Hindawi
Discrete Dynamics in Nature and Society
Volume 2022, Article ID 8190688, 11 pages
https://doi.org/10.1155/2022/8190688

Research Article

Machine Learning with Variable Sampling Rate for Traffic Prediction in 6G MEC IoT

Rongqun Peng,[1,2] **Xiuhua Fu**,[1,2] **and Tian Ding**[1,2]

[1]*School of Computer Science and Technology, Shandong University of Technology, Zibo 255000, China*
[2]*State Key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications), Beijing 100876, China*

Correspondence should be addressed to Rongqun Peng; pengrq@sdut.edu.cn

Received 8 July 2022; Revised 30 September 2022; Accepted 5 October 2022; Published 17 November 2022

Academic Editor: Bo Rong

The high-speed development of mobile broadband networks and IoT applications has brought about massive data transmission and data processing, and severe traffic congestion has adversely affected the fast-growing networks and industries. To better allocate network resources and ensure the smooth operation of communications, predicting network traffic becomes an important tool. We investigate in detail the impact of variable sampling rate on traffic prediction and propose a high-speed traffic prediction method using machine learning and recurrent neural networks. We first investigate a VSR-NLMS adaptive prediction method to perform time series prediction dataset transformation. Then, we propose a VSR-LSTM algorithm for real-time prediction of network traffic. Finally, compared with the traditional traffic prediction algorithm based on fixed sampling rate (FSR-LSTM), we simulate the prediction accuracy of the VSR-LSTM algorithm based on the variable sampling rate proposed. The experiment shows that VSR-LSTM has higher traffic prediction accuracy because its sampling rate varies with the traffic.

1. Introduction

As the global mobile industry moves toward 6G networks, mobile edge computing (MEC)-based network infrastructure has received unprecedented attention to support Internet of Things (IoTs) applications with diverse business needs [1]. To better serve users, the connectivity and intelligence provided by edge computing have tremendous advantages in terms of real-time services, smart living, security, and reliability [2]. At present, edge computing has been applied in smart campuses, video surveillance, industrial IoTs, augmented reality/virtual reality (AR/VR) and other application scenarios, which also proves that MEC-based network infrastructure is effective and fully capable.

The capabilities of edge computing rely on edge servers, which are usually deployed with base stations. It is expected that more IoTs applications will be carried out based on MEC in the future, and the massive data they generate will have great demands on network resources such as bandwidth, computing power, and storage [3, 4]. Therefore, in future, multiple MEC servers will be needed to jointly provide services for different applications. Since MEC servers have different computing processing capabilities and are deployed in a distributed manner, it is critical to offload computing tasks to these heterogeneous MEC servers according to different application requirements. Therefore, reasonable and effective resource allocation mechanisms according to the usage of wireless network resources and MEC server resources will effectively ensure the service requirements of users.

Resource allocation will directly affect the operating cost of MEC-based network infrastructure and the experience of IoTs users. Unreasonable resource allocation will not only increase the operating costs of communication networks but also may lead to serious energy waste. Accurate wireless network traffic prediction can intuitively reflect the changing trend of service requirements, provide an important reference for communication network resource allocation, and is an important guarantee to achieve reasonable and efficient resource allocation and computing task offloading [5, 6].

It has always been a meaningful research topic to analyze the distribution and demand of communication traffic by predicting the wireless network traffic to guide the communication resources' allocation [7]. The previous wireless traffic prediction adopted manual prediction and statistical-based prediction methods, and their limitations are obvious. Manual prediction is inefficient and cannot adjust the network resource allocation in real-time according to the change of service demands. The statistics-based traffic prediction models intelligently utilize certain statistical characteristics, but cannot effectively comprehensively utilize various information that has an important impact on wireless traffic [8]. As machine learning techniques are intensively researched, there have been some works using machine learning and deep learning algorithms for wireless network traffic prediction [9–18].

In the future, artificial intelligence will be one of the native features of the next-generation mobile communication network, namely the sixth-generation (6G). Through AI-based endogenous intelligent design in air interface algorithms, wireless network architecture, and wireless traffic prediction, etc., 6G-based communication network can better achieve network autonomy and intelligence, thus realizing intelligent operation and maintenance management of 6G network infrastructure, including MEC servers and efficient automatic deployment of services [19, 20]. The traffic prediction model based on AI algorithm can automatically mine various features contained in wireless data and comprehensively use these features to accurately predict wireless traffic in real-time. Accurate traffic prediction can directly reflect the spatial and temporal distribution of communication service demands, guide the network resources allocation in different network nodes, and then offload thecomputing tasks in different MEC servers, thereby improving user service experience and enhancing the autonomous and intelligent operation and maintenance of communication networks.

Machine-to-machine (M2M) communications, autonomous driving, and virtual reality are just a few of the new applications that 6G cellular networks are expected to make possible in the next few years. These applications all call for better network latency, capacity, and context awareness. It is critical to make the network aware of traffic demands to achieve these strict standards. The development of an intelligent network necessitates traffic analysis and accurate forecasting of user demand. Knowing user demand ahead of time allows the network to allocate resources more efficiently. The network can manage resource distribution between users who are competing for resources promptly.

The remainder of this paper is organized as follows. Related works closely to our research are listed in Section 2. A traffic prediction model for 6G MEC IoT is described in Section 3. Section 4 focuses on traffic prediction methods with variable sampling rates(VSR) using machine learning and RNN techniques. In this section, the VSR-NLMS adaptive prediction method to perform time series prediction dataset transformation and the VSR-LSTM algorithm for real-time prediction of network traffic are proposed. Simulation and performance analysis are presented in Section 5. Section 6 concludes the full text.

2. Related Works

Network traffic analysis and prediction are fundamental to traffic engineering, network planning, optimization, administration and maintenance, resource allocation, load balancing, etc.. The previous wireless traffic prediction adopted manual prediction and statistical-based prediction methods, which have lots of flaws such as Low efficiency and non-real-time. As machine learning techniques are intensively researched, there have been some works using machine learning and deep learning algorithms for wireless network traffic prediction.

