo, and V. Poor, "Dynamic UAV deployment for differentiated services: a multi-agent imitation learning based approach," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[6] H. Zhao, K. Ding, N. I. Sarkar, J. Wei, and J. Xiong, "A simple distributed channel allocation algorithm for D2D communication pairs," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 10960–10969, 2018.

[7] J. Li, H. Zhao, J. Wei, D. Ma, and L. Zhou, "Sender-jump receiver-wait: a simple blind rendezvous algorithm for distributed cognitive radio networks," *IEEE Transactions on Mobile Computing*, vol. 17, no. 1, pp. 183–196, 2018.

[8] J. Li, H. Zhao, S. Zhang, A. S. Hafid, D. Niyato, and J. Wei, "Cross-Layer analysis and optimization on access delay in channel-hopping-based distributed cognitive radio networks," *IEEE Transactions on Communications*, vol. 67, no. 7, pp. 4654–4668, 2019.

[9] S. Zhang, A. S. Hafid, H. Zhao, and S. Wang, "Cross-layer rethink on sensing-throughput tradeoff for multi-channel cognitive radio networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 10, pp. 6883–6897, 2016.

[10] W. Zhang, Y. Sun, L. Deng, C. K. Yeo, and L. Yang, "Dynamic spectrum allocation for heterogeneous cognitive radio networks with multiple channels," *IEEE Systems Journal*, vol. 13, no. 1, pp. 53–64, 2019.

[11] T. H. Lim, W. S. Jeon, and D. G. Jeong, "Centralized channel allocation scheme in densely deployed 802.11 wireless LANs," in *2016 18th International Conference on Advanced Communication Technology (ICACT)*, pp. 249–253, PyeongChang, Korea, 2016.

[12] P. Shetkar and S. B. Ronghe, "Spectrum sensing and dynamic spectrum allocation for cognitive radio network," in *2018 4th International Conference for Convergence in Technology (I2CT)*, pp. 1–5, Mangalore, India, 2018.

[13] Z. Ning, S. Sun, X. Wang et al., "Blockchain-enabled intelligent transportation systems: a distributed crowdsensing framework," *IEEE Transactions on Mobile Computing*, 2021.

[14] Q. Cong and W. Lang, "Deep multi-user reinforcement learning for centralized dynamic multichannel access," in *2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP)*, pp. 824–827, Xi'an, China, 2021.

[15] Q. Cong and W. Lang, "Double deep recurrent reinforcement learning for centralized dynamic multichannel access," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 5577756, 10 pages, 2021.

[16] J. Li, H. Zhao, A. S. Hafid, J. Wei, H. Yin, and B. Ren, "A bio-inspired solution to cluster-based distributed spectrum allocation in high-density cognitive internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9294–9307, 2019.

[17] X. Pei, "A novel dynamic spectrum allocation and power control algorithm with fairness for multi-hop cognitive radio networks," in *2015 IEEE 5th International Conference on Electronics Information and Emergency Communication*, pp. 440–443, Beijing, China, 2015.

[18] M. M. A. Osman, S. K. S. Yusof, and N. N. N. Abd Malik, "Load balanced clustering algorithm for cognitive radio ad hoc networks," in *2018 2nd International Conference on Telematics and Future Generation Networks (TAFGEN)*, pp. 43–48, Kuching, Malaysia, 2018.

[19] P. Li, Z. Zhao, D. Liu, and D. Hou, "The research of dynamic spectrum allocation based on game theory," in *2018 IEEE 3rd advanced information technology, Electronic and Automation Control Conference (IAEAC)*, pp. 14–17, Chongqing, China, 2018.

[20] P. Li, B. Han, H. Li, D. Hou, D. Liu, and G. Wang, "The research of dynamic spectrum allocation based on Nash bargaining game," in *2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC)*, pp. 70–74, Chongqing, China, 2018.

[21] E. E. Tsiropoulou, A. Kapoukakis, and S. Papavassiliou, "Energy efficient subcarrier allocation in SC-FDMA wireless networks based on multilateral model of bargaining," in *2013 IFIP Networking Conference*, pp. 1–9, Brooklyn, NY, USA, 2013.

[22] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, 2nd edition, 2018.

[23] M. Sohaib, J. Jeong, and S.-W. Jeon, "Dynamic multichannel access via multi-agent reinforcement learning: throughput and fairness guarantees," *ICC 2021 - IEEE International Conference on Communications*, pp. 1–6, 2021.

[24] J. Chen, Z. Gao, and Y. Xu, "Opportunistic spectrum access with limited feedback in unknown dynamic environment: a multi-agent learning approach," in *The 2014 5th International Conference on Game Theory for Networks*, pp. 1–6, Beijing, China, 2014.

[25] H.-H. Chang, H. Song, Y. Yi, J. Zhang, H. He, and L. Liu, "Distributive dynamic spectrum access through deep reinforcement learning: a reservoir computing-based approach," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1938–1948, 2019.

[26] H. Li, "Multi-agent q-learning for competitive spectrum access in cognitive radio systems," in *2010 Fifth IEEE Workshop on Networking Technologies for Software Defined Radio Networks (SDR)*, pp. 1–6, Boston, MA, USA, 2010.

[27] H. Agrawal and K. Asawa, "Decentralized learning for opportunistic spectrum access: multiuser restless multiarmed bandit formulation," *IEEE Systems Journal*, vol. 14, no. 2, pp. 2485–2496, 2020.

[28] Y. Li, Y. Wu, and W. Jia, "Dynamic spectrum allocation enabled multiuser latency minimization in mobile edge computing," in *2020 16th International Conference on Mobility, Sensing and Networking (MSN)*, pp. 161–168, Tokyo, Japan, 2020.

[29] Q. Li, Y. Sun, Z. Hao, and Y. Zhang, "Energy efficient spectrum resource allocation in mobile edge computing," in *2019 IEEE 5th International Conference on Computer and Communications (ICCC)*, pp. 1114–1119, Chengdu, China, 2019.

[30] Z. Ning, S. Sun, X. Wang et al., "Intelligent resource allocation in mobile blockchain for privacy and security transactions: a deep reinforcement learning based approach," *Science China Information Sciences*, vol. 64, no. 6, pp. 1–6, 2021.

[31] U. Kaytaz, S. Ucar, B. Akgun, and S. Coleri, "Distributed deep reinforcement learning with wideband sensing for dynamic spectrum access," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, Seoul, Korea, 2020.

[32] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for dynamic spectrum access in multichannel wireless networks," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pp. 1–7, Singapore, 2017.

[33] X. Tan, L. Zhou, H. Wang et al., "Cooperative multi-agent reinforcement learning based distributed dynamic spectrum access in cognitive radio networks," *IEEE Internet of Things Journal*, 2022.

[34] X. Wang, Z. Ning, S. Guo, M. Wen, and H. V. Poor, "Minimizing the age-of-critical-information: an imitation learning-based scheduling approach under partial observations," *IEEE Transactions on Mobile Computing*, vol. 21, no. 9, pp. 3225–3238, 2022.

[35] O. H. Toma, M. Lpez-Bentez, D. K. Patel, and K. Umebayashi, "Reconstruction algorithm for primary channel statistics estimation under imperfect spec trum sensing," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–5, Seoul, Korea, 2020.

[36] D. Silver and J. Veness, "Monte-Carlo planning in large POMDPs," in *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems*, pp. 2164–2172, Vancouver, British Columbia, Canada, 2010.

[37] A. Somani, N. Ye, D. Hsu, and W. S. Lee, "DESPOT: online POMDP planning with regularization," *Advances in Neural Information Processing Systems*, vol. 58, 2013.

[38] P. Cai, Y. Luo, D. Hsu, and W. S. Lee, "Hyp-DESPOT: a hybrid parallel algorithm for online planning under uncertainty," *Robotics: Science and Systems Foundation*, vol. 40, no. 2-3, pp. 558–573, 2021.

[39] M. Gupta and G. Yerma, "Improved weighted cooperative spectrum sensing algorithm based on reliability in cognitive radio networks," in *IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology*, pp. 609–612, Bangalore, India, 2017.

[40] D. Monderer and L. S. Shapley, "Potential games," *Games and Economic Behavior*, vol. 14, no. 1, pp. 124–143, 1996.

[41] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, *Qmix: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning*, PMLR, 2018.

[42] E. R. Gomes and R. Kowalczyk, "Modelling the dynamics of multiagent Q-learning with greedy exploration," in

Proceedings of the International Conference on Autonomous Agents & Multiagent Systems, pp. 1181-1182, Budapest, Hungary, 2009.

[43] M. Tan, “Multi-agent reinforcement learning: Independent vs. cooperative agents,” in *The 10th International Conference on Machine Learning*, pp. 330–337, Amherst: University of Massachusetts, 1993.

[44] P. Sunehag, G. Lever, A. Gruslys et al., “Value-decomposition networks for cooperative multi-agent learning based on team reward,” in *The 17th International Conference on Autonomous Agents and Multiagent Systems*, pp. 2085–2087, Stockholm, Sweden, 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 5031112, 18 pages
https://doi.org/10.1155/2022/5031112

Research Article

An Efficient Data Sharing Scheme for Privacy Protection Based on Blockchain and Edge Intelligence in 6G-VANET

Zhihua Wang,[1] **Yingheng Xu**,[1] **Jiahao Liu**,[1] **ZhenYu Li**,[1] **Zeminghui Li**,[1] **Hongyong Jia**,[1] **and Dinghua Wang**[2]

[1]*School of Cyber Science and Engineering, Zhengzhou University, Zhengzhou 450001, China*
[2]*National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China*

Correspondence should be addressed to Hongyong Jia; hyjia@zzu.edu.cn and Dinghua Wang; wangdinghua666@163.com

Received 1 May 2022; Revised 13 June 2022; Accepted 23 June 2022; Published 19 July 2022

Academic Editor: Zhaolong Ning

With the substantial increase in the number of smart cars, vehicular ad hoc network (VANET), where data can be shared between vehicles to enrich existing vehicle services and improve driving safety, is gaining more and more attention, thus creating a more efficient intelligent transportation system. Moreover, the in-depth research and development of 6G and AI technology further strengthen the interconnection of various entities in VANET and can realize edge intelligence, which fundamentally enhances the efficiency of data sharing. However, reliable transmission and secure storage of data have always been a great challenge in data sharing. Although some schemes store shared data in the blockchain, most of the consensus mechanisms they use employ full nodes to verify signature information and timestamps, which cannot effectively judge the reliability of the shared data itself. Some other schemes use scoring mechanisms to evaluate data uploaded by vehicles, but these methods can be affected by network hardware failures and cannot effectively detect duplicate data. In addition, participants' privacy may also be disclosed in the process of data sharing, such as participants' location and identity information. Therefore, to address the above problems, this paper proposes a data sharing scheme in 6G-VANET, which can not only ensure the reliability and security of shared data but also protect the privacy of participants. Firstly, a consortium chain is adopted to realize the secure storage of shared data in 6G-VANET, which meets the requirements of tamper-proof and traceability of data. Secondly, a voting consensus mechanism is designed in combination with smart contract to ensure the reliability of data. Thirdly, the trained word2vec natural language processing model is deployed to edge nodes to realize edge intelligence, effectively eliminate the duplicate shared data, and enhance storage efficiency. Finally, a participant privacy protection mechanism is designed using the Private Set Intersection (PSI) protocol, and a secure and efficient data sharing scheme is finally realized. The effectiveness of the proposed scheme is demonstrated by security analysis and experimental evaluation. The experimental results show that the time and space overhead of blockchain can meet the practical requirements, and the proposed PSI protocol of large-scale vehicles can be completed in a short time.

1. Introduction

Currently, vehicles are equipped with various wireless communication modules and an increasing number of sensors, which enable vehicles to generate a large amount of data during driving; for example, self-driving vehicles can collect 1 GB of data per second from in-vehicle cameras, radars, GPS, and other sensors [1]. Vehicles can share road information through these sensors to nearby vehicles or to the cloud, which has promoted to the rapid development of VANET. In general, the data shared between vehicles can be divided into two types, which are subjective type data and objective type data. Objective information is mainly related to road, traffic, and environment information, such as road congestion or damage, weather conditions, and parking lot location occupancy information. Subjective information is mainly related to the rating of service quality, such as the rating of roadside hotels or car repair shops.

In the future, with the increase of infrastructure and the number of vehicles, as well as larger network coverage, limited wireless resources and even wired connections greatly limit the development of Internet of Things (IOT) [2], and 5G network will not meet the requirements of VANET [3]. In contrast, 6G can seamlessly integrate various wireless networks spread over ground, underwater, air, and space [4], enabling complete automation. Therefore, in a oriented 6G-VANET, this big data can be collected and shared collaboratively among vehicles for higher quality of service [5].

In the process of sharing data, if the accuracy of data and the privacy of participants cannot be guaranteed, it will endanger people's life and property safety on the one hand and reduce the user's participation on the other hand. Therefore, security and efficient vehicle data sharing can improve the safety of vehicle driving process and improve road, traffic, and environment management; so, it is gradually becoming a current research hotspot.

In order to create a secure and efficient VANET data sharing environment, reliable collection and secure storage of data is always a great challenge. In a large-scale VANET, malicious vehicles spreading false information will greatly harm the traffic system and cause some losses to the vehicle users [6]. A number of scholars have studied the reliability of shared data, and Kang et al. [7] and Xie et al. [8] proposed a sharing scheme based on shared data and videos, respectively, with a reputation mechanism that can effectively prevent the proliferation of malicious data. However, network or hardware failures or viruses may cause vehicles to be rated too low or too high in a given event. Next, the use of reputation value may suffer from cooperative attacks, where some malicious vehicles intentionally give a high score to a certain vehicle, and then this vehicle can send some false messages. Again, false data may also be sent when the identity information of a vehicle with a high historical reputation value is stolen. Finally, most of the current data sharing articles do not involve data deduplication mechanisms, and uploading duplicate data can put pressure on computing storage resources. At the same time, secure storage of data is also important for data sharing. Due to heavy data traffic load, traditional centralized data storage may face many problems and challenges. Attackers may tamper with false data stored in data center nodes or evade punishment by launching various attacks (e.g., DDoS attacks and single point of failure attacks) on data center nodes to compromise data integrity, which leads to the requirement for distributed storage in VANET. In recent years, blockchain technology has received more and more attention and research in VANET because of its decentralization, anonymity, traceability, and tamper-evident features. Smart contracts in blockchain have the characteristics of being tamper-proof, distributed, and automatically triggered, which can ensure trusted transactions without the third party. Therefore, data sharing based on blockchain is a feasible solution to ensure reliable collection and secure data storage.

In order to achieve large-scale data sharing in VANET, it will be a challenge whether the communication resources as well as the computation and storage resources can meet the practical requirements. Due to the limitation of communication resources, it is difficult for vehicles to transmit massive amounts of data on a large scale in a short period of time, and the data generated by vehicles becomes increasingly fine-grained and complex, which increases the burden of data transmission [9]. Since its early days, VANET has used communication standards such as dedicated short range communication (DSRC), long-term evolution (LTE), and 802.11P. Due to the complex road environment, these technologies present challenges in terms of speed, delay, reliability, and connectivity during data propagation [10]. Although the communication resources of the Internet of things can be expanded through 5G, with the rapid growth of IOT terminals and big data services, the shortage of communication resources is still not effectively solved [11]. In order to solve this problem, a new VANET communication standard, cellular vehicular network (C-V2X), has been proposed internationally. The standard realizes assisted driving through vehicle cooperation, improves driving safety and traffic efficiency, and reduces costs. On November 18, 2020, Federal Communications Commission (FCC) decided to allocate the 5.9 GHz frequency band (5.850-5.925 GHz) originally allocated to DSRC (ieee802.11p) to WiFi and C-V2X, of which 30 MHz bandwidth (5.895-5.925 GHz) was allocated to C-V2X, which marked that the United States officially abandoned DSRC and turned to C-V2X. C-V2X will become a new international standard for VANET communication. The decoupling of data control in Software-Defined Network (SDN) enables more flexible control and optimization of the network and improves network performance [12]. The deployment of blockchain smart contracts in the SDN control plane can effectively prevent the SDN controller from being hijacked and is therefore suitable for the management of large-scale vehicular networks. In addition, Network Functions Virtualization (NFV) can decouple software and hardware and can provide network functions on standard servers. Therefore, by further combining SDN with NFV, a more efficient network system can be built for the network requirements of 6G-VANET.

For computational and storage resources, cloud computing can provide a large amount of computational resources and massive storage space, but the cloud cannot meet the latency requirements of in-vehicle terminals when they provide applications and services for transportation and entertainment [13]. To relieve the pressure on the cloud and meet the demand for computational and storage resources in VANET, MEC (mobile edge computing, MEC) becomes a natural candidate. MEC will complete computation at the local edge, which will undoubtedly greatly improve processing efficiency and reduce the load on the cloud. However, if the data uploaded by users is not filtered, MEC will also face great pressure on computing and storage. Traditional data similarity judgment methods include Levenshtein distance, Jaccard similarity coefficient, and cosine similarity, but these methods have low accuracy. In order to improve the data processing efficiency of MEC, it needs to be combined with edge intelligence. The sentence similarity methods based on machine learning include doc2vec and word2vec. The difference between them is that doc2vec is suitable for judging long multiparagraph text, while word2vec is suitable for judging short single paragraph text. Although there are a lot of shared data in VANET, the data shared by each vehicle during driving is not large; so, word2vec has become a perfect candidate. word2vec is a natural language processing toolkit based on deep learning launched by Google in 2013. It is a model for learning semantic knowledge from a large number of text messages in an unsupervised way. It can obtain word vectors in sentences, which is simple and efficient; so, it has attracted extensive attention. word2vec uses a standard three-layer neural network model, which is initially used to predict the relationship between word vectors. word2vec can be combined with cosine similarity to further judge the similarity of sentences. Deploying the model in MEC combined with the timestamp of shared data can effectively remove duplicate data in a short time and save the overhead of computing and storage resources. However, the traditional MEC is deployed in RSU (Road Side Unit) or BS (Base Station). Because the location of these facilities is difficult to move, the coverage will be limited and cannot meet the actual needs of large-scale VANET. In order to meet these challenges, with the characteristics of simple deployment and convenient movement, UAV (Unmanned Aerial Vehicle) can be combined with MEC to provide users with ubiquitous edge computing services through computing offload [14].

