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

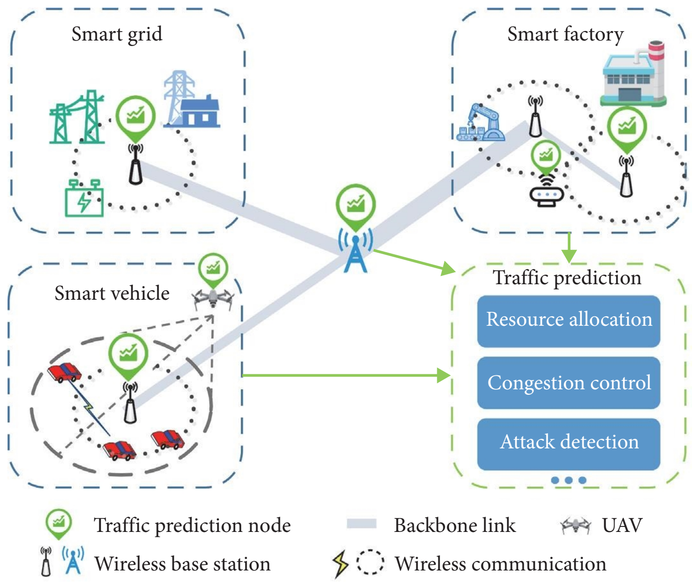

Figure 1: An illustration of edge network traffic prediciton in IIoT.

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:

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

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:

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

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:

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

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.

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.

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

4.2. 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

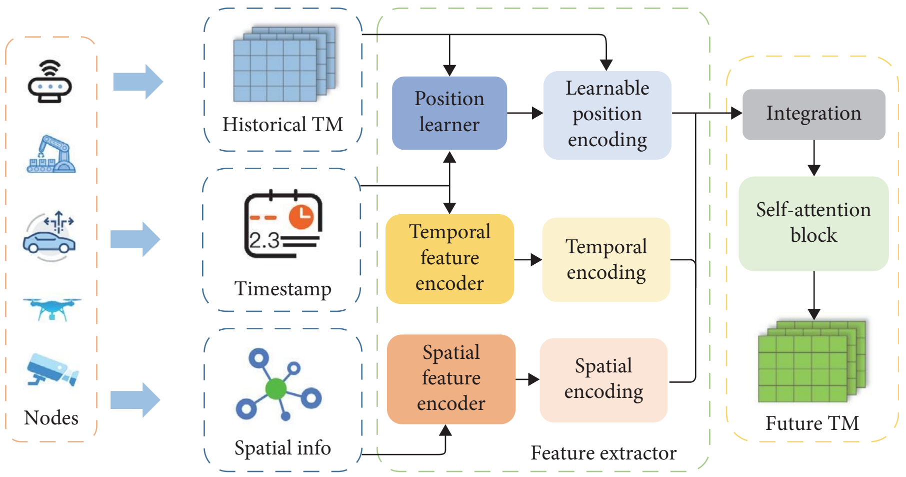

Figure 2: Main framework of spatio-temporal self-attention network TM prediction model.

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

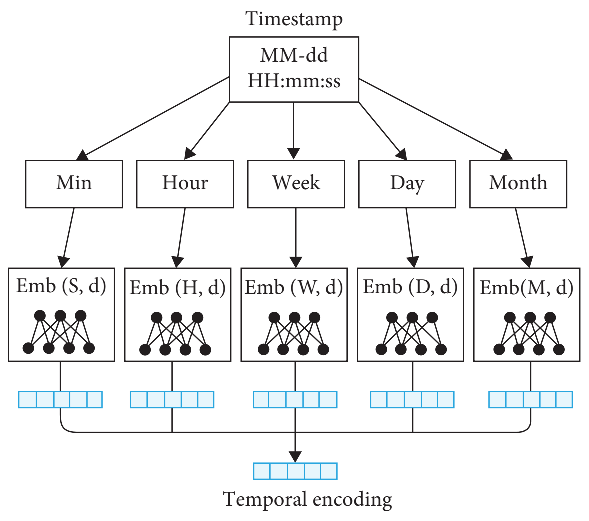

Figure 3: The encoding of timestamp.