All of [9–14] address network traffic prediction problems in the telecom network. In these research works except [12], LSTM or LSTM variants has been adopted and achieved good prediction performance compared with other algorithms such as ARIMA, SVR, FFNN, RFR, KNNR, etc.In [9–11, 14], the prediction focused on aggregated behavior, e.g., considering traffic volumes observed over a given time interval (normally 5–15 minutes), which is coarse-prediction.

In [12, 13], the author investigates and specializes a set of architectures selected among convolutional, recurrent, and composite neural networks to predict mobile-app traffic at the finest (packet-level/mobile app) granularity. To provide the AI-native services for the 6G vision, the author in [14] proposed a novel edge-native framework to provide an intelligent prognosis model using LSTM-based encoder-decoder for data traffic prediction. The prognosis model was trained on real time-series multivariate data records collected from the edge μ-boxes of a selected testbed network.

In [15–18], the traffic predictions of Internet or campus network, such as ARQ message and ping command, are all fine-grained prediction. Especially in [18], wavelet transform is used to preprocess the data before prediction, transforming the 1-dimensional time series data into 3-dimensional data, which is more conducive to GRU (a LSTM variant) feature extraction and then obtained a better performance than RNN.

A detailed comparison of these related works has been compared according to the aspects of research objects, traffic data granularity, used data sets, and algorithms in this paper, as shown in Table 1.

From the above discussion, it can be seen that compared with other algorithms, LSTM has better performance in predicting time series data such as network traffic. But although not specified in the literature, especially telecommunications network traffic prediction, the traffic data generated using a fixed sampling rate, without considering traffic speed changes, will lead to large complexity of computation or predict the problem of inaccuracy. Aiming at the above problems, a new algorithm, VSR-LSTM, which combines variable sampling rate and LSTM, is proposed in this paper.

In addition, for readers' convenience, all the acronyms in this paper have been collected and listed in Table 2.

3. System Model

As shown in Figure 1, the mobile edge computing system model for machine learning-based traffic prediction is a

Table 1: Related works performing prediction of different research objects and by means of different techniques.

[Ref]	Research object	Fine or coarse	Dataset	Simulator	Techniques
[9]	Telcom network	Coarse (15 min)	GEANTWIDE	Not mentioned	ARIMA SVR LSTM RCLSTM SR-based
[10]	Telcom user traffic and location	Coarse (15 min)	GENAT	Not mentioned	ARIMA SVR LSTM RCLSTM FFNN
[11]	Telcom. Network	Coarse	Operator data	Python	ARIMA FFNN LSTM
[12]	Mobile APP	Fine	MIRAGE-2019	Not mentioned	HMM RFR LR K-NNR
[13]	Mobile APP	Fine	MIRAGE-2019	Not mentioned	RFR MC CNN LSTM GRU
[14]	Mobile 6G network	Coarse (5 min)	Locally obtained	Edge μ-boxes, jupyter notebook	LSTM-based encoder and decoder
[15]	University campus datacenter	Fine	EDU1 dataset	Python, keras	CNN RF
[16]	SDN controller (ONOS)	Fine	Ping ARQ message	Not mentioned	DNN SF LDA
[17]	Internet	Fine	DNS traffic	Python, TensorFlow	SVR BPNN LSTM
[18]	Internet	Fine	User data	MATLAB	GRU RNN

Table 2: Acronyms list in this paper.

Acronyms	Full name
ARIMA	Auto regressive integrated moving average
ARQ	Automatic repeat response
CNN	Convolutional neural network
DNN	Deep neural network
FFNN	Feed forward neural network
FSR	Fixed sampling rate
GRU	Gated recurrent unit
HMM	Hidden markov models
MEC	Mobile edge computing
NRMSE	Normalized root mean square error
KNNR	K-nearest neighbor regressor
LSTM	Long short-term memory
LDA	Linear discriminant analysis
LR	Liner regression
MC	Markov chain
RCLSTM	Random connectivity LSTM
RF	Random forest
RFR	Random forest regressor
RNN	Recurrent neural network
SR-based	Sparse representation-based
SVR	Support vector regression
VSR	Variable sampling rate

three-tier hierarchy consisting of a cloud platform, multiple MEC gateways, and a large number of end-users, including multiple independent IoT networks. Each IoT network serves many end-users (i.e., different end devices). Different terminal types and usage scenarios have different computing, storage, and communication capabilities. For example, smartphones have relatively high computational storage and communication capabilities, and rechargeable batteries have high energy supply capabilities. But some IoT nodes have deficient capacity compared to smartphones, especially many nodes that cannot replace batteries, and their computational storage and communication capabilities are greatly limited. At the same time, various services have different QoS requirements for latency, energy consumption, communication bandwidth, and other indicators, which require different processing methods for different terminal types and different business needs. Generally speaking, services with low computation and power requirements but high latency requirements can be executed on local terminals with strong capabilities, such as smartphones. For IoT nodes with limited computation and power, data can only be transmitted to gateway nodes or edge servers for processing, and for that kind of computationally

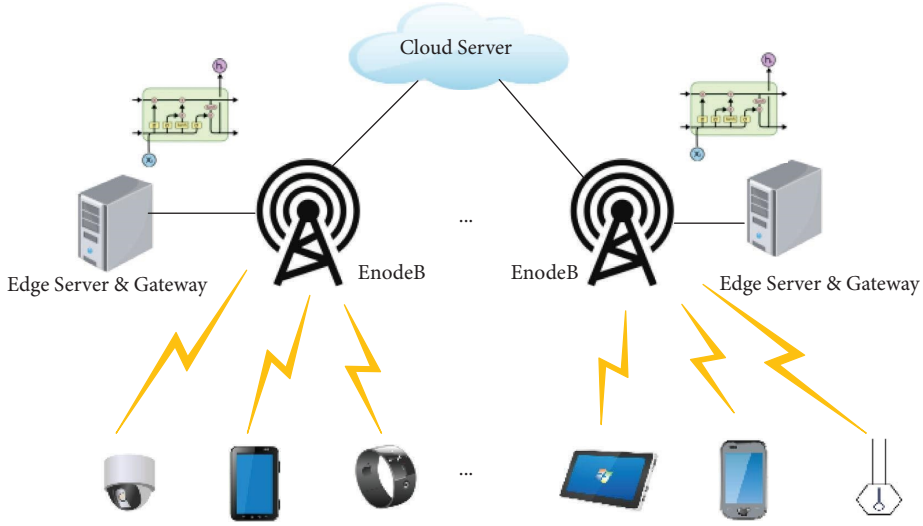

Figure 1: Traffic prediction for 6G MEC IoT.

intensive service, they also need to be offloaded to remote servers with unlimited energy computing power to process them.