Currently, there is a lack of incentives in most data sharing schemes, and vehicle users may be reluctant to participate in sharing data unless they can benefit from such participation. Therefore, users who contribute to sharing accurate data should receive some rewards. Zhang et al. [15] proposed a reward and punishment mechanism (VCoin) to encourage user participation, but the paper only mentions an idea and does not incorporate a practical application.

In the process of data sharing, identity privacy and data privacy must also be considered. For shared data, data sharing can be achieved by matching similar profile attributes for socialization purposes [16]. However, during this information matching process, vehicles cannot reveal their respective sensitive information (e.g., driver's location and identity information) to each other. If these sensitive information is disclosed, the attacker can infer the user's behavior patterns, preferences, habits, and interests through the user's sensitive information, which will threaten the user's property and security [17]. PSI protocol can perfectly solve this problem by completing the information matching of users on the one hand and protecting the privacy of each participating party on the other hand. Since the privacy of the announcement type data is not as high as that of the private data, the tedious data encryption and decryption operations can be eliminated in the process of verifying the accuracy of the data, and only the vehicle users with the same characteristics need to judge the data.

Therefore, to address the above problems, a secure and efficient data sharing solution that meets privacy requirements is proposed. Firstly, considering privacy requirements, a cloud-assisted PSI protocol is designed as a blockchain voting consensus mechanism, which allows vehicles with the same characteristics to make subjective judgments about the accuracy of data to improve the accuracy of data evaluation, thus protecting user privacy and resisting complicity attacks. Secondly, to prevent data tampering and ensure data integrity, this paper uses blockchain technology to design a data storage scheme and a smart contract for voting. The vehicle sharing data as well as the voting process will generate a transaction and pack into a block, and the identity of the vehicle can be verified through the transaction address, which replaces the traditional signature verification and further enhances the trust of the whole system. Thirdly, to facilitate blockchain maintenance and slow down the storage overhead of the blockchain, this paper uses word2vec and cosine similarity comparison to de duplicate data suspected to be duplicates in a short period of time. To mitigate the communication, this paper uses MEC, SDN, and other technologies to build a novel C-V2X communication architecture to meet the communication requirements in VANET. Finally, to address the incentive requirements, this paper gives a method of reward issuance in the constructed blockchain platform, which is done by the blockchain administrator node invoking a smart contract or directly sending a transfer transaction.

Overall, three contributions are as follows.

(1) By combining SDN, MEC, blockchain, and other technologies, a novel 6G-VANET network architecture is proposed to logically ensure the data sharing process is secure and efficient

(2) A PSI protocol combining hash and random number is designed and given to edge nodes for execution, which ensures data and vehicle privacy in the data sharing process while reducing the overhead in the matching process

(3) A blockchain system for 6G-VANET is constructed, and the reliability, integrity, and tamper-evidence of data in the data sharing process are ensured by designing smart contracts and consensus mechanisms

The rest of the paper is as follows: in Section 2, the current related research work is analyzed. In Section 3, the system architecture and problem description are given. Section 4 describes the data sharing scheme process in detail. Section 5 gives the security proof and experimental results. Section 6 concludes, and a future plan is made.

2. Related Work

2.1. Data Sharing. In traditional data sharing approaches, mostly centralized servers are used to collect and store the data shared by all entities. Ali et al. [18] proposed a cloud secure data sharing approach (SeDaSC) to ensure data confidentiality and integrity, and Dong et al. [19] proposed a framework for securely sharing sensitive data on a big data platform that enabled secure data delivery, storage, usage, and destruction. However, centralized storage approaches still faced the threat of security and privacy risks, where attackers could easily forge or tamper with data in open wireless communication environments, and may cause single points of failure. In recent years, blockchain technology has attracted a lot of attention in the field of IoT and security, and many scholars have introduced blockchain into data sharing. Kouicem et al. [20] proposed a decentralized and anonymous data sharing scheme based on the construction of blockchain, smart contracts, and zero-knowledge proof. Feng et al. [21] applied blockchain and attribute-based cryptography to propose a secure efficient data sharing model and used smart contracts for authentication and access control. Zhang and Chen [22] proposed a data sharing framework using a federated blockchain to maintain the data storage and sharing system by applying smart contracts through preselected nodes and used digital signature techniques to ensure the integrity and security of data during data sharing. The consensus process in most articles is more similar to [16], which only verified the digital signature and timestamp, which did not guarantee the correctness of the data itself and could not judge the repeatability of the data. The scheme proposed in this paper replaces the digital signature with the transaction information, relying on the tamper-evident nature of the transaction to be verified only once, which reduced the resource overhead and enables data deduplication by the word2vec method.

Data sharing is also widely used in the field of VANET, which can improve traffic management and promote the prosperity of in-vehicle application services. Luo [23] et al. proposed an efficient collaborative data sharing method in VANET assisted by edge computing, which could effectively solve the data sharing problem between vehicles by using RSUs and vehicles as a springboard for data transmission and storage, but in the process of data transmission that could not guarantee the integrity of data. To solve the above problems, it is a good choice to introduce blockchain into data sharing in vehicular networks. Khalid et al. [24] deployed blockchain into RSUs and used IPFS to store data related to traffic events. Fan et al. [25] and Horng et al. [26] proposed a data sharing scheme in vehicular social networks with authentication and revocation mechanisms, respectively. Ma et al. [27] proposed an attribute-based encryption algorithm and maintained by RSU, which could prevent unauthorized access to plaintext data as well as malicious tampering using blockchain, and showed that it was very feasible to use blockchain to share data in the field of vehicular networking, but it still lacked the judgment of the true accuracy and repeatability of the data.

2.2. Privacy Protection. With the advent of the era of big data, privacy preservation has always been a focus of attention in academia and industry. There are many privacy-preserving techniques, including differential privacy, federated learning, and secure multiparty computation. Wei et al. [28] and Zhao et al. [29] combined differential privacy and federated learning to propose a privacy-preserving framework to protect users from privacy leakage, respectively, the former by varying the artificial noise variance to satisfy differential privacy under different protection levels and the latter by scrambling the entity-generated gradient to avoid privacy threats and reduce communication costs. Zhao et al. [30] provided a comprehensive review of secure multiparty computing, and the article mentioned secure multiparty computing including OT protocols, secret sharing, obfuscated circuits, zero-knowledge proofs, homomorphic encryption, and other methods that could solve the problem of collaborative computation of private data from multiple participants in a distributed computing scenario in a secure manner. However, due to the consideration of personal privacy, strangers in the network are often unwilling to cooperate, and users may not intend to expose their personal data to the server, even if their privacy is safely protected. In order to promote this cooperative relationship, PSI protocol has gradually become a research hotspot [31].

Wang et al. [32] proposed a tag-based verifiable outsourced PSI protocol (TVDPSI) for the multisubset case, where each subset was associated with a separate tag for data classification. Liu and Zhang [11] proposed a fine-grained profile matching scheme with multiple tags attached to a subset of users for classification to achieve fine-grained matching. Wang et al. [33] proposed an application scenario for feature matching in in-vehicle social networks using techniques such as OT protocols and PSI protocols. However, most of the current PSI protocols are based on both users and have excessive overhead. The proposed scheme in this paper allows vehicles with the same features to be classified through PSI protocols, satisfying multiple users to jointly participate in set matching and reducing the overhead. Specifically, when a vehicle sends shared data to the cloud, vehicles with the same characteristics are enabled to judge this data and are combined it with the blockchain to reach a consensus mechanism. Each step of the operation generates a block, which ensures the integrity and reliability of the shared data and mitigates conspiracy attacks by using the pooled matching mechanism.

2.3. Application of Blockchain in VANET. In recent years, with the emergence of new technologies and applications, many academic institutions and standardization organizations have conducted research on 6G-based in-vehicle communication. Guo et al. [34] analyzed the structure of future 6G in-vehicle networks and proposed some application scenarios, one of which was that by deploying blockchain at the edge nodes at the edge nodes could provide data sharing, secure authentication, and privacy protection for users. Su et al. [5] designed a secure data storage sharing scheme in a vehicular edge network using federation chain and smart contract technology. Wang et al. [6] proposed a cloud-based trust management scheme for real-time video reporting and in-vehicle messages. Ahmed et al. [35] proposed a VANET emergency messaging protocol, specifically by storing the authentication information of vehicles and the messages sent through two chains, respectively. Gao et al. [36] combined SDN and blockchain to propose a trust-based model for controlling malicious behavior in vehicular networks. Zhang et al. [37] combined blockchain with deep reinforcement learning to propose a trust model to secure the communication links between vehicles. Ortega et al. [38] proposed a content-centric network (CCN) based on blockchain and network slicing to replace the traditional client-server architecture, thus eliminating the risk of data

Table 1: Comparison of research literatures.

Research literatures	Advantages	Disadvantages
Data sharing [12–21]	Ensure the security of data storage and improve storage efficiency through various technologies.	The correctness and repeatability of the data itself cannot be guaranteed.
Privacy protection [22–26]	Through various technologies, the privacy of each participant is protected, and the information security of users is guaranteed.	The actual implementation process is too expensive and is not suitable for large-scale VANET.
Application of blockchain in VANET [5, 6, 27–31]	The sensitive information in VANET is stored in the blockchain, which improves the safety of users during driving.	The consensus process of the blockchain used is inappropriate and cumbersome.

tampering and improving the robustness of VANETs. In this paper, a high-performance VANET architecture by combining blockchain with 6G is designed, SDN and MEC technologies, where blockchain is mainly deployed into the SDN control plane, and a data chain as well as a contractual link is designed to ensure the security of the whole system, respectively.

By comparing the above literatures (Table 1), we summarize their advantages and disadvantages.

3. Preliminaries

This section focuses on our solution architecture and process as well as the final security requirements and design goals to be achieved.

3.1. System Model. This subsection gives our proposed architectural model, namely, the blockchain-based 6G-VANET secure and trusted data sharing management model.

As shown in Figure 1, the proposed architecture in this paper mainly consists of the following entities:

Vehicles: vehicles, as the main component of the SDN data plane, are equipped with a large number of sensors as well as storage and wireless communication modules, and the communication between vehicles is carried out through wireless networks. Vehicles can generate road condition related announcement messages while driving, and these announcement messages can be passed to higher levels for judgment with the help of RSU, which can be shared to all vehicles in the area. The communication technology mentioned in the architecture diagram is C-V2X, which is a cellular network-based wireless communication technology for vehicles

UAVs: UAVs can act as edge nodes to provide edge computing services to remote areas

RSUs: RSUs are equipped with processing, storage, and wireless communication modules, and the RSU acts as a bridge between the SDN controller and the vehicle

Base stations: the base stations are deployed to multiple areas based on geographic location and are equipped with MEC functions as the main component of the SDN control plane. Each base station is also equipped with a data chain and a contract chain. The function of the data chain is to store the data shared by vehicles, and the function of the contract chain is to add vehicle information and vehicle voting. The function of the MEC is to perform the corresponding set matching and block out calculation operations and provide storage resources to store the information on the blockchain

Certification authority (CA): the CA is assumed to be a fully trusted authority that distributes key information for each vehicle as a way to generate the vehicle user's address in the blockchain. If any vehicle user sends a false message during data transmission, the CA can find the real identity of the malicious user and add him/her to the blacklist

Cloud: after the edge nodes store the shared messages from vehicles, they are also sent to the cloud. Some remote areas are not covered by the SDN control area and cannot receive the messages in time; so, the final data sharing also needs to broadcast the messages to the rest of the vehicles through the cloud. Thus, vehicles can be more aware of traffic conditions and help improve the security and effectiveness of system transmission

This architecture is mainly built based on the three SDN planes, at the bottom layer is the SDN data plane, which mainly consists of vehicles and RSUs. In this layer, the vehicle is responsible for collecting data during the driving process, and the RSU acts as a bridge between the SDN controller and the vehicle. The middle layer is the SDN control plane, which mainly consists of 6G base stations and UAVs with MEC functions, and the UAV can provide data collection, edge computing, and other services for vehicle users to expand the wireless network cache [39]. This plane deploys smart contracts and blockchain and can also deploy different contracts according to different situations, reflecting the programmability of the SDN control plane. Since the smart contracts cannot be changed after they are deployed, the SDN control plane can be prevented from being hijacked. To relieve the computational pressure of a single SDN controller, this paper proposes a distributed SDN controller system, which also reflects the decentralized feature of blockchain and can effectively prevent single-point attacks. The top layer is the SDN application plane, which mainly consists of CA and the cloud. The CA can issue keys to vehicles for authentication, and the cloud can provide various services to vehicles. Our scheme uses NFV to coordinate the devices in each SDN area and break the limitation of incompatible hardware and software. Also, in order to optimize network resource allocation, network slicing techniques can be used to assign priorities to different shared data requirements, giving priority to services with high network requirements.

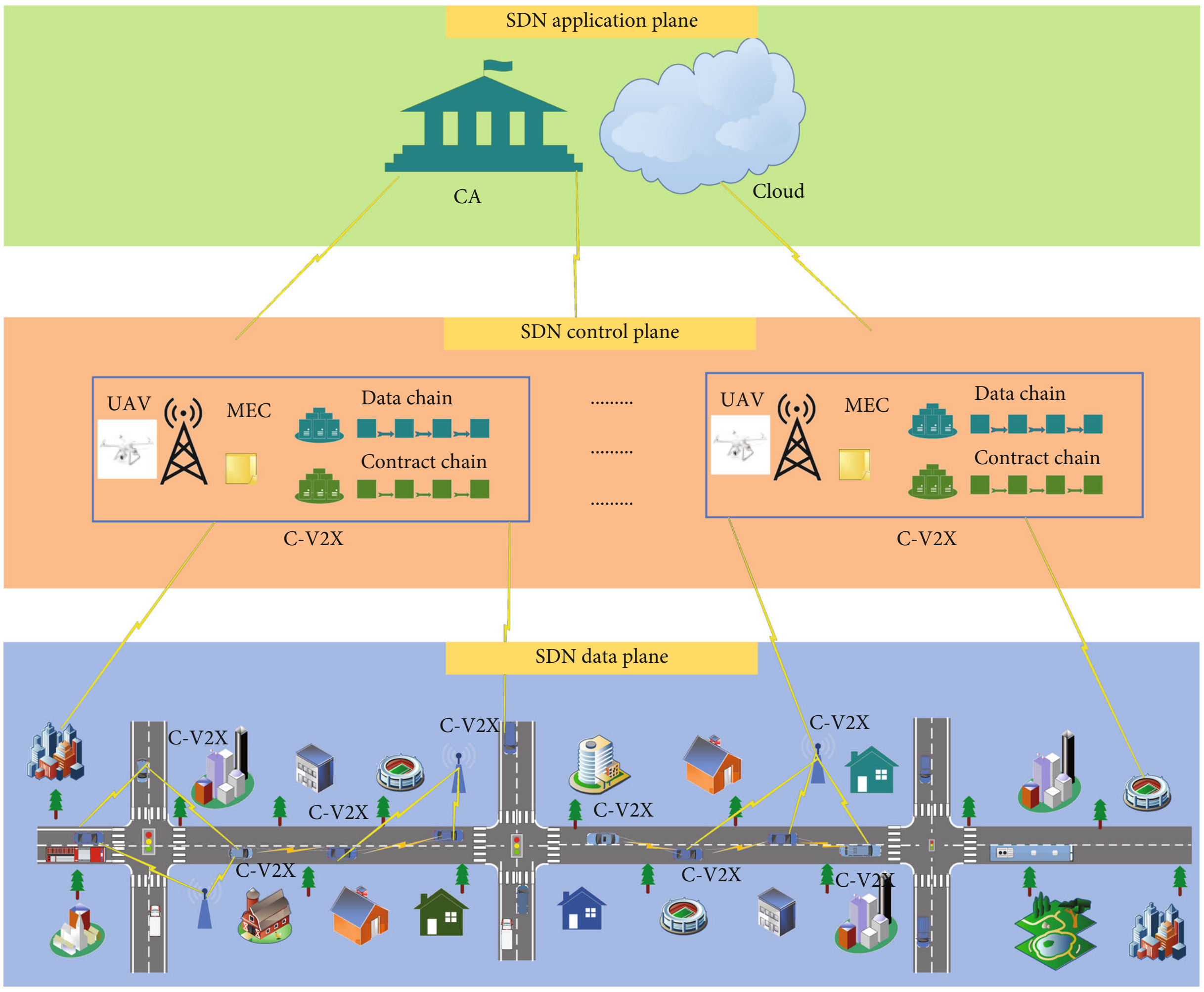


Figure 1: System architecture.

Table 2: Summary of notations.