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 $\boldsymbol{R} = \{F_{11}, F_{12}, \dots, F_{NN}\}$, 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 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 $\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_t$

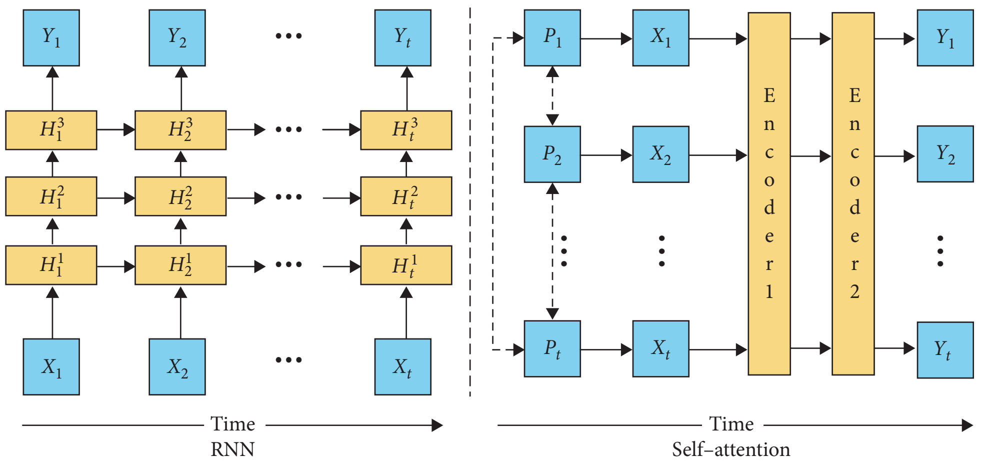

Figure 4: The training process of RNN and self-attention model.

is the input sequence, H_i^j is the j_{th} hidden layer of time step i, and $\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots, \boldsymbol{Y}_t$ 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.

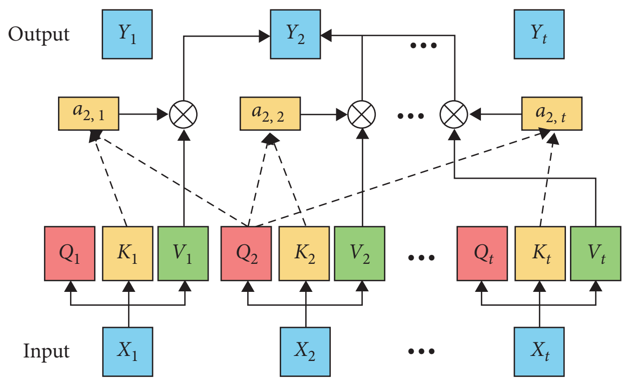

Figure 5: The calculation process of self-attention mechanism.

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:

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

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 $O(d^2 n)$ 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:

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

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

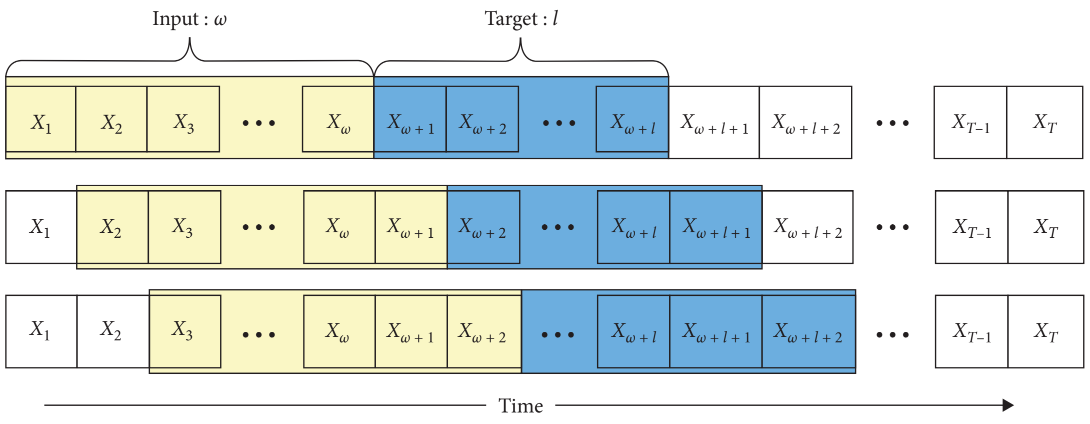

Figure 6: Sliding window for building training dataset.