In Figure 1, whether IoT end-users offload tasks to edge servers for execution or edge servers offload computationally intensive tasks to remote ends for execution, both require the system to allocate appropriate network bandwidth resources. Although 6G has increased the network speed and bandwidth a lot compared to 5G, the competition for network resources, especially bandwidth resources, still exists, and this resource competition may become more intense in the future, requiring dynamic control of network resources based on user demand. Traffic prediction is of great significance to achieving dynamic resource allocation and is a prerequisite and guarantee for the edge computing server to achieve dynamic resource allocation.

In addition to being able to complete the execution of tasks offloaded to it by users and offload tasks with greater computational demand to the cloud for execution, the edge server in Figure 1 should also have dynamic traffic prediction functions to accurately predict the bandwidth resources required for various offloading tasks and provide a basis for network bandwidth resource allocation. The cloud server mainly completes the computation-intensive tasks offloaded to it by the edge servers.

4. Prediction Methods

4.1. The Significance and Preliminary Solution of Developing Variable Sampling Rate. To avoid confusion in the forecast, if a constant sample rate is to be utilized, it must be double the maximum frequency of the traffic load profile. Since more traffic sample operations likewise generate more traffic prediction operations, a constant sampling rate puts significant computing complexity into the system. As a result, developing a low-sampling-rate solution seems appealing.

The traffic load curve, which has areas with slow and rapid changing movements, served as inspiration for the notion of VSR (variable sampling rate) [21, 22]. We discovered that the main source of traffic load volatility is the timeliness features of mobile user behavior. A notable example of this is the fact that students' real-time traffic on campus is scheduled and determined by the calendar. Then, the real-time network traffic load is also limited by the number of visits and applications. When the number of users fluctuates quickly, the fast-changing zone appears, but when the number of users remains consistent, the slow-changing zone appears. Based on these findings, the best VSR strategy is to sample at a low sampling rate in slowly changing regions and at a high sampling rate in rapidly changing regions. The constant sampling rate approach is often used when considering the entire traffic load distribution, whereas the variable sampling rate strategy is used when the traffic load distribution needs to be sampled and reconstructed independently, a practical scenario where the traffic load varies in speed from region to region. The maximum frequencies in the slow-change and fast-change zones, respectively, are represented by $f^s_{\max}$ and $f^f_{\max}$. From Nyquist's theory, the sampling rate must be twice as fast as $f^s_{\max}$ in the slow-changing region and twice as fast as $f^f_{\max}$ in the fast-changing region. Therefore, the average sampling rate R^{avg}_{VSR}:

$$R^{\text{avg}}_{VSR} = \frac{\left(2f^s_{\max}T_s + 2f^f_{\max}T_f\right)}{\left(T_s + T_f\right)}, \tag{1}$$

of which T_s and T_f represent the total length of time covered by the slow and fast-change zones, respectively. Considering

$$f^s_{\max} < f^f_{\max} = f_{\max}, \tag{2}$$

we have

$$R^{\text{avg}}_{VSR} < 2f_{\max} \tag{3}$$

indicating that the sampling rate of the VSR method is lower than that of the constant sampling rate method.

4.2. VSR-NLMS Adaptive Forecasting Method. As the load profile is unknown at the time of prediction, it is not possible

to classify the load profile into low- and high-speed types. The author in [21] created the VSR-NLMS adaptive forecasting method, which combines the FSR-NLMS (fixed step size-NLSM) predictor with VSR. The sampling rate of time t_n is defined as

$$R_{s(n)} = \frac{1}{\Delta t_{n-1,n}}. \tag{4}$$

in the VSR-NLMS scheme, and it is iteratively updated to show a negative correlation with the target prediction error bound $E_b > 0$. We further have the following constraint

$$R_s^{\min} \le R_s(n+1) \le R_s^{\max}, \tag{5}$$

where, $R_s^{\max}$ and $R_s^{\min}$ denote the system's highest and lowest sampling rates, respectively.

As the flow load curve is changing rapidly resulting in large errors, the VSR-NLMS scheme is used here to update the sampling rate in time to ensure accuracy. Based on the subsequent observation, the VSR-NLMS method adjusts the sampling rate. To achieve forecast accuracy, the sampling rate must be raised because the greater the forecast error, the greater the complexity of the traffic load situation. A small prediction error, on the other hand, indicates that the traffic load profile varies slowly, allowing the sampling rate to be adjusted to reduce computing complexity. When deciding on $R_s^{\max}$, we must choose between prediction accuracy and compilation efficiency. Large $R_s^{\max}$ values, obviously, can result in reduced prediction errors, but they also increase the computational complexity of the sampling and prediction method. When we choose $R_s^{\min}$, we infer that the linear predictor's reaction time cannot be too long to handle bursty traffic.

4.3. Dimensional Transformation in Time-Series Prediction. Network traffic prediction is a temporal sequential forecasting technique, the core idea of which is to analyze the nonlinear correlation between the previous data and its historical data at a certain time point [23]. The prediction of values at a future point in time is accomplished based on the outcomes of the modeling analysis. We must use traffic flow time series modeling in this paper. Traffic flow data was converted into multidimensional data, including input feature vectors and model output sample labels in a format to better model LSTM. Scholars now use the "sliding window" approach to convert one-dimensional (1D) data into two-dimensional (2D) data. The following are the primary phases in this method for properly converting 1D time series into 2D machine learning data type.