Symbols	Description
V_i	Legitimate vehicle in the VANET
PK_{vi}, SK_{vi}	Vehicle's public key and private key
CA	Certificate authority for managing vehicle enrollment
V_p	Vehicle with providing data
V_q	Vehicles with voting rights
M_p	Data submitted by vehicle
T_x	Transactions arising from vehicles
B_s	Base station with mobile edge computing

3.2. Problem Description. In our data sharing scenario, the main entities are the legal vehicle V_i, the trusted authority CA for issuing keys, the data provider V_p, the set of voters V_q with the same set properties as V_i, and the base station B_s with MEC function, where V_p and V_q belong to V_i. It is assumed that the CA is fully trusted, the MEC can correctly match the aggregated information of the vehicle, and the CA can trace the real identity information of the user. Simply put, after V_p shares data, MEC selects V_q through PSI protocol, then V_q votes and judges the data, and finally stores the data in the blockchain for other users to access. The notations used in this paper are shown in Table 2, and the detailed process will be described in Chapter 5.

3.3. Secure Data Sharing Process with Privacy Protections and Incentives. The data sharing task process is shown in Figure 2.

(1) Vehicle registration: vehicles join the network for the first time to be issued a pair of public and private keys by the CA in the SDN application plane as the vehicle's identity

(2) Vehicle data sending: vehicles located in the SDN data plane can collect traffic-related data and send them to the SDN control plane in the form of transactions. After receiving the data shared by vehicles, the SDN control plane performs data deduplication according to the time stamp

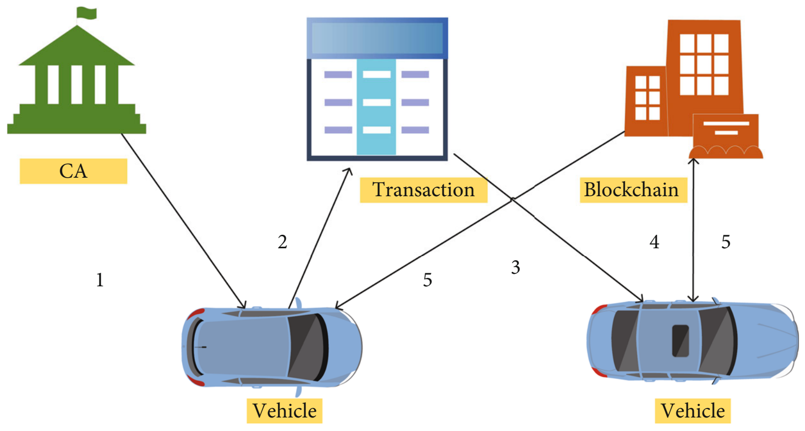


Figure 2: Task process.

(3) Voting consensus: the SDN control plane performs set matching based on the vehicle information in the current area, vehicles with the same characteristics can obtain voting rights and judge the accuracy of the data, and saves the voting records in the contract chain

(4) Packing the transaction into a block: after passing the PBFT consensus protocol, the transaction can be saved into the data chain, and then the cloud and edge nodes can broadcast the data out

(5) Issuing rewards: certain rewards should be given to those vehicles that share useful data and those that participate in voting normally

3.4. Security Requirements and Design Goals. The data sharing scheme proposed in this paper mainly includes the following security requirements and design goals.

3.4.1. Security Requirements

(1) Protecting the Privacy of Each Participant. When a user sends information to an edge node, an attacker may steal the user's information through eavesdropping attacks. Secure multiparty computing protects the sensitive information of vehicle users (e.g., driver's location and identity information) from disclosure, in which individual participants cannot access each other's information. In addition, to process these information, even if the attacker steals some information, he cannot get any useful content from it.

(2) Preventing Malicious Behavior of Edge Nodes. As an intermediary for set matching, edge nodes may obtain user information from them. Therefore, the malicious behavior of the edge node must be prevented, and the correctness of the calculation of the edge node can be verified, so as to prevent this type of man in the middle attack.

(3) Ensuring the Security of Data Storage. In the process of uploading and storing data, an attacker may attack the data storage system and tamper with the data. Therefore, the data shared by vehicle users must be traceable and tamper proof. Since the data needs to be voted on by other vehicle users in the process of storage, the voting behavior of participating users need to be constrained.

3.4.2. Design Goals

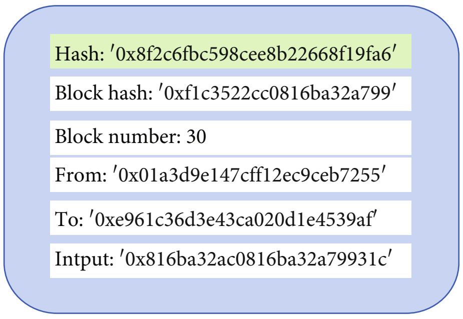
(1) Reducing the Overhead Generated in the Process of Data Storage. A suitable blockchain storage architecture should be designed a blockchain consensus block-out mechanism so that the communication overhead and storage overhead are sufficient to meet the actual requirement.

(2) Designing a PSI Protocol That Can Quickly Match the Set of Large-Scale Vehicles. The traditional PSI protocol is only applicable to two parties, but there are many vehicle users in VANET. In order to make the proposed scheme work smoothly, it is essential to design a lightweight multiparty PSI protocol.

(3) Reducing System Low Cost. The cost overhead of the whole data sharing system is simulated according to the current blockchain system, so that the system can be reasonably applied to the reality.

4. Data Sharing Scheme

In order to make the data sharing scheme run smoothly and ensure the security of data storage, we deploy the blockchain into MEC. Blockchain is a distributed database that backs up all transactions to each node. It is decentralized, tamper proof, irreversible, and traceable. It consists of a series of block links. Each block contains the current block number, the hash value of the current block, the hash value of the parent block, the hash value of the transaction, the timestamp, and other information. We have designed two chains based on Ethereum to realize the reliability of scheme storage. One is the data chain, which stores the hash value of the sent data transaction. Through this hash value, we can query the details of the vehicle shared data. The other is the contract chain, which stores the transaction information of contract deployment and contract call, from which we can query the specific situation of vehicle voting. A PBFT-based



Hash: '0x8f2c6fbc598cee8b22668f19fa6'

Block hash: '0xf1c3522cc0816ba32a799'

Block number: 30

From: '0x01a3d9e147cff12ec9ceb7255'

To: '0xe961c36d3e43ca020d1e4539af'

Intput: '0x816ba32ac0816ba32a79931c'

FIGURE 3: Data transaction information.

consensus mechanism is designed by combining the PSI protocol and smart contracts. The PSI protocol is used to select voters, and smart contracts are used to implement the voting process. The specific process includes system initialization, transaction generation, voting right vehicle selection, transaction authentication, data feedback, and incentive mechanism.

4.1. System Initialization. A new vehicleV_i joins the system for the first time to obtain a key pair PK_{vi} and SK_{vi} to prove its legitimate identity, which can be distributed by the CA located in the SDN application plane. Among them, in order to protect the security of the key, the public and private keys can be set as 64-bit hexadecimal strings by the secp256k1 elliptic curve with reference to the Ethernet ECDSA algorithm. In order to ensure the privacy of the user while not exposing the public key, the public key should be subjected to Keccak-256 hashing operation to take the last 40 bits to generate the account address of each vehicle V_i. In the process of data sharing, it is necessary to ensure that malicious vehicles can be identified and CA can reveal the identity of malicious vehicles, impose some penalties, and add them to the blacklist. But before that, a sufficient number of vehicles V_i need to authenticate the message M_p to determine if it is a malicious message.

Next is the deployment of smart contracts, which are deployed to the edge nodes B_s of each distributed SDN control plane, which are also blockchain administrator nodes. The role of the smart contract is mainly to add the information of the vehicle sending the message for other vehicles with voting rights to vote on the message and return the final number of votes to judge the accuracy of the message. When a vehicle V_p sends a message, the blockchain administrator node first loads V_p's address into the smart contract, followed by the rest of the vehicles V_q for voting judgment.

To prevent malicious users from continuously uploading data and thus wasting resources, the upload frequency of each user is also limited. For some similar announcement messages, the MEC can filter the uploaded duplicate data based on the timestamp.

4.2. Transaction Generation

4.2.1. Transactions Sent. If a vehicle V_p wants to share data M_p, M_p is first packed into a transaction T_x, and the details of T_x are shown in Figure 3.

hash indicates the current hash value of T_x; if T_x has been packed into blocks, *blockHash* and *blockNumber* will show the corresponding block information. *from* indicates who initiated the T_x, and since the transaction sender needs to unlock the account before sending the transaction, the identity of the sender can be proved here. *to* is the recipient of the T_x, which here can refer to the administrator node B_s in the SDN control plane that is responsible for broadcasting the message. *input* is the data information contained in this T_x, referring to the data information shared by the vehicle V_p, or the data information generated by the invocation of the smart contract. For a data transaction T_x the details of *input* are the hexadecimal strings of the data sent, and after the vehicle V_p sends the data V_q, it then authenticates the content of the message.

4.2.2. Data Deduplication. When a user uploads data, similar recent data may be uploaded. In order to save computational storage resources, users should compare the data with recent data by timestamps before uploading, and similar data should be discarded. Commonly used sentence similarity measures are vector space cosine similarity, which uses the cosine of the angle between two vectors in vector space as a measure of the size of the difference between two individuals. The closer the cosine value is to 1, the closer the angle is to 0 degrees, that is, the more similar the two vectors are, as follows.

Suppose there are two similar data sets: A, there was a car accident here and B, there was a traffic accident here. Take the intersection of which C: (there was a car traffic accident here). After that, A and B are compared with C and expressed in the form of a vector, with the same character as 1 and different characters as 0.At this time, $A = [1, 1, 1, 1, 0, 1, 1]$, and $B = [1, 1, 1, 0, 1, 1, 1]$. After that, the cosine formula is used to calculate the cosine values of A and B. The closer the cosine value is to 1, the more similar the two vectors are, and the two sets of data are more similar.

$$\cos\theta = \frac{\sum_{i=1}^{n}(X_i \times Y_i)}{\sqrt{\sum_{i=1}^{n}(X_i)2} \times \sqrt{\sum_{i=1}^{n}(Y_i)2}}. \tag{1}$$

The similarity between the two data sets A and B can be calculated by equation (1) as 0.83. However, the above method is a statistical-based method, and the statistical-based method cannot satisfy the semantic similarity matching. The following method is a method-based deep learning, which solves the semantic similarity matching to a certain extent.

The implementation process of judging sentence similarity through word2vec is as follows. First, the sentences are divided into words, the word2vec word vectors are trained using the Gensim library to obtain the word vectors corresponding to each word, then all the word vectors are summed and averaged to obtain the sentence vectors, and finally, the cosine values of the two sentence vectors are calculated. The similarity between A and B is 0.91 calculated by the Google open source word2vec model; so, the judgment by word2vec is obviously much better. The neural network model of word2vec is shown in Figure 4.

Among them, the input layer is a single hot code of multidimensional vector. For example, the input layer corresponding to incident in A is $A' = [0, 0, 0, 0, 1, 0]$, but in the actual training process, the dimension of the input layer will be very large. The hidden layer is a multidimensional vector weight matrix; that is, each word can be expressed as a multidimensional vector, and the output layer is the probability of similar single correspondence.

In this paper, the word2vec model is put into each edge node of the SDN control plane. Whenever a vehicle is to send data, the edge node compares this data with all the recently uploaded data and defines a threshold of size 0.8 to make reasonable trade-offs.

4.3. Voting Vehicle Selection. Once V_p has authenticated his identity, he can collect the information of V_i to start matching, and the V_i with voting rights eventually forms V_q. This process is a PSI protocol process with many vehicle users in VANET; so, a cloud-assisted PSI protocol is to be proposed.

The PSI protocol is a privacy-preserving protocol, assuming that Alice and Bob have their respective sets of attributes X and Y. Through this protocol, Alice and Bob can compute the intersection of $X \cap Y$. At the same time, this process does not reveal any information that is not in $X \cap Y$. The specific implementation of this protocol is left to the computationally powerful MEC. The specific implementation of this protocol is left to the computationally powerful MEC to perform the matching. Alice and Bob first divide their respective sets into t parts based on different attributes, where t parts are not necessarily the same, as shown in (2) and (3).

$$X_A = \chi_1 \| \chi_2 \| \cdots \| \chi_{t1}, \quad (2)$$

$$X_B = \chi_1 \| \chi_2 \| \cdots \| \chi_{t2}. \quad (3)$$

After that, MEC performs the set intersection operation between set A and set B as C, where $\chi_1 \chi_2 \, \chi_{t3} \in X_A / X_B$

$$X_C = \chi_1 \| \chi_2 \| \cdots \| \chi_{t3}. \quad (4)$$

A number of scholars have also studied cloud-assisted schemes for this protocol, Li et al. [40] proposed a PSI protocol based on homomorphic encryption, and Kerschbaum [41] proposed a one-way function PSI protocol, but the use of homomorphic encryption in vehicular networking brought pressure on computation and storage and hash functions alone may receive brute force cracking. The set is matched to the edge nodes for processing, but the edge nodes may be dishonest; so, to protect the user's privacy, the set attributes are to be fuzzed, to reduce the overhead of the whole system, in this paper, by using encryption, random number, and hash value matching for this protocol. Before each set matching, the same encryption once for each attribute in the set is performed, after which a random number is added to the ciphertext and finally, the corresponding hash value is calculated and given to the edge node for matching. Since the voters are only obtained based on the matching results, the user does not need to decrypt the set attributes, and the user only needs to know whether he or she has the right to vote, thus further protecting the privacy of each participant. The specific form of ensemble matching is as follows.

The original set X is as follows:

$$X = \chi_1 \| \chi_2 \| \cdots \| \chi_{t1}, \quad (5)$$

where $\chi_1, \chi_2 \cdots \chi_t$ is the value of the attribute in the set X. After performing an encryption operation, the set is as follows:

$$X' = \mathrm{Enc}(\chi_1) \| \mathrm{Enc}(\chi_2) \| \cdots \| \mathrm{Enc}(\chi_t). \quad (6)$$

The set the set is as follows after adding the random number λ

$$X'' = \mathrm{Enc}(\chi_1)\lambda \| \mathrm{Enc}(\chi_2)\lambda \| \cdots \| \mathrm{Enc}(\chi_t)\lambda. \quad (7)$$

Then, a hash value for each attribute is computing in the set

$$X'h = \mathrm{Hash}(\mathrm{Enc}(\chi_1)\lambda \| \mathrm{Hash}(\mathrm{Enc}(\chi_2)\lambda \cdots \| \mathrm{Hash}(\mathrm{Enc}(\chi_t)\lambda). \quad (8)$$

The random hash value of each set attribute can be obtained from (8), which can prevent violent inference attacks on set elements by untrustworthy edge nodes.

A set V_q of vehicle users that intersects with V_p is returned after set matching, and the same set elements as V_p are returned for each vehicle user in V_q. The set matching process is shown in Algorithm 1,assuming that the number of vehicles is m and the number of set elements of each vehicle is n, and the time complexity of the algorithm is $O(m^* n)$.

To avoid malicious edge nodes returning wrong set information, the final set intersection result should be verifiable. Zheng and Xu [42] and Qian et al. [43] proposed a digital signature-based set verification method, since it is not considered that the explicit text of the set intersection, but only the computational correctness here, using the OT protocol for verification. Suppose Alice and Bob first divide the intersection returned by MEC into t parts according to different attributes and arrange them in some order, when t1 = t2, then the sum is the hash value of the set attributes.

$$\begin{aligned} X_A &= \omega_1 \| \omega_2 \| \cdots \| \omega_{t1}, \\ X_B &= \varphi_1 \| \varphi_2 \| \cdots \| \varphi_{t2}. \end{aligned} \quad (9)$$

Alice then sends Bob a set S of arbitrary attribute number strings (S is the combination of any of the values 1, 2, 3,....., p any combination of values, here assume $S = 1, 3, 5$). Subsequently, Bob calculates the heterogeneous value

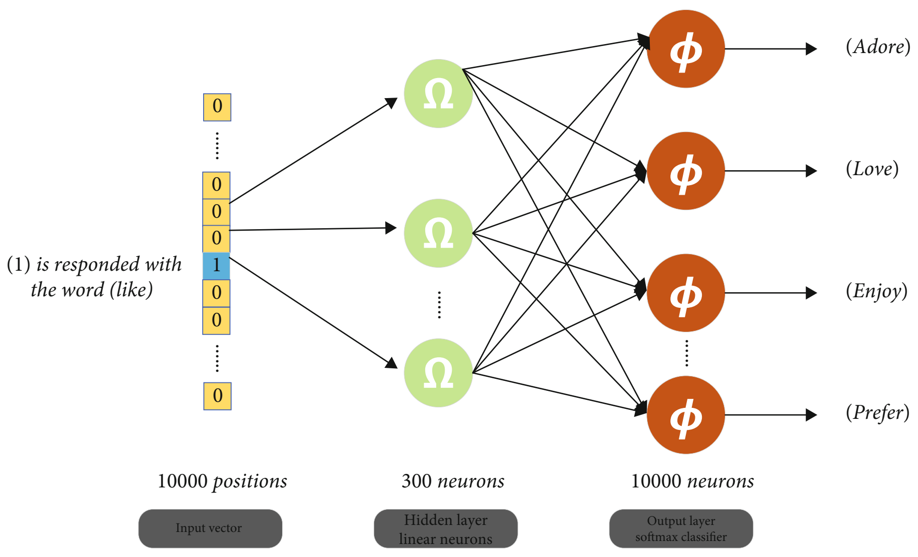

0
0
0
0
1
0
0
0
(1) is responded with the word (like)
Ω
Ω
Ω
ϕ
ϕ
ϕ
ϕ
(Adore)
(Love)
(Enjoy)
(Prefer)
10000 positions
300 neurons
10000 neurons
Input vector
Hidden layer linear neurons
Output layer softmax classifier

FIGURE 4: word2vec neural network model.

```
Input:
The collection content of $\mathbf{V_p}$;
Set B of the remaining vehicles in $\mathbf{V_i}$;
Output:
$\mathbf{V_q}$ with voting rights;
1: for each $i \in$ Bdo
2:   if $\mathbf{V_p} \cap i != \varnothing$ then
3:     Add vehicle $i$ to set $V_q$;
4:   else
5:     Vehicle $i$ has no voting rights;
6:   end if
7: end for
8: if $V_q == \varnothing$ then
9: Check the historical interactive information of $\mathbf{V_p}$ to decide whether the transaction is valid;
10: else
11:   return $V_q$;
12: end if
```

ALGORITHM 1: Calculating intersection to get the vehicles with voting rights.

of the corresponding attribute number S in his own intersection as B.

$$B = \varphi_1 \bigoplus \varphi_3 \bigoplus \varphi_5. \tag{10}$$

Alice also has to perform the corresponding calculation noted as A.

$$A = \omega_1 \bigoplus \omega_3 \bigoplus \omega_5. \tag{11}$$

From (10) and (11), the set of attributes numbered as S with different or values can be derived, if $A = B$, and then $X_A = X_B$.