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

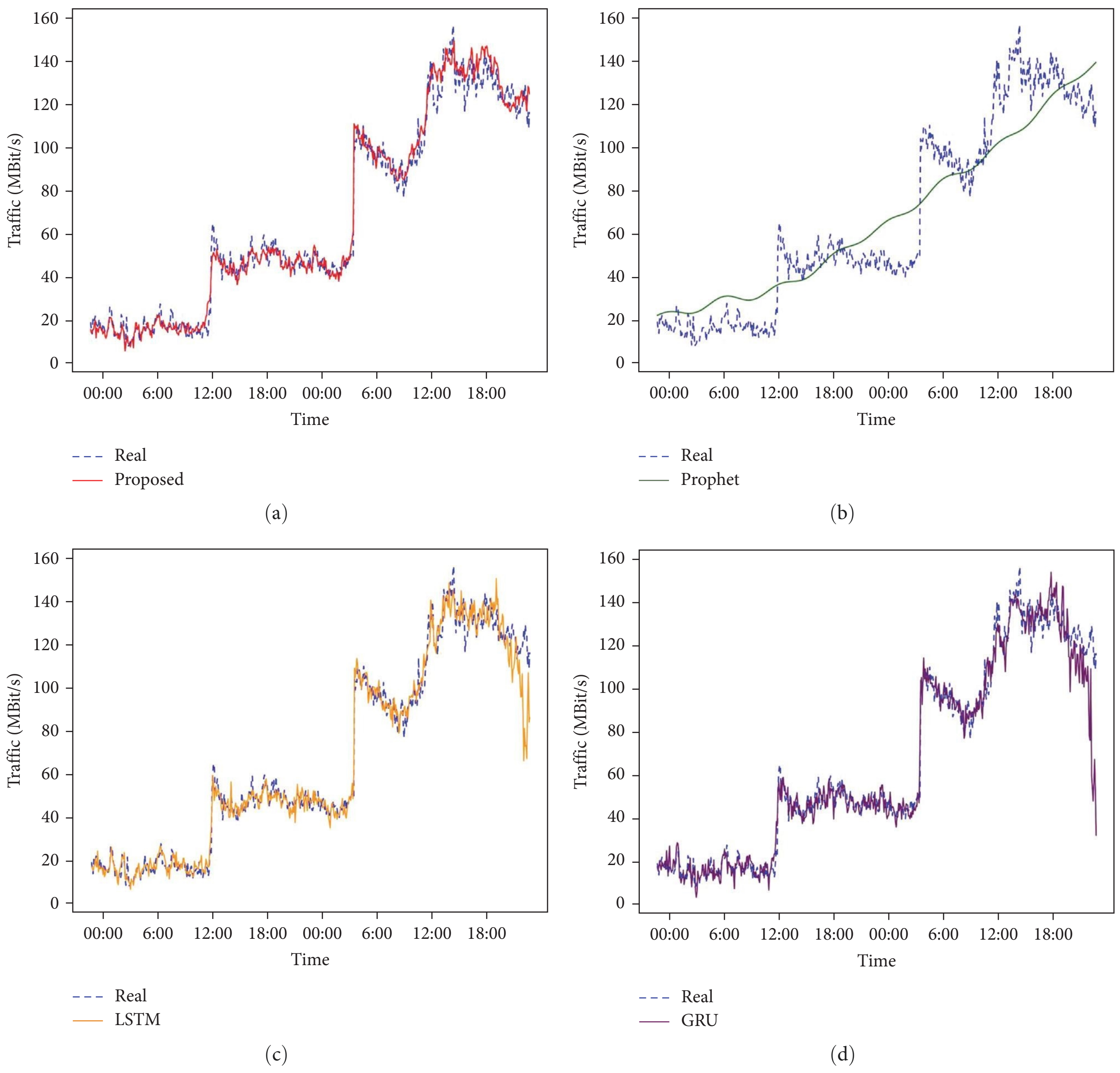

Figure 7: The prediction (48 hr) of SATMP, Prophet, LSTM and GRU on the Abilene dataset.