Step 1: Choose moment T and collect N historical values before moment T and set them as feature vectors. N is the length of the feature vector.

Step 2: Construct the output vector by taking moments $T+1$ to $T+M$ as the label values. M is the number of output variables, which represents the step size of the prediction.

The basic flow chart of the "sliding window" based method is shown in Figure 2.

4.4. RNN and LSTM. The earliest known RNN is Hopfield Network (HN) proposed by Hopfield in 1982 [24]. RNN, as a neural network model that can process time series and structural series data, has been widely used in handwriting recognition, speech discrimination, text translation, and other fields. The data contained in these fields have a common feature. That is to say, the input samples are all continuous sequence data, which can be a piece of text or a piece of speech. There is a great correlation between the preceding and the following, and the length of the data is different, which cannot be accurately divided into separate training samples by traditional neural networks. Compared with the traditional neural network, the RNN model can connect the output at the current moment with the output at the previous moment so that the neural network has the function of "memory". It can record the historical sequence information, continuously reduce the gradient error between the predicted value and the real value in the process of iteration, and finally obtain the optimal model. Normally, any sequence can be obtained by predicting through the RNN model.

However, in the actual training process of the RNN model, it is found that it is still a little insufficient to store the amount of historical information. Since the RNN model stores the corresponding historical information by the number of network layers, the less the number of network layers, the more incomplete the historical information recorded. In addition, the more the number of network layers, the more complex the training process will be, and it is easy to have the phenomenon that the gradient descent speed is fast or even disappears, both of which will lead to poor prediction performance of the model [25].

To solve the above problems, the long short-term memory network (LSTM) model is proposed by Hochreiter et al. in 1997 [26]. Subsequently, the LSTM model quickly made great achievements in speech recognition and machine translation. LSTM, as a variant of RNN, can solve the common problems in the RNN model by changing the structure of the hidden layer. Since the LSTM model is obtained by changing the RNN model, the two models have the same output layer and input layer structure, and the differences are mainly reflected in the structure of the hidden layer.

The structure of the hidden layer of the LSTM model is shown in Figure 3. It can be seen that the hidden layer is mainly composed of three gating units and a memory block (cell) unit, and the three adaptive multiplication gating units are the input gating unit, forgetting gating unit, and the output gating unit, respectively.

The input gating unit I_t can be used to control, where information can be saved to the memory unit at the current moment, and it consists of two network layers with activation functions of sigmoid function σ and tan h function, respectively.

$$\begin{aligned} I_t &= \sigma\left(A_i m_{t-1} + B_i x_t + b_i\right), \\ C_t &= \tan h\left(A_c m_{t-1} + B_c x_t + b_c\right), \end{aligned} \tag{6}$$

where x_t denotes the input vector at moment t, m_{t-1} is the output of the previously hidden layer neuron node, and by

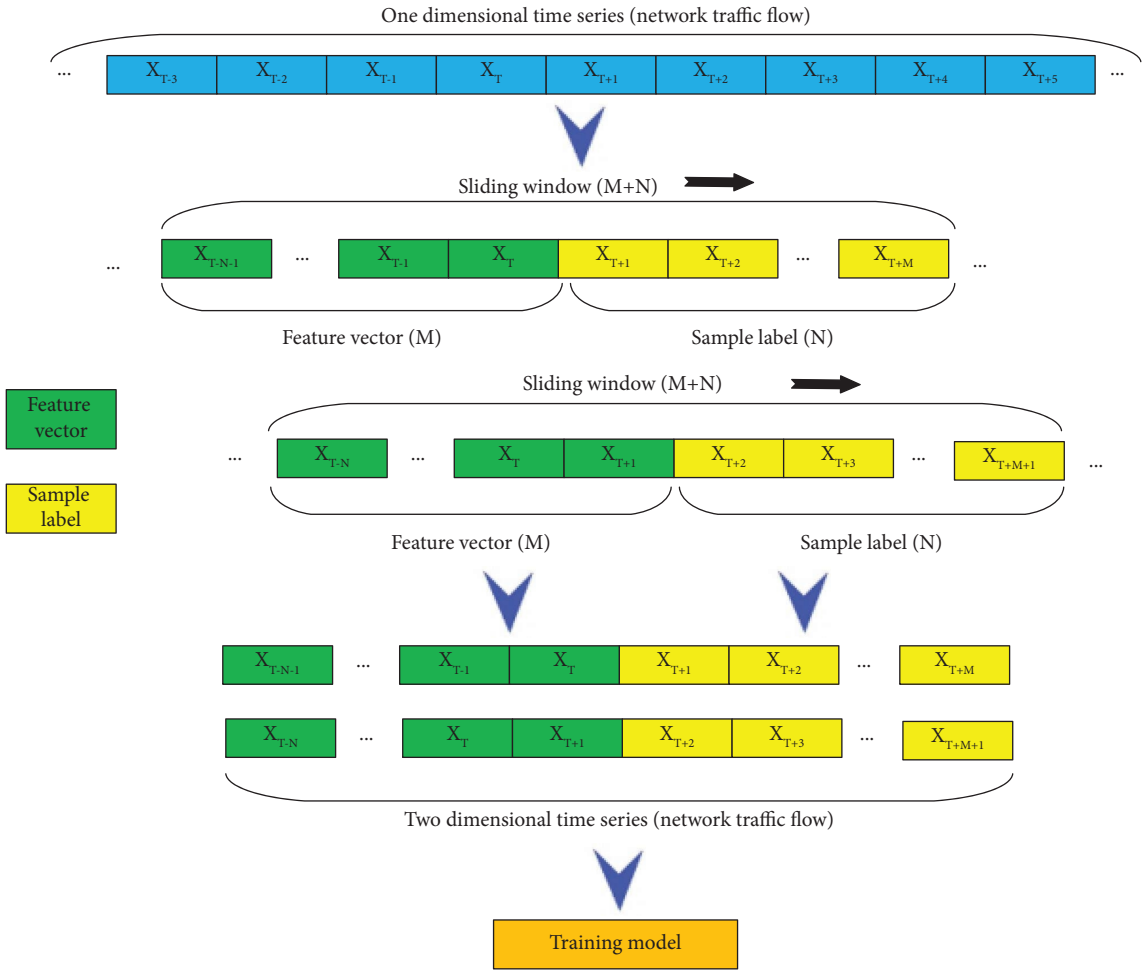

Figure 2: Time series prediction data set transformation.