In this, the SDN control plane collects all V_i messages and V_p in the current area for ensemble matching, vehicles with the same ensemble attributes are considered to the same characteristics, and these vehicles have the power to initially vote on the messages. To further mitigate the conspiracy attack, the vehicles with the final voting power are selected randomly by 50% from V_q. Since the voting process should follow the PBFT protocol, which is a protocol in which the minority obeys the majority, it is necessary to ensure that more than 2/3 of the nodes vote before subsequent operations. In some special cases it may encounter that the set matching is an empty set or the number of vehicles of V_q is less than 3. In this case, it is referred that the past interaction information of V_p to determine whether the message can be shared, and the interaction information refers to whether V_p has a record of multiple malicious votes. The interaction information records are determined as follows.

$$R = \frac{N}{M}, \tag{12}$$

where R denotes the confidence level of the current message, N denotes the record of correct V_p history voting, and M denotes the total number of V_p voting. If ≥ 0.8, and the message of the current vehicle is considered correct.

If V_p joins the vehicle network for the first time and fails to match the set of vehicles, it is initially recognized that the message sent by V_p is correct. After each vote is completed, the edge node will record the voting information, and the vehicles that vote incorrectly for many times should be added to the blacklist.

4.4. Transaction Certification. After the vehicle sends a message for the first time, the transactions generated are still pending and are not packed into blocks and need to go through the consensus mechanism and the mining operation performed by miners before the transactions can be packed into blocks. The introduction of blockchain will bring challenges to the delay overhead of the system [44]. Cryptocurrencies like Bitcoin put some restrictions on the generation of each block in order to ensure the security of the blockchain and prevent attackers from having a lot of arithmetic power to attack the various nodes of the blockchain. Most cryptocurrencies do this by finding a random number and then calculating a special form of hash value by a hashing algorithm, such as the first X bits are all zeros. However, generating a block by this form will waste a lot of computational resources, and some articles use an incentive mechanism to make cars become miners to mine but still consume a lot of time for computation in practical applications. In order to generate a block in a short time, the scheme proposed in this paper does not impose restrictions on the form of generating blocks, but a lot of verification of PBFT consensus mechanism is needed during the process of block generation; so, the security of the block chain can also be protected to some extent.

Once the set is matched, the content of the message can be voted on, and the voting process is represented as a smart contract. Before that, the information of V_p has to be loaded into the smart contract, and this process is a smart contract invocation. In order to rationalize the use of resources, the voting process is restricted in the smart contract so that each vehicle can only cast one vote for a given vehicle address at a time. Vehicle voting modifies the vote value defined in the smart contract, and this process generates a transaction packaged into a block as a record of each vehicle's voting process, further enhancing the trust of the system and alleviating the phenomenon of false voting. The smart contract voting algorithm is shown in Algorithm 2; assuming a total of n users vote, the time complexity of the algorithm is $O(n)$.

```
Input:
Transaction sent by $\mathbf{V_p}$;
$\mathbf{V_q}$ with voting rights;
        Number of $\mathbf{V_q}$ as Q;
Output:
        The number of votes P;
        Whether the transaction is valid or not;
1: for each $i \in \boldsymbol{V_q}$ do
2:   ifVehicle $i$ verified the transactionthen
3: P+=1;
4:   end if
5: end for
6: if $P \geq \mathbf{2/3}*$ Q then
7: Package transactions into blocks;
8: else
9:   Ignore this vote;
10: end if
```

Algorithm 2: Voting process.

V_q judges the M_p in T_x and adds one to the number of votes if the message is correct and finally counts the total number of votes received. After that, the PBFT protocol is followed, and the transaction is considered correct when the number of votes exceeds 2/3 of the total.

The miners mainly consist of BS with MEC function in each distributed SDN controller, which reflects the decentralized feature of blockchain. Secondly, MEC can calculate the hash value of a block in a short period of time with its powerful computing power, so that transactions can be packed into blocks and data can be shared out faster. Trusted data is propagated and replicated into the blockchain, which is maintained by decentralized edge nodes, making it accessible from anywhere. The associated computation is offloaded to the edge nodes, thus making the fully use of storage and computational resources. Once a transaction is successfully packed into a block, the blockchain administrator node can broadcast the messages in the transaction.

The transaction validation process is shown in Figure 5. When V_p sends a data transaction T_x, the blockchain administrator broadcasts the transaction to the rest of the vehicles in the SDN control domain for validation and finally obtains the validation result. When a data transaction T_x is successfully packaged into blocks, the whole blockchain system will synchronize according to the longest chain principle. This principle not only compares the length of the blockchain but also the content of the previous blockchain. If an attacker wants to change the direction of the entire blockchain, he needs to control more than 50% of the nodes in the system, which is extremely difficult to achieve in practice.

4.5. Disinformation Feedback and Incentives. Some malicious vehicles may form a group and vote for malicious false messages by pooling and matching, which can mislead the rest of the vehicles and endanger traffic safety, and messages such as road information may be outdated, such as road damage,

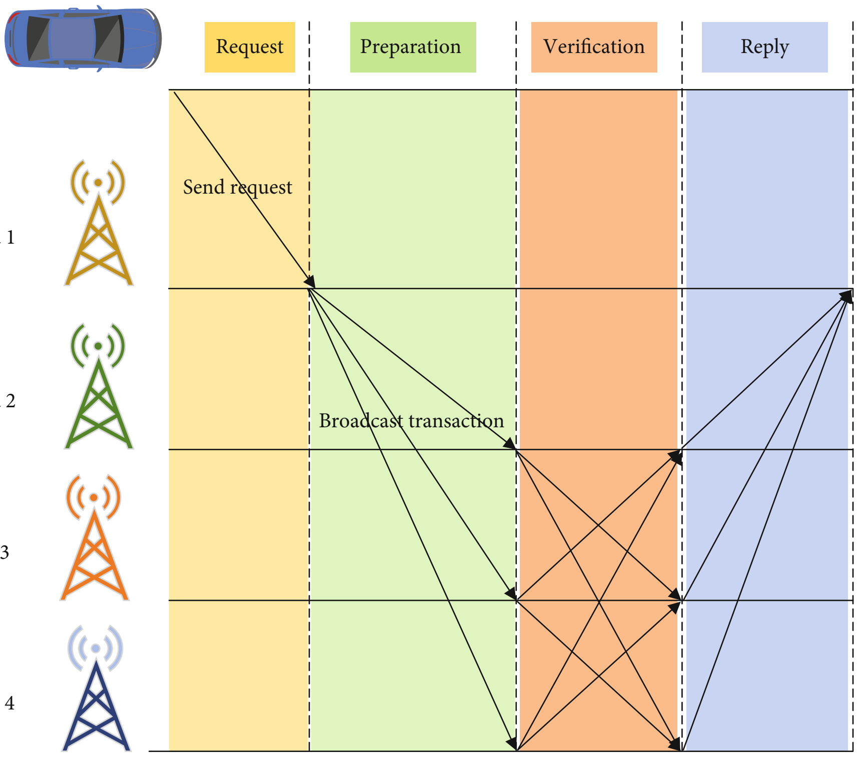
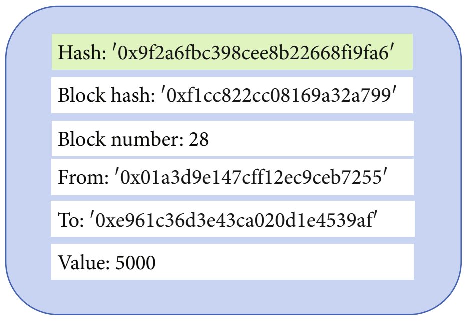


Figure 5: Transaction verification process.



Hash: '0x9f2a6fbc398cee8b22668fi9fa6'

Block hash: '0xf1cc822cc08169a32a799'

Block number: 28

From: '0x01a3d9e147cff12ec9ceb7255'

To: '0xe961c36d3e43ca020d1e4539af'

Value: 5000

Figure 6: Currency transaction information.

traffic jams caused by traffic accidents. The above two types of messages can be feedback by the rest of the vehicles to initiate a vote again and mark a response to the corresponding message after reaching a certain number of votes.

Since the process of sharing data consumes certain resources, incentives are essential in order to encourage user participation. However, designing effective incentives to encourage secure data sharing among multiple users remains a daunting task. In this paper, an incentive mechanism in the form of cryptocurrency is designed by the Ether platform, where vehicles that share correct messages and participate in voting can receive certain rewards. Blockchain transactions can send not only data but also cryptocurrency, the incentive mechanism is issued by blockchain administrator nodes, and the specific amount can be referred to the real-time Ether price. The transaction information is shown in Figure 6, where *value* indicates the amount of this transfer, and the rest of the content is similar to the data transaction information. The reward issuance process is shown in Algorithm 3; assuming that a total of N users participate in the data sharing, the time complexity of the algorithm is $O(n)$.

5. Security Analysis and Experimental Evaluation

5.1. Security Analysis. The proposed scheme in this paper uses PSI protocol to let vehicles with the same characteristics vote on messages and randomly selects a certain number of vehicles from the voting vehicles as the final vehicles with voting rights, which mitigates the conspiracy attack to a certain extent. In order to protect the privacy of each participant, the edge nodes perform set matching and then inform each vehicle separately whether it has the voting right or not, the vehicles cannot get any information from each other, and the vehicles with the voting right will get the transaction information of the shared data for the next judgment.

In addition, the scheme can prevent malicious users from performing set snapping to obtain other users' set information, because vehicle users can only perform set matching while sharing data and cannot match all the time. In order to avoid malicious edge nodes stealing users' set information or returning wrong set information, we add random numbers to the set and encrypt them, which can effectively prevent malicious edge nodes from brute force cracking. Even if the attacker steals some information, he cannot get any useful content from it and can verify the

```
Input:
Transaction sent by $\mathbf{V_p}$;
$\mathbf{V_q}$ with voting rights;
Output:
    Whether the participating vehicles can be rewarded;
1: for each $i \in V_q$ do
2:   if Vehicle $i$ correctlyverified the transactionthen
3: The administrator sends a transfer transaction to Vehicle $i$;
4:   end if
5: end for
6: if $\mathbf{V_p}$ shared the correct messagethen
7: The administrator sends a transfer transaction to $\mathbf{V_p}$;
10: end if
```

Algorithm 3: Reward process.

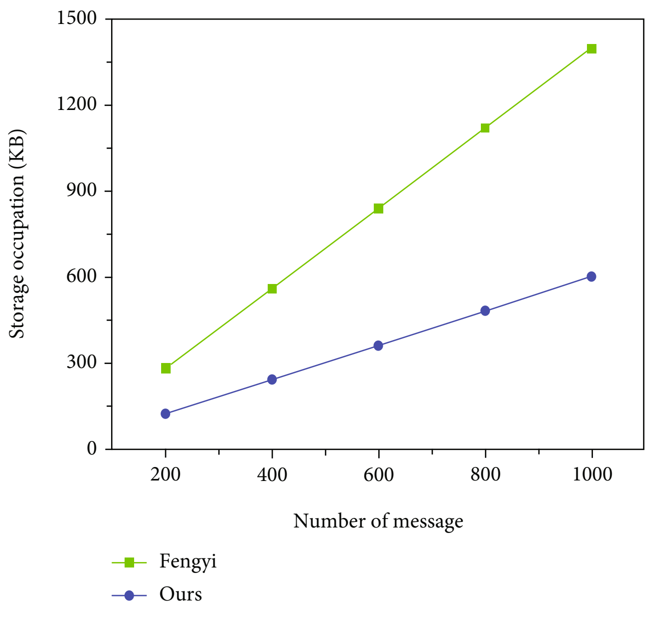


Figure 7: Storage overhead of shared data.

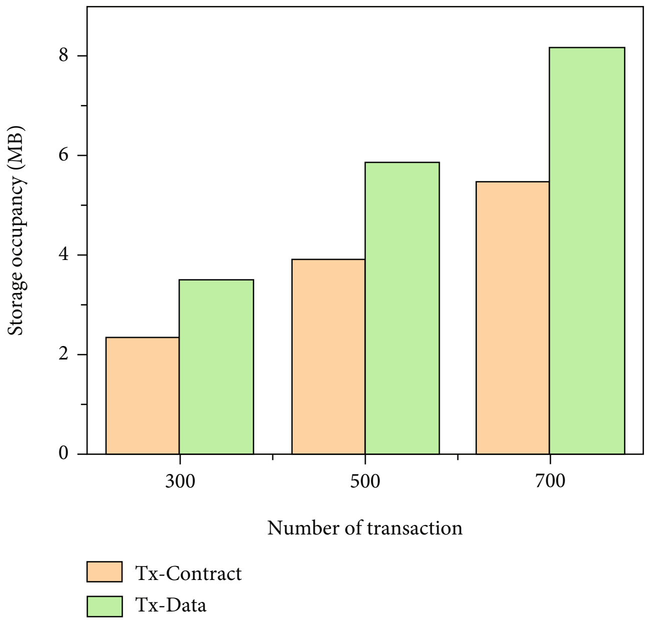


Figure 8: Storage space occupied by transactions.

results returned by the edge node with OT protocol ; so, it can effectively prevent the attacker from eavesdropping attacks and man in the middle attacks of edge nodes.

The data shared by the vehicle is loaded into a transaction, which is then packed into blocks, relying on the tamper-evident nature of the transaction and the blocks to effectively prevent the data from being maliciously tampered with, thereby protecting the storage security of the data. Each transaction contains the sending and receiving addresses of the data, which can meet the traceability of the data. In addition, the user's voting process will also generate a transaction and pack into a block, which can record the user's voting process and thus constrain the user's behavior.

5.2. *Experimental Evaluation*

5.2.1. Experimental Settings. The computer configuration for the experiment is Windows10 system AMD R7 4800H processor with 16G RAM. The simulation platform of Ethernet is Ganache2.13.2 version, and the execution script is Node 14.15.4 and Web3 0.20.1, the smart contract solidity language version 0.4.26. The experimental environment of PSI protocol is python pycryptodome Cryptographic library and hashlib module of sha256 hash algorithm.

5.2.2. Storage Overhead. Due to the introduction of blockchain in VANET, it inevitably leads to more storage usage. Zeng et al. [45] proposed a blockchain-based data sharing scheme Fengyi, but no smart contract was used; so, by comparing in terms of shared data storage, as shown in Figure 7, the memory space occupied by Fengyi for one data sharing is about 1.4 kb. As shown in Figure 7, Fengyi takes about 1.4 kb of memory space for one data sharing, while the sharing of announcement messages is targeted, eliminating part of the encryption and decryption operations, and theoretically, the storage overhead of one data

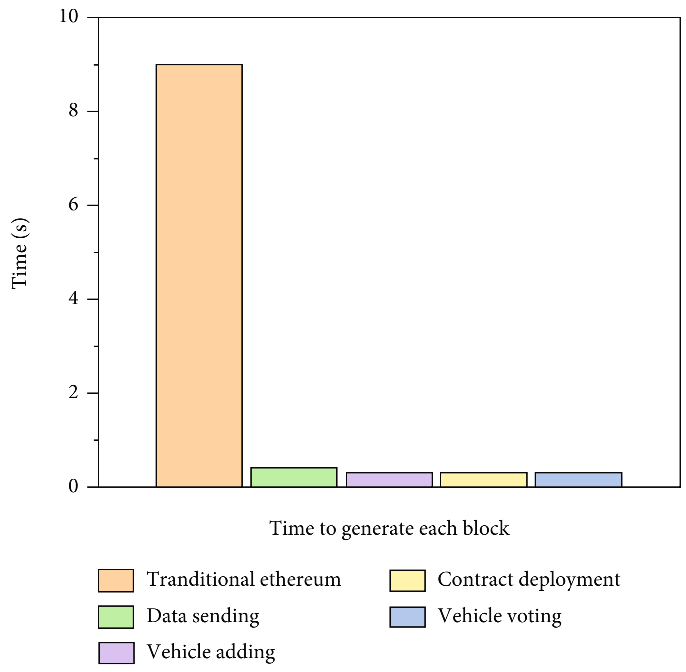


Figure 9: Block-out time comparison.