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

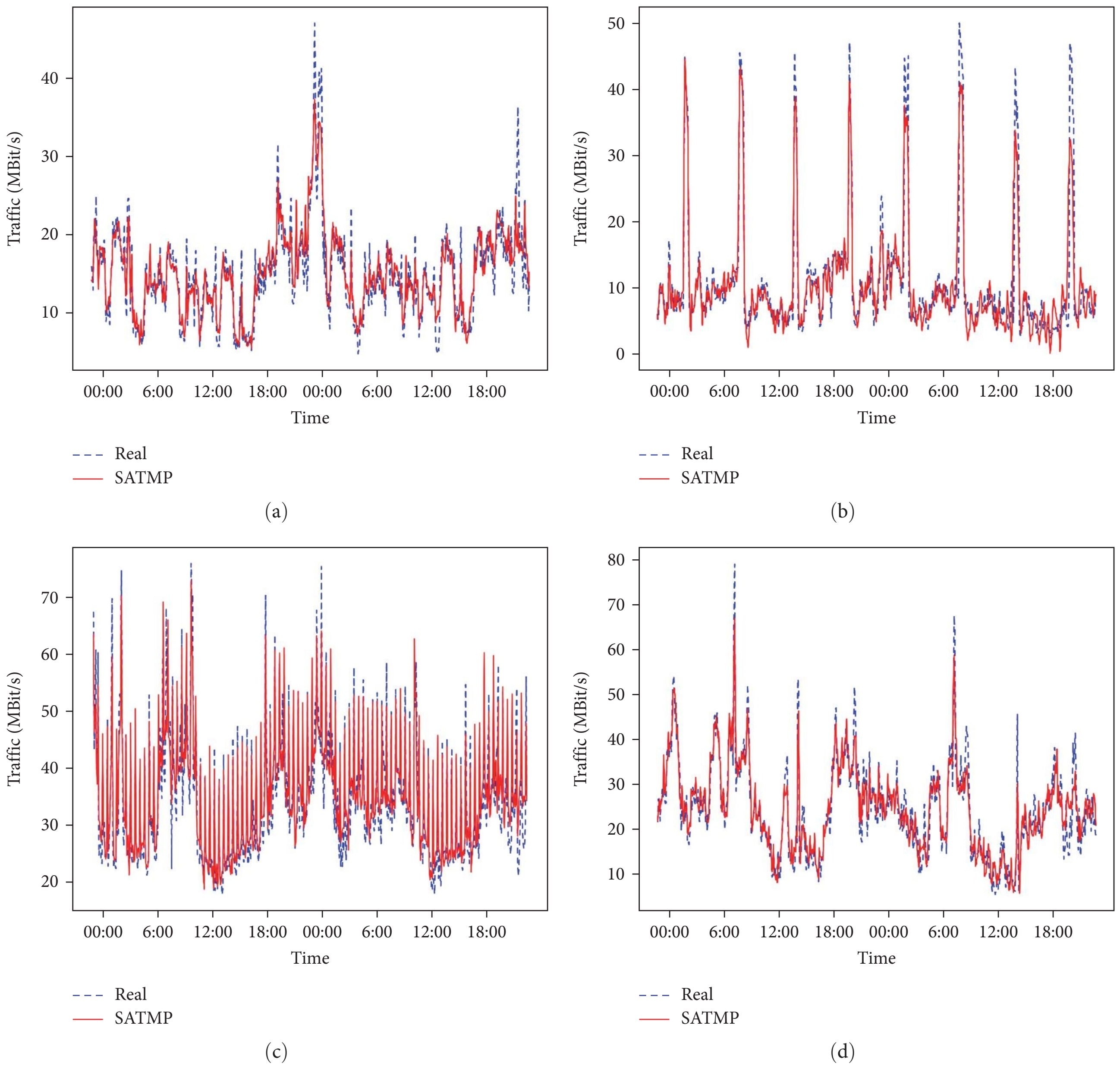

Figure 8: The prediction (48 hr) of SATMP on Abilene.

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

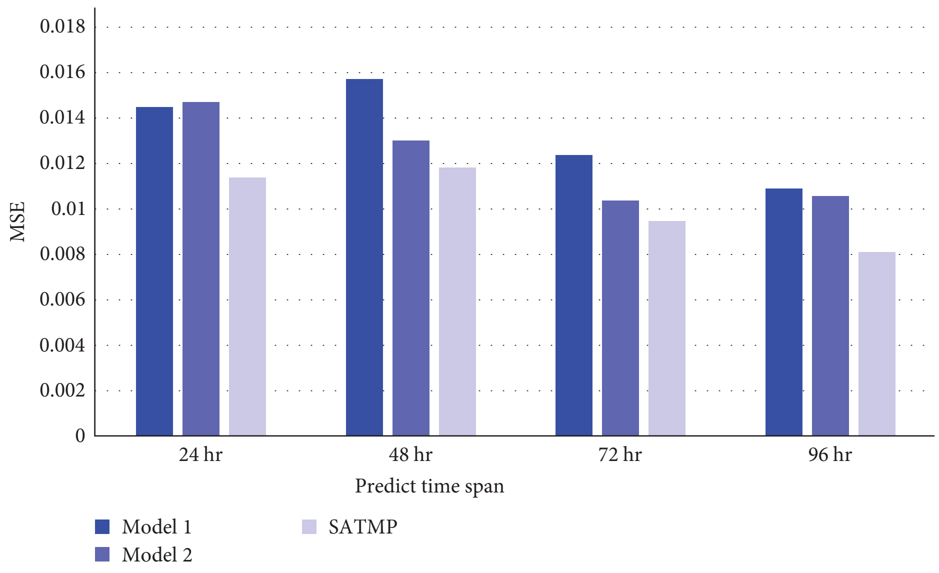

Figure 9: Ablation study of SATMP.

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.