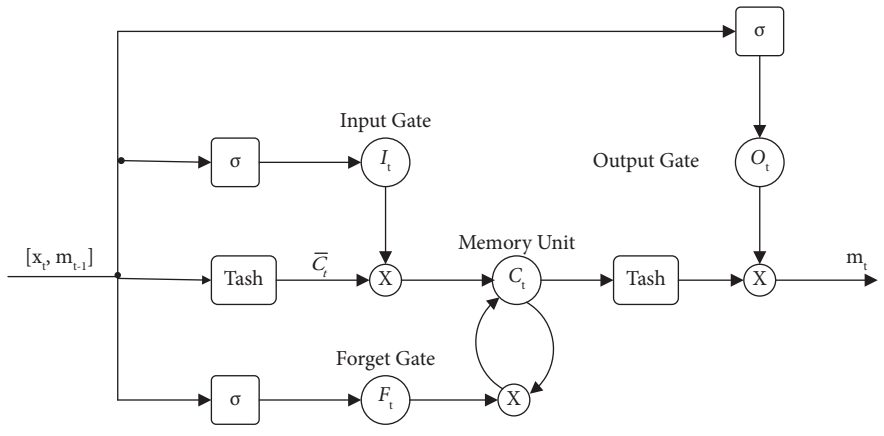

Figure 3: LSTM model hidden layer structure.

using the sigmoid function to activate x_t and m_{t-1}. A_i denotes the weight value corresponding to the output result of the previously hidden layer neuron at the input gate, while B_i is the weight value of input vector x_t and b_i is the bias parameter of the input gating unit at the time of calculation.

$\overline{C_t}$ the candidate cell state, A_c, B_c, and b_c are weight values and the bias parameter of $\overline{C_t}$ respectively.

Next, the result of multiplying the elements corresponding to the two results is used for the update of the memory block cell.

The forgetting gating unit F_t is mainly used to link the state of the memory block unit at the previous moment with the state of the memory block unit at the current moment.

The final result obtained through the forgetting gate

$$F_t = \sigma\left(A_f m_{t-1} + B_f x_t + b_f\right), \tag{7}$$

where A_f denotes the weight value corresponding to the output result of the previously hidden layer neuron at the input forgetting gate. The weight of the information entering the forgotten gating unit through the input gating unit at the current moment is denoted by B_f, and b_f is the bias parameter of the forgotten gating unit at the time of calculation.

The output gating unit is composed of two parts, the current moment input vector combined with the information obtained from short-term memory output results O_t and the output result of the information obtained by combining the input vector with long-term memory at the current moment m_t. The activation functions used are the sigmoid function and the $\tan h$ function, respectively.

$$\begin{aligned} O_t &= \sigma\left(A_\circ m_{t-1} + B_\circ x_t + b_\circ\right), \\ m_t &= O_t \circ \tan h\left(C_t\right), \end{aligned} \tag{8}$$

where, $\circ$ is the element-wise multiplication, which implements the product of the corresponding positional elements of two matrices. As mentioned above, $A_\circ$, $B_\circ$, and $b_\circ$ are weight values and the bias parameter of the O_t respectively.

The state of the memory block unit is determined by both the past moment state and the current moment state, where the past moment state is obtained by multiplying the past moment unit state with the output result of the forgetting gating unit in accordance with the corresponding element, and the current moment state is obtained by multiplying the current moment unit state with the current moment input gating unit in accordance with the corresponding element.

$$C_t = C_{t-1} \circ F_t + I_t \circ \overline{C_t} \tag{9}$$

After the above discussion, we have clearly understood the importance of controlling the sampling rate. The core of this article is to continuously adjust the sampling rate by analyzing the error. The processing method of this article is to apply the LSTM algorithm. How does the LSTM algorithm work in the past? The error value is input, forgotten, and output as a sample, which will be discussed below.

The reason why the LSTM algorithm is different from the traditional RNN algorithm is that it can delete some unnecessary data through the forget gate. Among them, A_i, B_i, and b_i in the input gating unit; A_f, B_f, and b_f in the forgetting gating unit; and A_o, B_o, and b_o in the output gating unit need to be obtained by the LSTM model through training. The training method is as follows: the error at time t is defined in LSTM as ϑ_t and the sum of squares of all node errors in the output layer is represented by L. At this time, we have the following formula

$$\begin{aligned} L &= \frac{1}{2}\sum_{t=1}^{\tau}\left(\widehat{y}_t - y_t\right), \\ \vartheta_t &= \frac{\partial L}{\partial m_t}. \end{aligned} \tag{10}$$

Among them, $\widehat{y}_t$ is the prediction value, y_t is the true value, and m_t represents the hidden state.

Then, we need to go back in time and use the error of the latter time state to calculate the error of the previous time state.