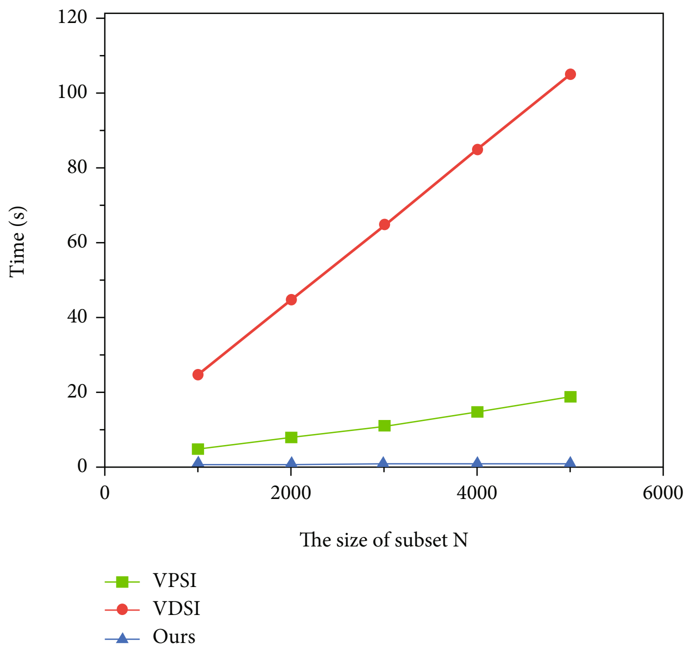


Figure 10: Time to outsource set attributes to edge nodes.

Table 3: Cost consumption.

Process execution	Gas	Eth	$
Contract deployment	926414	0.00853	15.4
Vehicle adding	138482	0.00097	1.7
Vehicle voting	1146919	0.00114	2.1
Data sending	118428	0.00057	1.0

sharing is about 0.6 kb, which is a great improvement in the data storage performance.

To further evaluate the storage overhead of the scheme proposed in this paper, the real storage overhead with Ganache-CLI is simulated, which includes the storage of the two smart contracts of adding voting vehicles and vehicle voting, as well as the transactions generated by sending data and packing the information into blocks, and filtered the block and transaction information in the output results. As

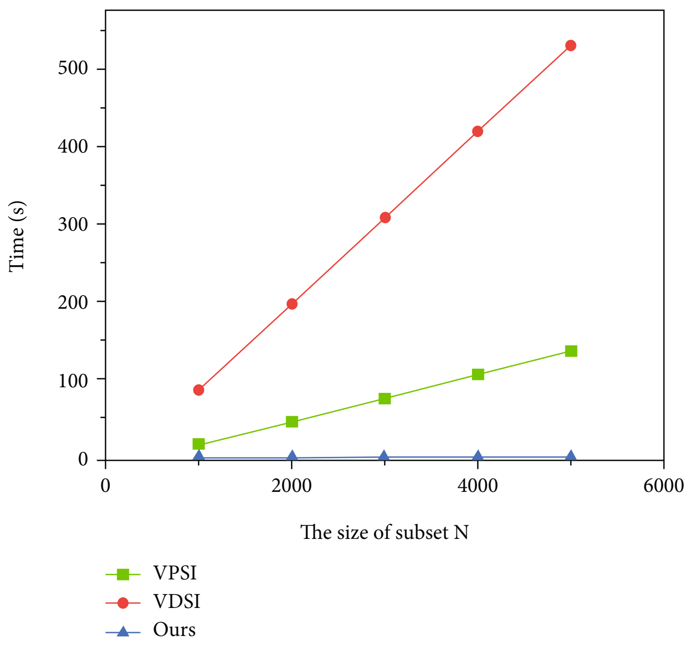


FIGURE 11: Set matching time.

seen in Figure 8, the overall storage occupancy increases linearly and is positively correlated with the number of transactions. Sending data transactions (Tx-data) in Figure 8 refers to transactions sent by vehicle shared data, and this type of transaction is directly initiated by Web3 objects, which require specifying the transaction initiator and transaction receiver as well as the transaction information. The storage overhead of this part is mainly related to the size of the data in the transaction information, and the data storage field in the experiment is set to empty, as in the storage simulation in Fengyi. The contract invocation transactions in Figure 8 refer to the contract transactions (Tx-contract) generated by adding voting vehicles and vehicle voting, which are generated differently from sending data transactions and are generated by invoking smart contracts. Since adding vehicle information and vehicle voting are essentially two methods in the same smart contract, the block storage overhead is actually the result of invoking the same smart contract, and eventually, the two transactions are tested to occupy the same amount of memory.

For the storage overhead of the vehicles themselves, our scheme is similar to the VANET data sharing scheme mentioned by Shen et al. [46], where the storage overhead of the vehicles is constant and is only the respective key pair is stored, and the only difference being that our key pair is the address of the blockchain account and a random AES key that is encrypted for each set of attributes.

5.2.3. Communication Overhead. Figure 9 shows the time taken to pack various transactions into blocks generated by traditional Ethereum, vehicle send data, add vehicle information, and vehicle voting. Unlike traditional blockchains, the block-out difficulty is not set. This is mainly because our approach does not consume as much arithmetic power as traditional blockchains, such as using the POW consensus mechanism. PBFT is used to let some nodes implement the consensus process instead of all nodes, thus greatly increasing the throughput of the whole network. As seen in Figure 9, starting from the deployment of smart contract requires about 1255 ms to complete message, which is sent to the packing into operation, but in the practical application of smart contract that needs to be deployed only once, there will be other factors (such as data transmission rate and bandwidth) and time overhead of the vehicle to subjective judgment message. So, the final time will be greater than that.

The time of outsourcing the set attributes to edge nodes is simulated by using the python pycryptodome cryptographic library and sha256 in the hashlib module. Compared with VDSI mentioned by Guo et al. [34] and VPSI mentioned by Ahmed et al. [35], digital signature is not used to verify the set matching result, but use OT protocol for verification; so, the overhead of signature generation is reduced at this stage. Since it is not needed to calculate the decryption of the set matching result, and the set matching result is only for selecting voters, a more lightweight symmetric encryption algorithm AES is used, which generates a random key for each set matching and encrypts all set attributes with this key to ensure that the result is the same after encrypting the hash of each set matching, which is convenient for the edge nodes to calculate, thus replacing the VPSI. This replaces the asymmetric encryption algorithm ElGamal used in VPSI. From Figures 10 and 11, it can be seen that the processing of the set by asymmetric encryption and digital signature methods and the matching time of the set by the cloud is in seconds, and this is only the result of two-party set operation, which does not meet the communication requirements of large-scale 6G-VANET. Our proposed method as shown in Figures 10 and 11 is millisecond in time and supports multiparty set operations, where our matching scheme is to match the encrypted hash value directly, unlike the first two which compute the cipher text, and to facilitate the display of the set matching time, 200 vehicle messages are simulated at a time, and the number of set elements is 1000-5000. The set matching time is simulated. It can be seen that MEC can select vehicle voters in a short time, which is applicable to the dynamic changes of large-scale VANET and greatly improves the efficiency of VANET. In order to enable voters to judge the accuracy of the message more realistically, a threshold value is also set for screening the number of identical elements in the intersection of matching results according to the actual situation of pooled matching.

5.2.4. Cost. The cost overhead of the system is calculated based on the units of gas consumed by transactions and smart contracts. Gas is a unit of measurement used to measure the computational power of executing transactions in the Ethereum platform. For cost analysis, several parameters are considered: (1) gas consumed for contract deployment, (2) gas consumed for adding voting vehicles, (3) gas consumed for vehicle voting, and (4) gas consumed for sending data transactions. All the above four parameters are the total amount of gas consumed from transaction generation to packing into blocks in a single time. Both gasPrice and gasLimit in ganache-CLI are default values.

Table 3 shows the cost of each implementation process. The dollar consumption in the last column refers to the Ethereum price on May 30, 2022, where $1\,\text{eth} \approx 1800$ dollars.

6. Conclusion

In this paper, we design a new data sharing scheme in 6G-VANET. Specifically, we design a new VANET architecture combining 6G, SDN, edge computing, and other technologies and then realize the effective deduplication of shared data by word2vec. Finally, two different forms of blockchain are designed to store various information needed in the process of sharing data. Among them, the vehicle layer acts as the SDN data plane to collect all kinds of shared data. MEC deployed in the SDN control plane is mainly used to calculate vehicle set matching and block producing operation of blockchain. In addition, we deployed word2vec and two blockchains into MEC, taking full advantage of MEC's computational storage resources and embodiment edge intelligence. In order to reflect the decentralized characteristics of blockchain and relieve the pressure on the SDN control plane, this paper designs the SDN control plane as distributed according to different regions, and the certification center and other service coproviders are deployed in the SDN application plane.

For judging the accuracy of the data, most current data sharing schemes are by calculating the reputation value of the data itself or the data provider, but this may be subject to some problems such as collusion attacks, network, or hardware failures. There are also some articles that use all-node voting; however, for certain announcement types of shared messages, those vehicles in different characteristics may not be able to accurately determine the authenticity of the messages. This paper designs a lightweight PSI protocol for vehicle-specific matching, in which vehicles with the same characteristics obtain voting rights. The voting process is carried out in the form of smart contracts, and each vehicle leaves a block transaction record when voting is completed, which constrains users to participate normally, and false announcement messages can be stopped from the source.

In our future work, the performance of the solution by other simulation platforms will be evaluated. In addition, more data security protection features will be considered, such as state access, and extend more application services on the blockchain platform. Finally, we will consider combining the data sharing mechanism and reputation value in this paper to create a more secure and efficient data sharing system.

Data Availability

The proposed scheme and its analysis need only theoretical and experimental support. There is no additional data set to be provided in this paper.

Conflicts of Interest

The authors declare that they have no conflicts of interest in this paper.

Acknowledgments

This work was sponsored in part by the Henan Provincial Department of Science and Technology Project (No. 212102210408) and by the Key Scientific Research Project of Henan Province (No. 22A550036).

References

[1] W. Xu, H. Zhou, N. Cheng et al., "Internet of vehicles in big data era," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 1, pp. 19–35, 2018.

[2] X. Liu and X. Zhang, "NOMA-based resource allocation for cluster-based cognitive industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5379–5388, 2019.

[3] M. S. Omar, S. A. Hassan, H. Pervaiz et al., "Multiobjective optimization in 5G hybrid networks," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1588–1597, 2017.

[4] M. Z. Chowdhury, M. Shahjalal, S. Ahmed, and Y. M. Jang, "6G wireless communication systems: applications, requirements, technologies, challenges, and research directions," *IEEE Open Journal of the Communications Society*, vol. 1, pp. 957–975, 2020.

[5] Z. Su, Y. Hui, and Q. Yang, "The next generation vehicular networks: a content-centric framework," *IEEE Wireless Communications*, vol. 24, no. 1, pp. 60–66, 2017.

[6] Z. Wang, J. Liu, C. Guo, S. Hu, Y. Wang, and X. Yang, "An efficient and secure malicious user detection scheme based on reputation mechanism for mobile crowdsensing VANET," *Wireless Communications and Mobile Computing, vol.*, vol. 2021, article 5302257, pp. 1–16, 2021.

[7] J. Kang, R. Yu, X. Huang et al., "Blockchain for secure and efficient data sharing in vehicular edge computing and networks," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4660–4670, 2018.

[8] L. Xie, Y. Ding, H. Yang, and X. Wang, "Blockchain-based secure and trustworthy internet of things in SDN-enabled 5G-VANETs," *IEEE Access*, vol. 7, pp. 56656–56666, 2019.

[9] J. He, Y. Ni, L. Cai, J. Pan, and C. Chen, "Optimal dropbox deployment algorithm for data dissemination in vehicular networks," *IEEE Transactions on Mobile Computing*, vol. 17, no. 3, pp. 632–645, 2018.

[10] Z. Wang, H. Wang, Y. Wang, and X. Yang, "CLASRM: a lightweight and secure certificateless aggregate signature scheme with revocation mechanism for 5G-enabled vehicular networks," *Wireless Communications and Mobile Computing, vol.*, vol. 2022, article 3646960, pp. 1–20, 2022.

[11] X. Liu and X. Zhang, "Rate and energy efficiency improvements for 5G-based IoT with simultaneous transfer," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 5971–5980, 2018.

[12] S. Garg, K. Kaur, S. H. Ahmed, A. Bradai, G. Kaddoum, and M. Atiquzzaman, "MobQoS: mobility-aware and QoS-driven SDN framework for autonomous vehicles," *IEEE Wireless Communications*, vol. 26, no. 4, pp. 12–20, 2019.

[13] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—a key technology towards 5G," *ETSI White Paper*, vol. 11, no. 11, pp. 1–16, 2015.

[14] Z. Ning, P. Dong, M. Wen et al., "5G-enabled UAV-to-community offloading: joint trajectory design and task scheduling," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 11, pp. 3306–3320, 2021.

[15] L. Zhang, M. Luo, J. Li et al., "Blockchain based secure data sharing system for internet of vehicles: a position paper," *Vehicular Communications*, vol. 16, pp. 85–93, 2019.

[16] Y. Qian, X. Xia, and J. Shen, "A profile matching scheme based on private set intersection for cyber-physical-social systems," in *2021 IEEE Conference on Dependable and Secure Computing (DSC)*, pp. 1–5, Aizuwakamatsu, Fukushima, Japan, 2021.

[17] Z. Wang, C. Guo, J. Liu et al., "Accurate and privacy-preserving task allocation for edge computing assisted mobile crowdsensing," *IEEE Transactions on Computational Social Systems*, vol. 9, no. 1, pp. 120–133, 2022.

[18] M. Ali, R. Dhamotharan, E. Khan et al., "SeDaSC: secure data sharing in clouds," *IEEE Systems Journal*, vol. 11, no. 2, pp. 395–404, 2017.

[19] X. Dong, R. Li, H. He, W. Zhou, Z. Xue, and H. Wu, "Secure sensitive data sharing on a big data platform," *Tsinghua Science and Technology*, vol. 20, no. 1, pp. 72–80, 2015.

[20] D.-E. Kouicem, A. Bouabdallah, and H. Lakhlef, "An efficient and anonymous blockchain-based data sharing scheme for vehicular networks," in *2020 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–6, Rennes, France, 2020.

[21] C. Feng, K. Yu, A. K. Bashir et al., "Efficient and secure data sharing for 5G flying drones: a blockchain-enabled approach," *IEEE Network*, vol. 35, no. 1, pp. 130–137, 2021.

[22] X. Zhang and X. Chen, "Data security sharing and storage based on a consortium blockchain in a vehicular ad-hoc network," *IEEE Access*, vol. 7, pp. 58241–58254, 2019.

[23] G. Luo, H. Zhou, N. Cheng et al., "Software-defined cooperative data sharing in edge computing assisted 5g-vanet," *IEEE Transactions on Mobile Computing*, vol. 20, no. 3, pp. 1212–1229, 2019.

[24] A. Khalid, M. S. Iftikhar, A. Almogren, R. Khalid, M. K. Afzal, and N. Javaid, "A blockchain based incentive provisioning scheme for traffic event validation and information storage in VANETs," *Information Processing & Management*, vol. 58, no. 2, article 102464, 2021.

[25] K. Fan, Q. Pan, K. Zhang et al., "A secure and verifiable data sharing scheme based on blockchain in vehicular social networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5826–5835, 2020.

[26] S.-J. Horng, C.-C. Lu, and W. Zhou, "An identity-based and revocable data-sharing scheme in VANETs," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 15933–15946, 2020.

[27] J. Ma, T. Li, J. Cui, Z. Ying, and J. Cheng, "Attribute-based secure announcement sharing among vehicles using blockchain," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10873–10883, 2021.

[28] K. Wei, J. Li, M. Ding et al., "Federated learning with differential privacy: algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[29] Y. Zhao, J. Zhao, M. Yang et al., "Local differential privacy-based federated learning for internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8836–8853, 2020.

[30] C. Zhao, S. Zhao, M. Zhao et al., "Secure multi-party computation: theory, practice and applications," *Information Sciences*, vol. 476, pp. 357–372, 2019.

[31] X. Wang, Z. Ning, S. Guo, M. Wen, and V. Poor, "Minimizing the age-of-critical-information: an imitation learning-based scheduling approach under partial observations," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[32] Q. Wang, F. Zhou, J. Xu, and S. Peng, "Tag-based verifiable delegated set intersection over outsourced private datasets," *IEEE Transactions on Cloud Computing*, vol. 10, no. 2, pp. 1201–1214, 2020.

[33] X. Wang, X. Kuang, J. Li, J. Li, X. Chen, and Z. Liu, "Oblivious transfer for privacy-preserving in VANET's feature matching," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4359–4366, 2020.

[34] H. Guo, X. Zhou, J. Liu, and Y. Zhang, "Vehicular intelligence in 6G: networking, communications, and computing," *Vehicular Communications*, vol. 33, p. 100399, 2022.

[35] M. Ahmed, N. Moustafa, A. F. M. S. Akhter et al., "A blockchain-based emergency message transmission protocol for cooperative VANET," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2021.

[36] J. Gao, K. O.-B. O. Agyekum, E. B. Sifah et al., "A blockchain-SDN-enabled internet of vehicles environment for fog computing and 5G networks," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4278–4291, 2019.

[37] D. Zhang, F. R. Yu, and R. Yang, "Blockchain-based multi-access edge computing for future vehicular networks: a deep compressed neural network approach," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–15, 2021.

[38] V. Ortega, F. Bouchmal, and J. F. Monserrat, "Trusted 5G vehicular networks: blockchains and content-centric networking," *IEEE Vehicular Technology Magazine*, vol. 13, no. 2, pp. 121–127, 2018.

[39] X. Wang, Z. Ning, S. Guo, M. Wen, L. Guo, and V. Poor, "Dynamic UAV deployment for differentiated services: a multi-agent imitation learning based approach," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[40] S. Li, S. Zhou, Y. Guo, J. Dou, and D. Wang, "Secure set computing in cloud environment," *Journal of Software*, vol. 27, no. 6, pp. 1549–1565, 2016.

[41] F. Kerschbaum, "Collusion-resistant outsourcing of private set intersection," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pp. 1451–1456, Trento, Italy, 2012.