At state t_n, the error of the input gating unit is

$$\vartheta_{i_t} = \vartheta_t \circ O_t \circ \overline{C}_t \circ i_t \circ \left(1 - i_t\right) \circ \left(1 - \tan h\left(c_t\right)^2\right). \tag{11}$$

The forgetting gating unit error is

$$\vartheta_{f_t} = \vartheta_t \circ o_t \circ c_{t-1} \circ f_t \circ \left(1 - f_t\right) \circ \left(1 - \tan h\left(c_t\right)^2\right). \tag{12}$$

The output gating unit error is

$$\vartheta_{o_t} = \vartheta_t \circ \tan h\left(c_t\right) \circ t \circ \left(1 - o_t\right). \tag{13}$$

The error of the memory block unit is

$$\vartheta_{\overline{c_t}} = \vartheta_t \circ o_t \circ i_t \circ \left(1 - \overline{c_t^2}\right) \circ \left(1 - \tan h\left(c_t\right)^2\right). \tag{14}$$

Using the above four values, the error value of the state t_{n-1} can be calculated as

$$\vartheta_{t-1} = A_o \vartheta_{o_t} + A_f \vartheta_{f_t} + A_i \vartheta_{i_t} + A_{\overline{c}} \vartheta_{\overline{c_t}}. \tag{15}$$

Then, the gradient of the bias term and the weight gradient A, B, and b are derived by the chain rule, and then the gradient is updated by the gradient descent method. To be accurate enough for the calculated A, B, and b values, after training with a sufficient time-span, we use the sampling error of the previous time state E_b as the object to use the LSTM algorithm to predict. When E_b is too high, it means that the sampling accuracy is not enough and we need to increase the sampling rate; when the value E_b is too low, it means that the sampling rate is too high and too many computing resources are wasted. In this case, the sampling rate needs to be reduced.

In addition, it should be pointed out that gated recurrent unit (GRU) is also a variant of RNN, and its structure is similar to LSTM, with one less gate than LSTM, which reduces matrix multiplication and can save a lot of time without sacrificing performance in small dataset scenario. But in the scenario of large datasets, LSTM has better performance than GRU, such as prediction accuracy, recall, AUC, etc. In our study, the dataset obtained from the MEC servers is large, so LSTM with better performance is used for prediction in this paper [27].

4.5. The Proposed VSR-LSTM Model. In this paper, we make full use of the excellent advantage of LSTM in time series prediction, which combines the idea of variable sampling rate, improves the VSR-NLMS algorithm, proposes a network highway traffic prediction model based on LSTM variable sampling rate, named as VSR-LSTM, which contains the following three main parts.

(1) Transform the original data form and divide the training set and test set.

(2) Train the LSTM neural network.

(3) Realize traffic flow prediction and verify the results.

The model's fundamental stages are listed below.

Table 3: Summary of notations in Section 3.

Symbol	Description
T_f	The total length of time covered by the fast-change zones
T_s	The total length of time covered by the slow-change zones
$f^s_{\max}$	Maximum frequencies in the slow-change
$f^f_{\max}$	Maximum frequencies in the fast-change
R^{avg}_{VSR}	Average sampling rate
Rs(n)	The sampling rate of time t_n
E_b	Target prediction error bound
$R^s_{\max}$	The system's highest sampling rates
$R^s_{\min}$	The system's lowest sampling rates
O_t	The output of the model at instant t
I_t	Input gating unit at instant t
F_t	Forgetting gating unit at instant t
C	The internal state
m_t	The hidden state
x_t	Input vector at moment
ϑ_t	The error at time t is defined in LSTM
A_c, A_f, A_o	The weight values of x_t
B_c, B_f, B_o	The weight values of m_{t-1}
b_c, b_f, b_o	The bias parameters
L	The sum of squares of all node errors in the output layer

Step 1: Based on the actual collected traffic flow data, the training set and test set are divided in the ratio of 7 : 3.

Step 2: Based on the basic principle of "sliding window" method, the original one-dimensional traffic data is converted into two-dimensional data.

Step 3: Preprocess the raw data by normalization and other methods.

Step 4: Input the training data into the LSTM network, train the model parameters, and build the prediction model.

Step 5: Input the input vectors of the test set into the trained LSTM neural network and compare the prediction results with the real values to get the error of the model.

The summary of notations in Section 3 is listed in Table 3.

5. Numerical Results

In this section, we implemented the proposed VSR-LSTM and compared it with the traditional fixed sampling rate algorithm FSR-NLSM. Tto facilitate performance comparison, the original fixed sampling step size algorithm FSR-NLMS is also implemented based on LSTM, namely FSR-LSTM.

5.1. Evaluation Setup. We assess the performance of the suggested architecture using a collection of mobile traffic statistics from ten separate MEC servers that we gathered over the course of 1 month. According to Section 4.5, we determine the aggregate cell traffic for each server. We use normalized root mean square error (NRMSE) as a measure of the prediction algorithm's efficacy, which is defined as

$$\text{NRMSE} = \frac{1}{x}\sqrt{\frac{\sum_{t=1}^{N}\left(\widetilde{x}_t - x_t\right)^2}{N}}, \tag{16}$$

where $\widetilde{x}_t$ and x_t are the predicted value and its corresponding observation at the time t, respectively, and $\overline{x}_t$ is their mean. N is the total number of points. The accuracy of the suggested architecture is compared using the same metric with that found using other prediction algorithms.

Table 4: Training hyperparameters.

Hyperparameters name	Value
Initial learning rate	0.001
Number of epochs	100
LSTM hidden states	64
LSTM hidden layers	5
Optimization algorithm	Adam
Loss function	MAE

The simulation is implemented using Python, and the backend uses TensorFlow and Keras. Table 4 reports the selected hyperparameters. One of the hyperparameters that must be chosen that might influence the trade-off between the amount of time and the prediction accuracy required to train the network is the number of hidden layers, which is fixed at 5. The amount of information that must be maintained and utilized by the network is determined by the connection between the quantity of earlier observed values and the accuracy of the multistep prediction, which is the focus of our attention. The prediction accuracy could be enhanced by adding more layers. We set the total number of epochs to 100 for the same reason. The architecture was trained and validated using three weeks of data. The following results relate to the previous week. We iteratively change the network weights based on the training data using the Adam optimization.

5.2. Results' Analysis. The results of multistep prediction, which involves making predictions for future time instants while delaying the output by a predefined number of timeslots, are then shown. We demonstrate how accuracy suffers when we attempt to forecast traffic statistics for upcoming timesteps. We also look at how the length of the timeslots T and the number of observations that the LSTM network can observe affect the results. These design parameters must be computed since they have an impact on the LSTM network's memory capacity and the amount of traffic data that must be kept for an accurate prediction.

Finally, we compare the proposed algorithm, i.e., VSR-LSTM with a classical time-series network traffic prediction method (FSR-LSTM). For a fair comparison, the same number of hidden layers are used.Figure 4 illustrates the traffic prediction for the same time-span using both methodologies. Using the FSR-NLMS model has lower accuracy because the predictions tend to be closer to the mean of the flow, while VSR-LSTM has a higher flow prediction

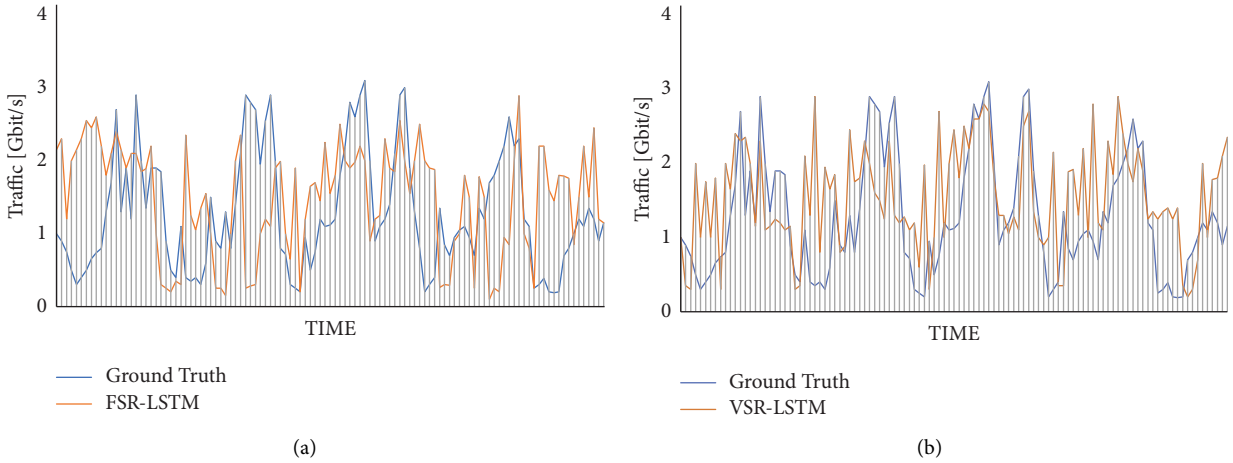

Figure 4: Traffic prediction carve obtained with different models. (a) FSR-LSTM. (b) VSR-LSTM.

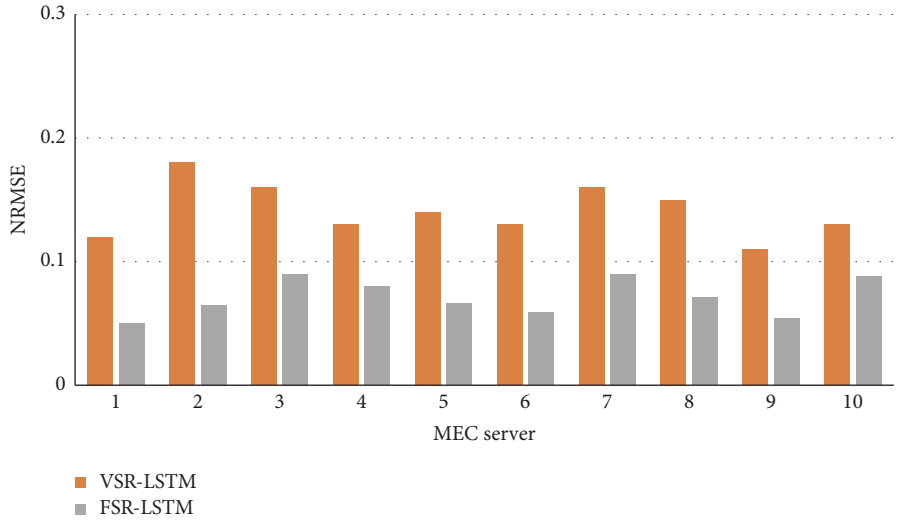

Figure 5: Traffic prediction errors obtained with different models.

accuracy because its sampling speed increases with changes in flow. Furthermore, for the two prediction methods on the 10 flow profiles, we compare their mean errors. As expected, the proposed algorithm obtains less prediction error of the moving flow due to the VSR properties and relative to the FSR-LSTM model, as shown in Figure 5.

The difference in computational complexity between the proposed VSM-LSTM and the baseline FSR-LSTM is mainly due to the difference in sampling rate. Based on the previous analysis in Section 4.1, the average sample rate R_{VSR}^{avg} in VSR-LSTM is lower than the fixed sample rate in FSS-LSTM. For example, if we suppose the $f_{\max}^{f}$ in the traffic load curve is twice the $f_{\max}^{f}$ and also assume $T_s = T_f$, then, according (1), R_{VSR}^{avg} in VSR-LSTM is 25% lower than the fixed sample rate in FSS-LSTM.

6. Conclusions

In this paper, we propose a network traffic prediction method with a variable sampling rate using machine learning and LSTM techniques. In the variable sampling rate case, the sampling rate that determines the accuracy of traffic prediction can change in real time with the dynamic changes of network traffic. Therefore, compared with the traditional traffic prediction methods based on fixed sampling rate, the traffic prediction method proposed in this paper can more accurately reflect the real-time changes of network traffic to further guide the reasonable and effective allocation of network resources.

In the next work, based on the traffic prediction method with variable sampling rate proposed in this paper, we will further investigate the intelligent distributed allocation and management of multidimensional resources for the different demands on resources such as computation, communication, and storage in 6G MEC IoTs networks for different vertical applications in the future.

Furthermore, we should also try to combine our proposed algorithm with Markov chains, hidden Markov models, and other classic prediction theories to improve the prediction accuracy, computational complexity, and other

performances, especially on small granular flow performance, such as a packet, a mobile app, etc. Meanwhile, we also try to extend the application of the novel prediction algorithm to new scenarios such as 6G, digital twin, and metaverse in the future.

Data Availability

The dataset used to support the findings of the study can be obtained from a private company.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Open Foundation of State Key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications) (Funding no. SKLNST-2020-1-10).