[42] Q. Zheng and S. Xu, "Verifiable delegated set intersection operations on outsourced encrypted data," in *2015 IEEE International Conference on Cloud Engineering*, pp. 175–184, Tempe, AZ, USA, 2015.

[43] Y. Qian, J. Shen, P. Vijayakumar, and P. K. Sharma, "Profile matching for IoMT: a verifiable private set intersection scheme," *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 10, pp. 3794–3803, 2021.

[44] Z. Ning, S. Sun, X. Wang et al., "Blockchain-enabled intelligent transportation systems: a distributed crowdsensing framework," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[45] C. Zeng, Y. Wang, F. Liang, and X. Peng, "Fengyi: trusted data sharing in VANETs with blockchain," in *2020 IEEE 25th Pacific Rim International Symposium on Dependable Computing (PRDC)*, pp. 11–20, Perth, WA, Australia, 2020.

[46] J. Shen, T. Zhou, J. Lai, P. Li, and S. Moh, "Secure and efficient data sharing in dynamic vehicular networks," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8208–8217, 2020.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 4621440, 12 pages
https://doi.org/10.1155/2022/4621440

Research Article

Joint Radio Map Construction and Dissemination in MEC Networks: A Deep Reinforcement Learning Approach

Xingguang Liu, Li Zhou, Xiaoying Zhang, Xiang Tan, and Jibo Wei

College of Electronic Science and Technology, National University of Defense Technology, Changsha, Hunan 410073, China

Correspondence should be addressed to Li Zhou; zhouli2035@nudt.edu.cn

Received 16 May 2022; Accepted 29 June 2022; Published 19 July 2022

Academic Editor: Xiangjie Kong

With the development of 6G, the rapidly increasing number of smart devices deployed in the Industrial Internet of Things (IIoT) environment has been witnessed. The radio environment is showing a trend of complexity, and spectrum conflicts are becoming increasingly acute. User equipment (UE) can accurately sense and utilize spectrum resources through radio map (RM). However, the construction and dissemination of RM incur a heavy computational burden and large dissemination delay, which limit the real-time sensing of spatial spectrum situations. In this paper, we propose an RM construction and dissemination method based on deep reinforcement learning (DRL) in the context of mobile edge computing (MEC) networks. We formulate the dissemination modes selection and resource allocation problems during RM construction and dissemination as a mixed-integer nonlinear programming problem. Then, we propose an actor-critic-based joint offloading and resource allocation (ACJORA) algorithm for intelligent scheduling of computational offloading and resource allocation. We design a novel weighted loss function for the actor network, which combines the discrete actions for offloading decisions and the continuous actions for resource allocation. And the simulation results show that the proposed algorithm can reduce the cost of dissemination by optimizing the offloading strategies and resources, which is more applicable for real-time RM applications in MEC networks.

1. Introduction

With the development of 6G and the rapid growth of mobile data traffic, new business scenarios are constantly emerging. The Industrial Internet of Things (IIoT) is expected to be a crucial technology changing the manufacturing way [1–3]. IIoT is a variety of acquisition or controllers with sensing and monitoring capabilities. And it integrates mobile communication, intelligent analysis, and other technologies into all aspects of the industrial production process, thereby greatly improving manufacturing efficiency and realizing the intelligence of traditional industries. However, with the explosion of IIoT applications, the radio environment has become increasingly complex, which brings unparalleled challenges such as scarce spectrum resources, intermittent wireless connections, and high propagation delays. Further research is needed to address the above issues.

Radio map (RM) is an important tool for understanding radio environments and analyzing network performance. It incorporates geographic information to describe the radio environment from multiple dimensions such as time, frequency, space, and power [4]. RM can not only effectively acquire the distribution of the radio spectrum resources, but also utilize the multidimensional spectrum data and manage the spectrum resources in a straightforward and flexible way. It has been widely used in cognitive radio [5–7], interference management [8], coverage analysis [9–11], and active resource allocation [12–14]. The spectrum data can be collected by interconnected sensors or smart devices, and the spectrum data needs to be further processed to be constructed as an RM. In the traditional cloud-based network architecture, all spectrum data must be uploaded to a centralized cloud server, constructed as an RM, and disseminated to user equipment (UE). Due to the large size of RM, the traditional dissemination scheme from cloud server to UE consumes more bandwidth and time, which cannot meet the low-latency requirements of IIoT.

In recent years, edge intelligence has integrated edge computing and artificial intelligence (AI) technologies to effectively promote edge-end collaboration [15]. In mobile edge computing (MEC) systems, various services originally deployed on the central cloud server can be deployed on MEC servers, which fundamentally shortens the data transmission delay [16]. At the same time, AI technology represented by reinforcement learning (RL) can schedule the computation offloading and resource allocation in MEC networks [17–19], which improves the efficiency of edge computing. The authors in [20] explored the joint optimization of computational offloading and resource allocation in dynamic multiuser MEC systems and proposed a Q-learning-based method and a double deep Q-networks- (DDQN-) based method to determine joint strategies for computational offloading and resource allocation. [21] considers both the multiuser computation offloading and edge server deployment in an unmanned aerial vehicle- (UAV-) enabled MEC network. The authors proposed two learning algorithms to minimize the system-wide computation cost under a dynamic environment. To solve the joint optimization of computing offloading and service caching in the edge computing-based smart grid, the authors in [22] proposed a gradient descent allocation algorithm to determine the computing resource allocation strategy, and an algorithm based on game theory to determine the computing strategy. The authors in [23] proposed a method of saving the content service provider (CSP) based on a method of motivating drivers and deep Q-networks (DQN). [24] proposed an auction algorithm and a dynamic task admission algorithm to maximize the system average throughput in a 5G-enabled UAV-to-community offloading system. [25] proposed a deep reinforcement learning (DRL) additional particle swarm optimization algorithm to maximize the long-term utility of all mobile devices in the MEC-based mobile blockchain framework, which takes into account the limited bandwidth and computing power of small base stations. Edge intelligence technology empowers edge users with more powerful information processing and content delivery capabilities by scheduling computing, storage, and other resources for users. It profoundly changes the function of mobile applications and the utilization mode of network resources. What is more, it provides inspiration for constructing and disseminating RM with computational complexity, high bandwidth, and delay-sensitive requirements.

In this paper, we decompose the RM construction task into two subtasks, which are deployed in the MEC server and UEs according to offloading modes. In MEC networks, RMs are compressed before transmission in order to reduce bandwidth consumption. Then, we propose an actor-critic-based joint offloading and resource allocation (ACJORA) algorithm for intelligent scheduling of computation offloading and resource allocation in MEC networks. Our principal contributions are summarized as follows:

(1) We propose three modes of disseminating RMs in MEC networks and formulate the process as a mixed-integer nonlinear programming (MINLP) problem. Our objective is to minimize the energy consumption and delay of RM construction and dissemination

(2) To solve the above problem, we propose a DRL algorithm based on actor-critic for joint computation offloading and resource allocation. Considering the offloading decision actions are discrete and resource allocation actions are continuous, we design a weighted loss function including the two types of actions in one actor network, which significantly reduces the number of training parameters and improves the convergence efficiency of the algorithm

(3) Simulation experiments prove that the proposed ACJORA algorithm can find offloading and resource allocation strategies for RM dissemination effectively, which is more applicable for real-time RM applications

The remaining sections of this paper are organized as follows. Section 2 reviews the related work on the problem. Then, the network model and problem formulations are introduced in Section 3. Section 4 specifies the implementation details of our ACJORA algorithm. Performance evaluations are provided in Section 5, and Section 6 concludes the paper.

2. Related Work

2.1. Construction of RM. Accurate RM can provide better services, and the ways to improve the accuracy of RM mainly include optimizing the deployment of sensor devices and improving the accuracy of spatial interpolation [26, 27]. However, in some cases, sensor devices are predeployed and the deployment may not be optimal [28]. What is more, it is an uneconomical way to increase the number of sensor devices. Therefore, the interpolation accuracy needs to be improved under the limit of the number of sensors [29]. However, the construction complexity also increases as the accuracy increases. At present, RM is mainly constructed in the central cloud server, which is used for network planning or spectrum management and control in advance or for a long period. Some research has been carried out to reduce the complexity of RM construction. The authors in [30] proposed a method based on the Kalman filter. The work in [31] proposed a method based on regression kriging and incremental clustering. Both works of [30, 31] reduced the complexity of RM construction. [32] proposed RM construction method based on a superresolution (SR) algorithm which greatly shortens the construction time while improving the accuracy. This algorithm contains two phases: offline training and online conversion. In the offline training phase, RM images with different interpolation resolutions are used to train the return forest model with the best parameters. In the online conversion phase, the trained model can directly convert LR RM to high-resolution (HR) RM. In addition, the dissemination of accurate RM requires the scheduling of computing and communication resources, and edge intelligence provides us with solutions.

2.2. Edge Intelligence-Based IIoT. There has been a lot of research on edge intelligence in recent years, and some are used to solve problems in IIoT. The authors in [33] make a review of research results that expounds on the development and convergence process of IIoT and edge computing. They propose an architecture for edge computing in IIoT and comprehensively explain it from multiple performance metrics. The authors in [34] considered the energy cost optimization problem of computing and caching in the Internet of Vehicles and integrated a deep deterministic policy gradient (DDPG) algorithm to solve this problem. [35] constructed a blockchain-enabled crowdsensing framework in intelligent transportation system. The authors proposed a DRL-based algorithm and a distributed alternating direction method of multipliers algorithm for distributed traffic management. [36] proposes a novel framework for optimizing edge collaborative network (ECN) to improve the stability between edge devices and the performance of edge computing tasks. [37] proposed a multiagent imitation learning-enabled UAV research deployment approach, which enables different UAV owners to provide services with differentiated service capabilities in a shared area. In a survey paper [38], the authors explored the emerging opportunities brought by 6G technologies in IoT networks and applications. They shed light on some 6G technologies that are expected to empower future IIoT networks, including edge intelligence, massive ultrareliable and low-latency communications, and blockchain.

2.3. Video Streaming Based on Edge Intelligence. Since the transmission of video streams is time-sensitive, and many video streaming tools (i.e. smartphones and VR devices) are limited by energy and computational capacity, just like the case we proposed. There are some research on video streaming transmission. The authors in [39] used a DRL algorithm to simultaneously optimize energy consumption and quality of service (QoS) for users during video streaming in edge networks. [40] proposed a peer-to-peer video streams transfer method based on MEC, which can perceive QoS for different users. The author modeled the transmission, computation, and offloading problem of video streams as a problem of maximizing QoS for users and then implemented an anti-fuzzy particle swarm optimization algorithm to optimize it. Distributed edge computing was used to optimize bandwidth consumption during video streaming [41]. The above methods of video streaming transmitting in edge intelligent networks mainly focused on computation offloading and resource allocation. However, RM dissemination needs to consider the construction and dissemination of RM at the same time. It is necessary to optimize the construction algorithms and compression coding of RM to reduce the use of computing and communication resources. Therefore, further research is needed on the construction and dissemination of RM.

3. System Model and Problem Formulation

We consider a MEC network with a base station (BS), a MEC server, and N UEs, as shown in Figure 1. The set of UEs is denoted by $\mathcal{N} = \{1, 2, \cdots, N\}$. UEs have radio receivers that collect spectrum data 5 times/sec based on crowdsensing without deploying radio receivers additionally. They send spectrum and location information to the BS with newly collected spectrum data.

We adopt the RM construction method based on the SR model in [32], which was trained by RM images with different interpolation resolutions. Due to the small size of the trained SR model, it consumes less bandwidth for transmission. The construction task of RM can be divided into two subtasks, e.g., the kriging interpolation algorithm and the SR model. The kriging interpolation algorithm can construct spectrum data as low-resolution (LR) RMs, and SR models convert LR RMs into HR RMs. In order to reduce time and energy consumption, the task of RM construction is decomposed and offloaded to the MEC server and UE. Therefore, it can transform the task of disseminating RMs into the task of disseminating mode with a small amount of data. The MEC server can determine whether it is necessary to offload RM construction task to the UE due to computation resources and bandwidth resources in the downlink. The dissemination mode decision variable of UE i can be denoted as $m_i \in \{0, 1, 2\}$. The three dissemination modes are described as follows:

(a) All server (mode 0, $m_i = 0$): in this mode, the construction task of RM only occurs on MEC server. First, the edge server completes the construction of the HR RM. Then, the HR RM is compressed and sent to UEs. This mode is suitable for situations in which the bandwidth of the dissemination link is abundant or the processing power of UE i is limited

(b) Partial offloading (mode 1, $m_i = 1$): in this mode, UE i needs to execute part of RM construction task. First, the edge server completes the construction of the LR RM and the training of the SR model. Then, the edge server disseminates the compressed LR RM and SR model to UE i, and UE i only needs to complete the SR conversion task. This mode is suitable for situations in which the dissemination link bandwidth is abundant and UE i has a certain processing capability

(c) All local (mode 2, $m_i = 2$): in this mode, MEC server disseminates the raw spectrum data of all UEs and trained SR model to UE i. UE i constructs the raw spectrum data into an LR RM. Then, it is transformed into an HR RM by the SR model. This mode is suitable for situations in which the bandwidth of the dissemination link is limited or UE i has a certain processing capability

We denote the RM construction computation task of UE i as $\tau_i = \{s_i, c_i, T_i^{\max}\}$. Here, s_i expresses the size of computation input data, c_i represents the number of CPU cycles required to accomplish the computation task, and $T_i^{\max}$ is the maximum tolerant delay of the task. In our model, computation tasks are considered to be decomposable. As a result, we decompose the construction task of RM into two subtasks, LR RM construction, and SR transformation. In addition, there is also a process of compression when MEC server disseminates RM to UE in mode 0 and mode 1. Table 1 shows the computation tasks in MEC networks.

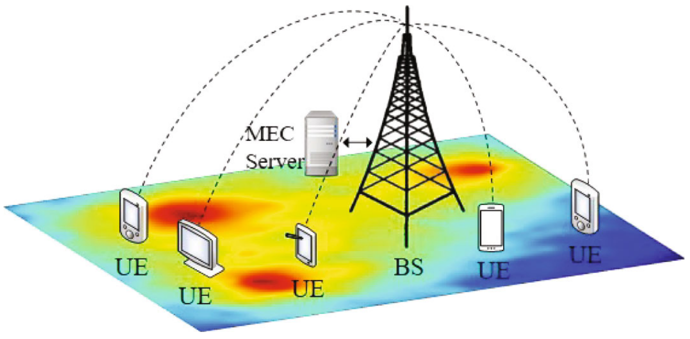


Figure 1: Network model.

Table 1: Computation tasks in MEC networks.

Computation tasks	Input data	CPU cycles
Spectrum data→LR RM	$s_{i,1}$	$c_{i,1}$
LR RM→HR RM	$s_{i,2}$	$c_{i,2}$
RM compression	$s_{i,3}$	$c_{i,3}$

3.1. Edge Server Execution Model. We denote the computational capacity (i.e., CPU cycles per second) of the edge server as F. And the computational capacity allocated by the edge server to UE i is F_i. The energy consumption of the edge server is calculated as

$$e_i^{\text{server}} = kF_i^2 c_i^{\text{server}}, \tag{1}$$

where $k = 10^{-27}$ is the effective switched capacitance of the CPU, determined by the CPU hardware architecture. c_i^{server} is the number of CPU cycles of computation tasks of UE i executed on the edge server, which is determined by the dissemination mode. (1) Mode 0: MEC server executes the computation tasks of constructing the spectrum data into an LR RM, constructing the LR RM into an HR RM, and compression of the RM. So, the number of CPU cycles of computation tasks executed by the edge server is $c_i^{\text{server}} = c_{i,1} + c_{i,2} + c_{i,3}$. (2) Mode 1: MEC server executes the computation task of constructing the spectrum data into an LR RM and compression of RM, $c_i^{\text{server}} = c_{i,1} + c_{i,3}$. (3) Mode 2: MEC server does not execute the computation task, $c_i^{\text{server}} = 0$. The time for edge server to execute the computation task can be expressed as

$$t_i^{\text{server}} = \frac{c_i^{\text{server}}}{F_i}. \tag{2}$$

3.2. Cognitive User Execution Model. We denote the computational capacity of UE i as f_i. The computation energy consumption of UE i is calculated as

$$e_i^{\text{local}} = kf_i^2 c_i^{\text{local}}, \tag{3}$$

where c_i^{local} is the number of CPU cycles of computation tasks executed by UE i, which is determined by the dissemination mode. (1) Mode 0: UE does not execute the computation task of RM, $c_i^{\text{local}} = 0$. (2) Mode 1: UE executes the computation task of SR model transformation, $c_i^{\text{local}} = c_{i,2}$. (3) Mode 2: UE executes the computation tasks of LR RM construction and SR model transformation, $c_i^{\text{local}} = c_{i,1} + c_{i,2}$. The time for edge server to execute the computation task can be expressed as

$$t_i^{\text{local}} = \frac{c_i^{\text{local}}}{f_i}. \tag{4}$$

3.3. Communication Model. The total communication bandwidth of the edge network is W, and the bandwidth allocated to UE i is W_i. Hence, the downlink transmission rate of UE i is calculated as

$$r_i = W_i \log_2\left(1 + \frac{ph_i}{\sigma^2}\right), \tag{5}$$

where p denotes the transmit power of the base station which is constant. h_i expresses the channel gain between UE i and the base station. σ^2 represents the noise power. The data transmission delay can be represented as

$$t_i^{\text{trans}} = \frac{d_i}{r_i}, \tag{6}$$

where d_i denotes the size of downlink transmission data between BS and UE i, which is determined by the dissemination mode. (1) Mode 0: MEC server transmits the HR RM to UE i, so the size of transmission data is denoted as $d_i = d_i^{\text{mode0}}$. (2) Mode 1: the edge server transmits the LR RM to UE i, so the size of transmission data is denoted as $d_i = d_i^{\text{mode1}}$. (3) Mode 2: the edge server transmits the raw spectrum data and trained SR model to UE i, so the size of transmission data is denoted as $d_i = d_i^{\text{mode2}}$. The data of computation and communication in different modes are shown in Table 2.