References

[1] I. Khan, X. Tao, G. M. S. Rahman, W. U. Rehman, and T. Salam, "Advanced energy-efficient computation offloading using deep reinforcement learning in MTC edge computing," *IEEE Access*, vol. 8, pp. 82867–82875, 2020.

[2] X. Liu, J. Yu, J. Wang, and Y. Gao, "Resource allocation with edge computing in IoT networks via machine learning," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3415–3426, April 2020.

[3] M. Merluzzi, P. D. Lorenzo, and S. Barbarossa, "Wireless edge machine learning: resource allocation and trade-offs," *IEEE Access*, vol. 9, pp. 45377–45398, 2021.

[4] J. Han, G. H. Lee, S. Park, and J. K. Choi, "Joint subcarrier and transmission power allocation in OFDMA-based WPT system for mobile edge computing in IoT environment," *IEEE Internet of Things Journal*, vol. 9, no. 16, pp. 15039–15052, 2022.

[5] Z. Chen, J. Hu, X. Chen, J. Hu, X. Zheng, and G. Min, "Computation offloading and task scheduling for DNN-based applications in cloud-edge computing," *IEEE Access*, vol. 8, pp. 115537–115547, 2020.

[6] P. W. Khan, K. Abbas, H. Shaiba, A. Muthanna, A. Abuarqoub, and M. Khayyat, "Energy efficient computation offloading mechanism in multi-server mobile, edge computing—an integer linear optimization approach," *Electronics*, vol. 9, no. 6, p. 1010, 2020.

[7] I. Lohrasbinasab, A. Shahraki, A. Taherkordi, and A Delia Jurcut, "From statistical- to machine learning based network traffic prediction," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 4, Article ID e4394, 2022.

[8] T. Zhang, L. Feng, P. Yu, S. Guo, W. Li, and X. Qiu, "A handover statistics-based approach for cell outage detection in self-organized heterogeneous networks," in *Proceedings of the 15th IFIP/IEEE International Symposium on Integrated Network and Service Management (IM '17)*, IEEE, Lisbon, Portugal, pp. 628–631, May 2017.

[9] Y. Wang and T. Nakachi, "Prediction of network traffic through light-weight machine learning," *IEEE Open Journal of the Communications Society*, vol. 1, pp. 1919–1933, 2020.

[10] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, "Deep learning with long short-term memory for time series prediction," *IEEE Communications Magazine*, vol. 57, no. 6, pp. 114–119, 2019.

[11] H. D. Trinh, L. Giupponi, and P. Dini, "Mobile traffic prediction from raw data using LSTM networks," in *Proceedings of the 2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pp. 1827–1832.

[12] G. Aceto, G. Bovenzi, D. Ciuonzo, A. Montieri, V. Persico, and A. Pescape, "Characterization and prediction of mobile-app traffic using Markov modeling," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 907–925, 2021.

[13] A. Montieri, G. Bovenzi, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescape, "Packet-level prediction of mobile-app traffic using multitask deep learning," *Computer Networks*, vol. 200, no. 2021, Article ID 108529, 2021.

[14] S. Zeb, M. A. Rathore, A. Mahmood, S. A. Hassan, J. W. Kim, and M. Gidlund, "Edge intelligence in softwarized 6G: deep learning-enabled network traffic predictions," in *Proceedings of the 2021 IEEE Globecom Workshops (GC Wkshps)*, December 2021.

[15] T. Ko, S. M. Raza, D. T. Binh, M. Kim, and H. Choo, "Network prediction with traffic gradient classification using convolutional neural networks," in *Proceedings of the 2020 14th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, pp. 1–4, Taichung, Taiwan, January 2020.

[16] J. Kwon, D. Jung, and H. Park, "Traffic data classification using machine learning algorithms in SDN networks," in *Proceedings of the 2020 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 1031–1033, Jeju, Korea, October 2020.

[17] H. Lu and F. Yang, "Research on network traffic prediction based on long short-term memory neural network," in *Proceedings of the 2018 IEEE 4th International Conference on Computer and Communications (ICCC)*, pp. 1109–1113, Chengdu, China, December 2018.

[18] J. Fan, D. Mu, and Y. Liu, "Research on network traffic prediction model based on neural network," in *Proceedings of the 2019 2nd International Conference on Information Systems and Computer Aided Education (ICISCAE)*, pp. 554–557, Dalian, China, September 2019.

[19] J. Kaur, M. A. Khan, M. Iftikhar, M. Imran, and Q. Emad Ul Haq, "Machine learning techniques for 5G and beyond," *IEEE Access*, vol. 9, pp. 23472–23488, 2021.

[20] H. Yang, A. Alphones, Z. Xiong, D. Niyato, J. Zhao, and K. Wu, "Artificial-intelligence-Enabled intelligent 6G networks," *IEEE Network*, vol. 34, no. 6, pp. 272–280, November/December 2020.

[21] B. Rong, S. Sun, and M. Kadoch, "Traffic prediction for reliable and resilient video communications over multi-location WMNs," *Journal of Network and Systems Management*, vol. 24, no. 3, pp. 516–533, 2016.

[22] S. Wu, W. Mao, C. Liu, and T Tang, "Dynamic traffic prediction with adaptive sampling for 5G HetNet IoT applications," *Wireless Communications and Mobile Computing*, vol. 2019, pp. 1–11, 2019.

[23] Y. Zhao, "Highway traffic flow prediction based on simple recursive unit network," *China's transportation information*, no. 02, pp. 123–126, 2022.

[24] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities"," *Proceedings of the National Academy of Sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[25] K. Om, S. Boukoros, A. Nugaliyadde et al., "Modelling email traffic workloads with RNN and LSTM models," *Human-centric Computing and Information Sciences*, vol. 10, no. 1, pp. 39–13, 2020.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] S. Yang, X. Yu, and Y. Zhou, "LSTM and GRU neural network performance comparison study: taking yelp review dataset as an example," in *Proceedings of the 2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*, pp. 98–101, Shanghai, China, June 2020.