The transmission energy consumption of the data transmitted by MEC server to UE i can be calculated as

$$e_i^{\text{trans}} = pt_i^{\text{trans}}. \tag{7}$$

Table 2: The data of computation and communication in different modes.

Offloading modes	Computing data at edge server	Computing data at UE i	Transmission data
Mode 0	$c_i^{\text{server}} = c_{i,1} + c_{i,2} + c_{i,3}$	$c_i^{\text{local}} = 0$	d_i^{mode0}
Mode 1	$c_i^{\text{server}} = c_{i,1} + c_{i,3}$	$c_i^{\text{local}} = c_{i,2}$	d_i^{mode1}
Mode 2	$c_i^{\text{server}} = 0$	$c_i^{\text{local}} = c_{i,1} + c_{i,2}$	d_i^{mode2}

3.4. Problem Formulation. The total energy consumption of RM construction and dissemination for UE i can be calculated as

$$E_i = w_{\text{server}}^e e_i^{\text{server}} + w_{\text{local}}^e e_i^{\text{local}} + w_{\text{trans}}^e e_i^{\text{trans}}, \tag{8}$$

where w_{server}^e, w_{local}^e, and w_{trans}^e are the energy consumption weights for edge server execution, UE execution, and communication, respectively.

The total time spent on computation and communication can be calculated as

$$T_i = w_{\text{server}}^t t_i^{\text{server}} + w_{\text{local}}^t t_i^{\text{local}} + w_{\text{trans}}^t t_i^{\text{trans}}, \tag{9}$$

where w_{server}^t, w_{local}^t, and w_{trans}^t are the execution delay weights for edge server execution, UE execution, and communication, respectively.

In order to minimize the sum cost of execution delay and energy consumption for RM construction and dissemination, we formulate the weighted sum of energy and delay as the total consumption of the MEC system. Under the constraint of computation and bandwidth capacity and maximum tolerable delay, the problem can be optimized as follows:

$$\begin{aligned} &\min_{m,W,F} \sum_{i=1}^{n} (1-\varpi)E_i + \varpi T_i \\ &\text{s.t.} C1 : m_i \in \{0,1,2\} \\ &C2 : k f_i^{\,2} c_i^{\text{local}} \le e_i^{\text{local}} \\ &C3 : \sum_i^N F_i \le F_{\max} \\ &C4 : \sum_i^N W_i \le W_{\max} \\ &C5 : 0 \le T_i^{\text{sum}} \le T_{\max}. \end{aligned} \tag{10}$$

In the above problem, ϖ is the weight for execution delay. $m = (m_1, m_2, \cdots, m_n)$ is the offloading decision vector, $W = (W_1, W_2, \cdots, W_n)$ is the bandwidth allocation, and $F = (F_1, F_2, \cdots, F_n)$ is the computation resource allocation of edge server. Besides, $C1$ represents the offloading mode of UE i. $C2$ expresses the energy consumption of UE i that does not exceed its remaining energy. $C3$ indicates the sum of computation resources allocated to all UEs that cannot exceed the computation capacity of MEC server. $C4$ expresses that the sum of the bandwidth allocated to all UEs cannot exceed the available bandwidth of MEC network. $C5$ represents that the sum time for RM construction and dissemination does not exceed the tolerance time of the task. T_i^{sum} denotes the total time of RM construction and dissemination.

$$T_i^{\text{sum}} = t_i^{\text{server}} + t_i^{\text{local}} + t_i^{\text{trans}}. \tag{11}$$

Note that the offloading decision variables and the resource allocation variables correspond to integer variables and continuous variables, respectively. Therefore, it is an MINLP and NP-hard problem for the objective function, which has no convex feasible set. And the complexity of the feasible set grows exponentially with the number of UEs. Since traditional model-based methods are incapable of dealing with dynamic scenarios, we adopt a DRL approach, which is model-free.

4. DRL-Based Joint Offloading and Resource Allocation Algorithm

According to the optimization objectives and constraints of the problem, we solve it with an ACJORA algorithm. This section first defines the state space, action space, and reward function of the model. Then, we introduce the proposed actor-critic algorithm framework in detail.

4.1. State Space, Action Space, and Reward Function. According to the system model, the state space, action space, and reward function are defined as follows.

4.1.1. State Space. The state space can be presented by

$$S = \left\{ s_t \middle| s_t = \left(e_t^{\text{local}}, F_t^{\text{remain}}, W_t^{\text{remain}} \right), t \in M \right\}, \tag{12}$$

where s_t denotes the network state at step t. The available resource state at t is represented as $F_t^{\text{remain}} = F_{\max} - \sum_i^n F_i$ and $W_t^{\text{remain}} = W_{\max} - \sum_i^n W_i$. F_t^{remain} expresses the available computation resources of the MEC server, and W_t^{remain} denotes the available communication bandwidth resource of the MEC network. The purpose of observing them is to ensure meet the constraints of computational capacity and communication channel capacity. In addition, we also needs to observe the remaining energy of UEs e_t^{local} to avoid that the energy of UE is not enough to complete the computation tasks allocated in the next period.

4.1.2. Action Space. The action space can be denoted as

$$A = \{ a_t | a_t = (m_t, F_t, W_t), t \in M \}, \tag{13}$$

which consists of three vectors: offload decision vector $m_t = (m_1^t, m_2^t, \cdots, m_n^t)$, computation resource allocation vector $F_t = (F_1^t, F_2^t, \cdots, F_n^t)$, and spectrum resource allocation vector $W_t = (W_1^t, W_2^t, \cdots, W_n^t)$. In a MEC system network, MEC server disseminates offload strategies to UEs. Meanwhile, the computation and communication resources allocated to UEs should also be determined.

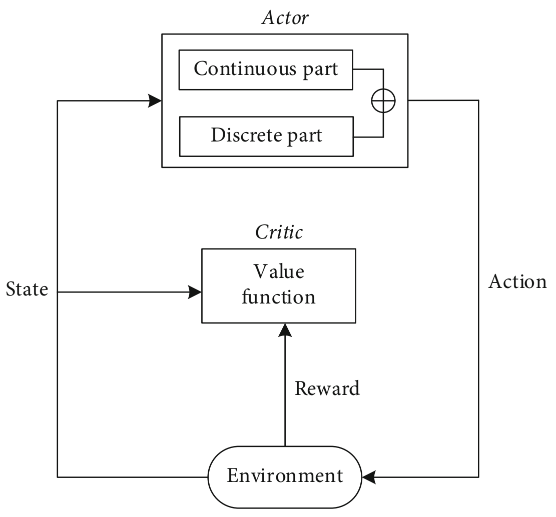


Figure 2: The actor-critic algorithm framework.

4.1.3. Reward Function. The immediate reward function is generally related to the objective function. In Equation (10), our goal is to obtain the smallest sum cost of energy and time consumption, while the goal of reinforcement learning is to obtain the largest reward. Therefore, the value of the reward needs to be negatively related to the value of the sum cost. The sum cost of the system at time t is denoted as cost_t. Thus, the immediate reward obtained by executing policy a_t in state s_t can be defined as

$$r_t = \text{-cost}_t. \tag{14}$$

4.2. DRL-Based Algorithm. Basically, reinforcement learning algorithms can be classified into three types: actor-only, critic-only and actor-critic [42]. Actor-only methods employ policy functions (i.e., policy gradient methods) to learn stochastic policies efficiently for models with large action spaces and converge asymptotically to local optima, which is more feasible for the models with continuous actions. However, they often cause high variance in expected reward estimates and slow learning. Critic-only methods with value functions (i.e., action-value methods) typically use time difference (TD) iterations and thus have lower variance in expected reward estimates. However, they need to use optimizers in each state encountered to find an action with the highest expected rewards. Therefore, they are not effective to solve problems with large action spaces, which is the case for the problem that we have stated. Furthermore, they need to discretize continuous actions since they are based on discrete action values. Consequently, we adopt an actor-critic approach, which combines the merits of critic-only and actor-only algorithms. The actor can produce continuous or discrete actions without requiring an optimizer for the value function. The critic employs the estimate function to estimate the output of the actor, and the actor updates the policy parameters according to the estimated value to make the variance lower. [43, 44].

We propose an actor-critic algorithm to determine continuous actions for resource allocation and discrete actions for task offloading, as shown in Figure 2. The actor-critic model contains a critic network and an actor network. For the actor network, we derive a novel weighted loss function with two different action outputs, the continuous part and the discrete part. The learning rate of discrete and continuous action training is updated according to the weighted loss function in the actor training phase. And the actor parameters of continuous actions are iterated by gradient ascent based on DDPG [45], which is given as

$$\frac{1}{Z}\sum_k \nabla_{a_c} Q\left(s, \mu(s|\theta^\mu)\middle|\theta^Q\right)\Big|_{s=s_k} \nabla_{\theta^\mu}\mu(s|\theta^\mu)\Big|_{s_k}, \tag{15}$$

where Z represents the size of sample batch; θ^Q and θ^μ, respectively, express the weight and bias parameters in the critic network and the actor network; $\mu(\cdot)$ and $Q(\cdot)$ correspond to the output of actor network and critic network, respectively; and a_c is the continuous part action component for resource allocation F_t and W_t. We compute the gradient of $Q(s, \mu(s|\theta^\mu)|\theta^Q)$ to a_c instead of the whole $\mu(\cdot)$. Then, we consider it is constant for the discrete part action component (i.e., offloading decision vector m_t). The gradient of $Q(s, \mu(s|\theta^\mu)|\theta^Q)$ for the discrete action is calculated as

$$-\frac{1}{Z}\sum_k Q\left(s, \mu(s|\theta^\mu)\middle|\theta^Q\right)\Big|_{s=s_k} \cdot \nabla_{\theta^\mu}\left(\frac{1}{n}\sum_i m_i^k p_i^k\right), \tag{16}$$

where $m_i^k \in \{0, 1, 2\}$ is the offloading decision variable in the kth sample from the replay buffer and $(p_1^k, p_2^k, \cdots, p_n^k)$ denotes the probability of offloading modes for UEs, which is the first component of the actor output $\mu(\cdot)$. Different from Equation (15), the continuous part action a_c and $Q(s, \mu(s|\theta^\mu)|\theta^Q)$ are, respectively, fixed at constant action and constant weight. The weighted loss function for discrete and continuous actions can be expressed as

$$\begin{aligned} L^{\text{actor}} = &-w_c\sum_k Q\left(s, \mu(s|\theta^\mu)\middle|\theta^Q\right)\Big|_{s=s_k} \\ &- w_d\sum_k Q\left(s, \mu(s|\theta^\mu)\middle|\theta^Q\right)\Big|_{s=s_k} \cdot \left(\frac{1}{n}\sum_i m_i^k p_i^k\right), \end{aligned} \tag{17}$$

where w_c and w_d, respectively, correspond to the weight of the discrete and continuous part loss function. α_d and α_c denotes the learning rates of discrete and continuous part training phases. And they are update according to Equation (17) to get a good convergence performance.

For the critic network, we use the average square error loss function to iterate the parameters, which is defined as

$$L^{\text{critic}} = \frac{1}{Z}\sum_k \left(y_k - Q\left(s_k, \mu(s_k|\theta^\mu)\middle|\theta^Q\right)\right)^2. \tag{18}$$

Based on the above definitions, the proposed ACJORA algorithm is presented in Algorithm 1.

```
Input: actor network parameters $\theta^{\mu}$, critic network parameters $\theta^{Q}$, actor target network parameters $\theta^{\mu'}$, critic target network parameters $\theta^{Q'}$, discount factor $\gamma$, replay buffer $B$, batch size $Z$, epsilon greedy $\varepsilon$
Output: the best strategy $(m, W, F)$
1: Initialize: randomly initialize $\theta^{\mu}$ and $\theta^{Q}$, $\theta^{\mu'} \longleftarrow \theta^{\mu}$, $\theta^{Q'} \longleftarrow \theta^{Q}$, $B \longleftarrow \varnothing$
2: forepisode = 1 to $M$do
3:   Initialize state $s_0$
4:   fort = 1 to $T$do
5:     Actor output $(\widehat{m}_t, \widehat{F}_t, \widehat{W}_t) \longleftarrow \mu(s_t|\theta^{\mu})$.
6:     Add noise on $\mu(s_t|\theta^{\mu})$ with $\varepsilon$-greedy on $\widehat{m}_t$ and Gaussian distribution with mean $(\widehat{F}_t, \widehat{W}_t)$.
7:     Get action $a_t = (\widehat{m}_t, \widehat{F}_t, \widehat{W}_t)$ with exploration variance $V$.
8:     Take action $a_t$, observe reward $r_t$ and next state $s_{t+1}$.
9:     Store transition $(s_t, a_t, r_t, s_{t+1})$ in $B$.
10:      Sample a random batch of $Z$ transitions $(s_k, \mu(s_k|\theta^{\mu}), r_k, s_{k+1})$ from $B$.
11:      Set $y_k = r_k + \gamma Q'(s_{k+1}, \mu'(s_{k+1}|\theta^{\mu'}|\theta^{Q'}))$.
12:      Update the critic with minimizing the loss $L^{\text{critic}}$ by Equation (18).
13:      According to the loss $L^{\text{actor}}$, update the actor through the continuous part training phase $lr_c$ and the discrete part training phase $lr_d$ by Equations (15) and (16)
14:      Update the target networks: $\theta^{Q'} \longleftarrow \lambda\theta^{Q} + (1-\lambda)\theta^{Q'}$, $\theta^{\mu'} \longleftarrow \lambda\theta^{\mu} + (1-\lambda)\theta^{\mu'}$.
15:   end for
16: end for
```

Algorithm 1: Actor-critic-based joint offloading and resource allocation algorithm.

Table 3: Simulation parameters.

Parameters	Value
The number of UEs (N)	5
The communication bandwidth (W)	10 MHz
The computational capacity of MEC server (F)	8 GHz
The computational capacity of UE i (f_i)	[0.5,1.5] GHz
The distance between BS and UEs	[0, 200] m
The transmit power of BS (p)	500 mW
The maximum tolerated delay for RM construction and dissemination	1 s
The LR RM size	100 Kbit
The HR RM size	250 Kbit
Discount factor (γ)	0.99
Replay buffer (B)	100
Batch size (Z)	32
Epsilon greedy (ε)	0.9

Table 4: The average latency of three scheme.

	All server	All local	Random offloading	DQN-based	Proposed
Average latency (ms)	878	643	735	286	194

5. Simulation Results

5.1. Parameter Setting. In the simulations, we consider a MEC network as show in Figure 1, which includes a BS and N UEs. A MEC server is connected to the BS. UEs are randomly distributed in an area of [0, 200] meters from the BS. Communication bandwidth is $W = 10$ MHz. The computational capacity of MEC server is $F = 8$ GHz, the CPU frequency of the UE is random in the range [0.5, 1.5] GHz. The LR RM size is set as 100 Kbit (e.g., a LR image is 1280×720 with 16 bit/pixel, using a compression ratio of 150 : 1 [46]). And the HR RM size is set as 250 Kbit (e.g., a HR image is 1920×1080 with 16 bit/pixel, using a compression ratio of 150 : 1). The maximum tolerated delay for RM construction and dissemination is $T_{\max} = 1$ s. The transmit power of BS is $p = 500$ mW. Detailed simulation parameters are listed in Table 3.

5.2. Result Analysis. For fair performance evaluation, we compare our proposed scheme with four baseline schemes: (1) All server: the offloading decision variable of all UEs is $m_i = 0$. The MEC server constructs RM and disseminates the compressed RM. (2) All local: the offloading decision

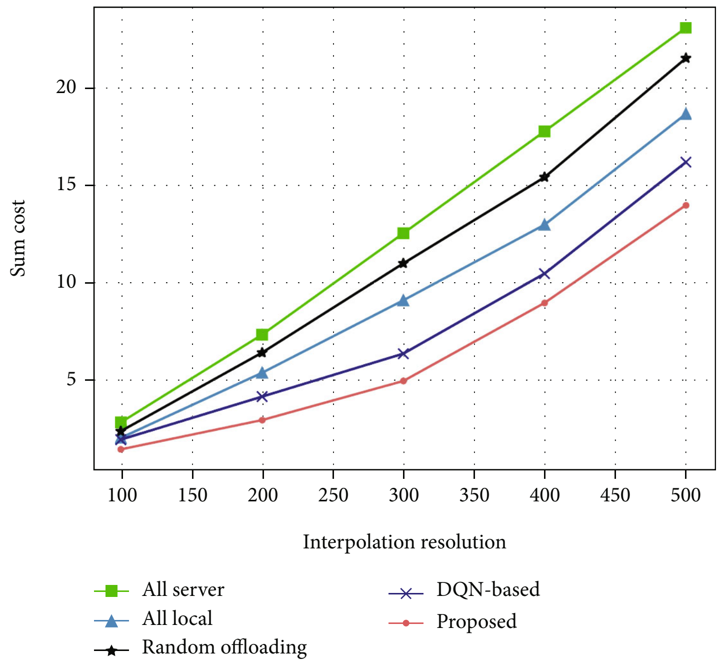


Figure 3: The effect of interpolation resolution on the sum cost.

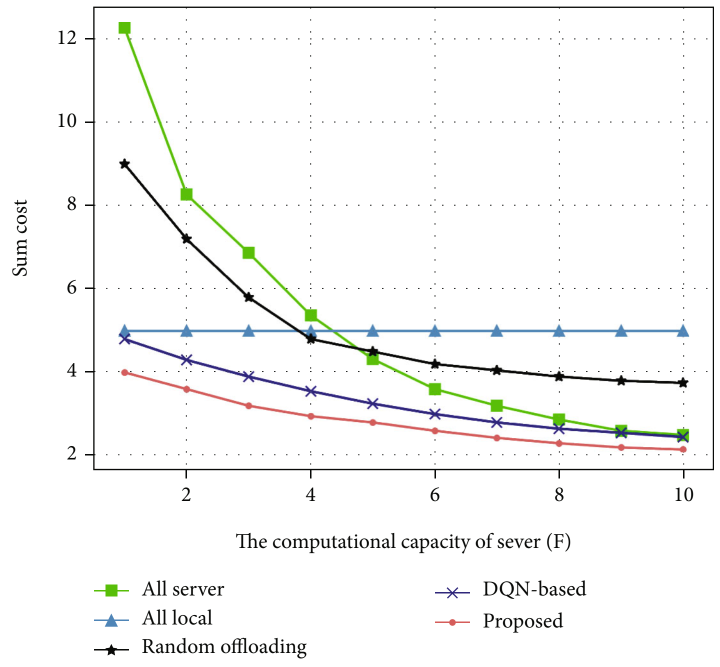


Figure 4: The effect of the computational capacity for server on total consumption.

variable of all UEs is $m_i = 2$, distributing global spectrum information and constructing RM at user equipment. (3) Random offloading: all UEs randomly select one of the three dissemination methods: all server, all local, and partial offloading, $m_i \in \{0, 1, 2\}$. (4) DQN-based: a resource scheduling scheme based on deep Q-network (DQN), which is an action-value method [47].

Table 4 shows the average latency of disseminating RM to UEs in a scenario where the number of UEs is 5, the computational capacity of MEC server is 8 GHz/sec, and the communication bandwidth is 10 MHz. We conducted 5000 Monte Carlo experiments and took the average value of the experimental results. The average delay of the proposed scheme is 194 ms, which is 77.90% lower than that of all server construction scheme, 69.83% lower than that of all local construction scheme, and 73.61% lower than that of the random offloading scheme, 32.17% lower than DQN-based scheme.

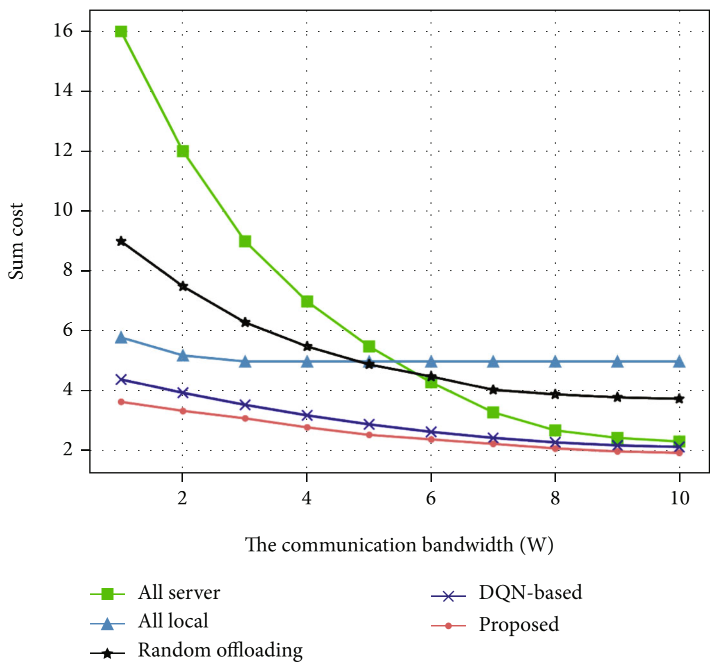


Figure 5: The effect of communication bandwidth on total consumption.

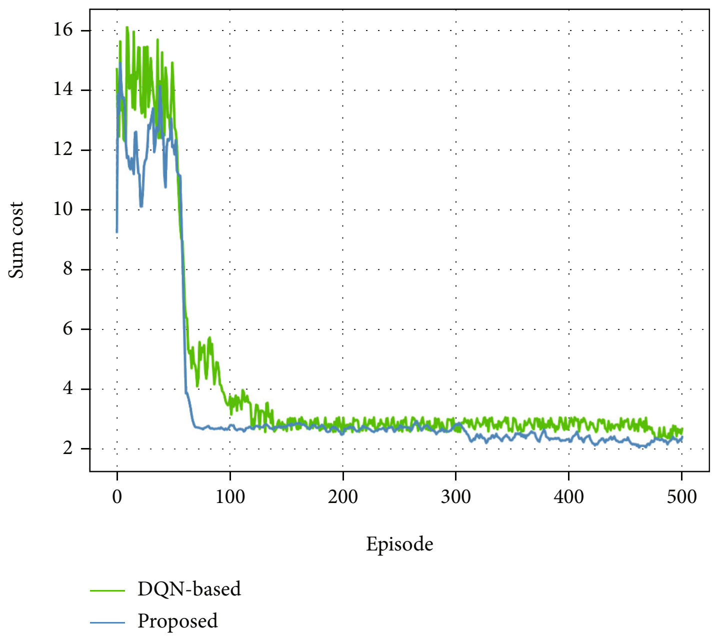


Figure 6: The comparison of convergence performance.

As shown in Figure 3, with the increasing of interpolation resolution (i.e., the number of interpolation points by Kriging interpolation), the sum cost of all schemes increase at the same time. Because the computational complexity and the data size of RM dissemination will increase as the interpolation resolution increases. The DRL scheme can efficiently allocate computation and communication resources. However, DQN-based scheme cannot produce continuous actions, so it is need to discretize continuous variables. Thus, the strategies of the DQN-based scheme are worse than that of the proposed scheme in resource allocation. The proposed scheme can achieve the best performance with minimal sum cost with the increase of RM interpolation resolution.

Figure 4 illustrates the sum cost of the MEC system as the computational capacity of the edge server increases. The all local curve does not change with the increase of the computation resources of MEC server because UEs does not use the computation resources of the server. The other curves decrease as the computational capacity of MEC server increases. Because each UE is allocated more server computation resources, the computation time will be shortened accordingly. In addition, when $F > 8$ GHz/sec, the sum cost of all server scheme and the proposed scheme decreases slowly. The result shows that when the computation resources of MEC server are far more than the computation resource of UE, the sum cost of MEC network is mainly limited by other factors such as communication bandwidth resources.

Figure 5 depicts the sum cost of the MEC system as communication bandwidth increases. Due to the small transmission data of the all local scheme, it is less affected by the communication bandwidth. And the sum cost of the all local scheme at $W = 3$ MHz does not change with the increase of communication bandwidth. The sum cost of other schemes decrease with the increase of the communication bandwidth, because each UE can be allocated more bandwidth resources, and the communication transmission time will be shortened. And the proposed scheme has the least sum cost. Figures 4 and 5 show that the proposed scheme has good adaptability in a varying radio environment.

We compared the convergence performance of DQN-based scheme and proposed scheme in Figure 6. The sum cost of the both RL learning schemes has decreased rapidly with the number of episodes. Finally, the most effective offloading and resource allocation strategies are learned and the sum cost of the system has stabilized. Compared with the DQN-based scheme, the proposed scheme can converge with fewer episodes, and its sum cost is less. Figure 6 shows that the proposed scheme can efficiently train offloading and resource allocation strategies.

6. Conclusions

RM is an important tool for cognitive radio in the 6G era, which can provide data support for IIoT. To solve the problem of RM construction and dissemination in resource-limited and delay-sensitive MEC networks, we have proposed a joint RM construction and dissemination approach based on DRL, which is described as actor-critic-based joint offloading and resource allocation (ACJORA) algorithm. In the algorithm, we designed a novel actor-critic model with a weighted loss function for the actor network, which combines the discrete actions for task offloading and continuous actions for resource allocation. Compared with baseline schemes, the proposed algorithm can effectively reduce the energy consumption and delay of RM construction and dissemination, which significantly reduces the cost of acquiring RM for UEs.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants 62171449, 62001483, and U19B2024.

References

[1] K. K. R. Choo, S. Gritzalis, and J. H. Park, "Cryptographic solutions for industrial internet-of-things: research challenges and opportunities," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3567–3569, 2018.

[2] M. Z. Hasan and H. al-Rizzo, "Optimization of sensor deployment for industrial internet of things using a multiswarm algorithm," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10344–10362, 2019.

[3] K. Tange, M. De Donno, X. Fafoutis, and N. Dragoni, "A systematic survey of industrial internet of things security: requirements and fog computing opportunities," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2489–2520, 2020.

[4] K. Katagiri and T. Fujii, "Mesh-clustering-based radio maps construction for autonomous distributed networks," in *2021 Twelfth International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 345–349, Jeju Island, Korea, August 2021.

[5] P. Bednarek, J. Łopatka, and D. Bicki, "Radio environment map for the cognitive radio network simulator," *International Journal of Electronics and Telecommunications*, vol. 64, no. 1, pp. 45–49, 2018.

[6] N. Ezzati and H. Taheri, "Distributed spectrum sensing in rem based cognitive radio networks," *Journal of Modeling in Engineering*, vol. 17, no. 56, pp. 223–233, 2019.

[7] P. Kaniewski, J. Romanik, E. Golan, and K. Zubel, "Spectrum awareness for cognitive radios supported by radio environment maps: zonal approach," *Applied Sciences*, vol. 11, no. 7, p. 2910, 2018.

[8] Y. H. Santana, D. Plets, R. M. Alonso et al., "Tool for recovering after meteorological events using a real-time REM and IoT management platform," *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 9767404, 13 pages, 2019.

[9] S. C. Arum, D. Grace, and P. D. Mitchell, "A review of wireless communication using high-altitude platforms for extended coverage and capacity," *Computer Communications*, vol. 157, pp. 232–256, 2020.

[10] H. Braham, S. B. Jemaa, G. Fort, E. Moulines, and B. Sayrac, "Fixed rank kriging for cellular coverage analysis," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 5, pp. 4212–4222, 2017.

[11] H. Braham, S. B. Jemaa, G. Fort, E. Moulines, and B. Sayrac, "Spatial prediction under location uncertainty in cellular networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 11, pp. 7633–7643, 2016.

[12] L. Yin, J. Luo, and H. Luo, "Tasks scheduling and resource allocation in fog computing based on containers for smart manufacturing," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4712–4721, 2018.

[13] M. G. Kibria, K. Nguyen, G. P. Villardi, O. Zhao, K. Ishizu, and F. Kojima, "Big data analytics, machine learning, and artificial intelligence in next-generation wireless networks," *IEEE Access*, vol. 6, pp. 32328–32338, 2018.

[14] R. Atawia, H. S. Hassanein, N. Abu Ali, and A. Noureldin, “Utilization of stochastic modeling for green predictive video delivery under network uncertainties,” *IEEE Transactions on Green Communications and Networking*, vol. 2, no. 2, pp. 556–569, 2018.

[15] M. Li, J. Gao, L. Zhao, and X. Shen, “Deep reinforcement learning for collaborative edge computing in vehicular networks,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, pp. 1122–1135, 2020.

[16] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge computing: vision and challenges,” *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[17] F. Khoramnejad and M. Erol-Kantarci, “On joint offloading and resource allocation: a double deep q-network approach,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 4, pp. 1126–1141, 2021.

[18] S. Nath and J. Wu, “Deep reinforcement learning for dynamic computation offloading and resource allocation in cache-assisted mobile edge computing systems,” *Intelligent and Converged Networks*, vol. 1, no. 2, pp. 181–198, 2020.

[19] A. M. Seid, G. O. Boateng, S. Anokye, T. Kwantwi, G. Sun, and G. Liu, “Collaborative computation offloading and resource allocation in multi-UAV-assisted IoT networks: a deep reinforcement learning approach,” *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12203–12218, 2021.

[20] H. Zhou, K. Jiang, X. Liu, X. Li, and V. C. M. Leung, “Deep reinforcement learning for energy-efficient computation offloading in mobile-edge computing,” *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1517–1530, 2021.

[21] Z. Ning, Y. Yang, X. Wang et al., “Dynamic computation offloading and server deployment for UAV-enabled multi-access edge computing,” *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[22] H. Zhou, Z. Zhang, D. Li, and Z. Su, “Joint optimization of computing offloading and service caching in edge computing-based smart grid,” *IEEE Transactions on Cloud Computing*, p. 1, 2022.

[23] H. Zhou, T. Wu, H. Zhang, and J. Wu, “Incentive-driven deep reinforcement learning for content caching and D2D offloading,” *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2445–2460, 2021.

[24] Z. Ning, P. Dong, M. Wen et al., “5G-enabled UAV-to-community offloading: joint trajectory design and task scheduling,” *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 11, pp. 3306–3320, 2021.

[25] Z. Ning, S. Sun, X. Wang et al., “Intelligent resource allocation in mobile blockchain for privacy and security transactions: a deep reinforcement learning based approach,” *Science China Information Sciences*, vol. 64, no. 6, article 162303, 2021.

[26] M. Höyhtyä, A. Mämmelä, M. Eskola et al., “Spectrum occupancy measurements: a survey and use of interference maps,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2386–2414, 2016.

[27] L. Zhou, Z. Sheng, L. Wei et al., “Green cell planning and deployment for small cell networks in smart cities,” *Ad Hoc Networks*, vol. 43, pp. 30–42, 2016.

[28] X. Wang, Z. Ning, X. Hu et al., “Future communications and energy management in the internet of vehicles: toward intelligent energy-harvesting,” *IEEE Wireless Communications*, vol. 26, no. 6, pp. 87–93, 2019.

[29] H. B. Yilmaz, T. Tugcu, F. Alagöz, and S. Bayhan, “Radio environment map as enabler for practical cognitive radio networks,” *IEEE Communications Magazine*, vol. 51, no. 12, pp. 162–169, 2013.

[30] V. P. Chowdappa, C. Botella, J. J. Samper-Zapater, and R. J. Martinez, “Distributed radio map reconstruction for 5G automotive,” *IEEE Intelligent Transportation Systems Magazine*, vol. 10, no. 2, pp. 36–49, 2018.

[31] A. P. Sergeev, D. A. Tarasov, A. G. Buevich et al., “High variation subarctic topsoil pollutant concentration prediction using neural network residual kriging,” *AIP Conference Proceedings*, vol. 1836, no. 1, article 020023, 2017.

[32] Y. Deng, L. Zhou, L. Wang et al., “Radio environment map construction using super-resolution imaging for intelligent transportation systems,” *IEEE Access*, vol. 8, pp. 47272–47281, 2020.

[33] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman, and D. O. Wu, “Edge computing in industrial internet of things: architecture, advances and challenges,” *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2462–2488, 2020.

[34] X. Kong, G. Duan, M. Hou et al., “Deep reinforcement learning-based energy-efficient edge computing for internet of vehicles,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6308–6316, 2022.

[35] Z. Ning, S. Sun, X. Wang et al., “Blockchain-enabled intelligent transportation systems: a distributed crowdsensing framework,” *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[36] X. Kong, S. Tong, H. Gao et al., “Mobile edge cooperation optimization for wearable internet of things: a network representation-based framework,” *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 5050–5058, 2021.

[37] X. Wang, Z. Ning, S. Guo, M. Wen, L. Guo, and V. Poor, “Dynamic UAV deployment for differentiated services: a multi-agent imitation learning based approach,” *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[38] D. C. Nguyen, M. Ding, P. N. Pathirana et al., “6G internet of things: a comprehensive survey,” *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 359–383, 2022.

[39] J. Du, F. R. Yu, G. Lu, J. Wang, J. Jiang, and X. Chu, “MEC-assisted immersive VR video streaming over terahertz wireless networks: a deep reinforcement learning approach,” *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9517–9529, 2020.

[40] A. E. M. Taha, N. Abu Ali, H. R. Chi, and A. Radwan, “MEC resource offloading for QoE-aware HAS video streaming,” in *ICC 2021 - IEEE International Conference on Communications*, pp. 1–5, Montreal, QC, Canada, June 2021.

[41] K. Genda, M. Abe, and S. Kamamura, “Video communication optimization using distributed edge computing,” in *2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pp. 381–384, Daegu, Korea (South), September 2020.

[42] Y. Wei, F. R. Yu, M. Song, and Z. Han, “User scheduling and resource allocation in HetNets with hybrid energy supply: an actor-critic reinforcement learning approach,” *IEEE Transactions on Wireless Communications*, vol. 17, no. 1, pp. 680–692, 2018.

[43] R. Li, Z. Zhao, X. Chen, J. Palicot, and H. Zhang, “TACT: a transfer actor-critic learning framework for energy saving in cellular radio access networks,” *IEEE Transactions on Wireless Communications*, vol. 13, no. 4, pp. 2000–2011, 2014.

[44] W. Jiang, D. Feng, Y. Sun, G. Feng, Z. Wang, and X.-G. Xia, "Proactive content caching based on actor–critic reinforcement learning for mobile edge networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 2, pp. 1239–1252, 2021.

[45] T. P. Lillicrap, J. J. Hunt, A. Pritzel et al., "Continuous control with deep reinforcement learning," 2015, https://arxiv.org/abs/1509.02971.

[46] E. Bastug, M. Bennis, M. Medard, and M. Debbah, "Toward interconnected virtual reality: opportunities, challenges, and enablers," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 110–117, 2017.

[47] Z. Wu and D. Yan, "Deep reinforcement learning-based computation offloading for 5G vehicle-aware multi-access edge computing network," *China Communications*, vol. 18, no. 11, pp. 26–41, 2021.