Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 5366222, 12 pages
https://doi.org/10.1155/2021/5366222

Research Article

Prediction of Traffic Generated by IoT Devices Using Statistical Learning Time Series Algorithms

Shilpa P. Khedkar,[1] **R. Aroul Canessane**,[1] **and Moslem Lari Najafi**[2]

[1]*Department of Computer Science and Engineering, Sathyabama Institute of Science and Technology, Chennai, India*
[2]*Pharmaceutical Science and Cosmetic Products Research Center, Kerman University of Medical Sciences, Kerman, Iran*

Correspondence should be addressed to Moslem Lari Najafi; m.larinajafi@kmu.ac.ir

Received 19 June 2021; Accepted 19 July 2021; Published 2 August 2021

Academic Editor: VIMAL SHANMUGANATHAN

An IoT is the communication of sensing devices linked to the Internet in order to communicate data. IoT devices have extremely critical reliability with an efficient and robust network condition. Based on enormous growth in devices and their connectivity, IoT contributes to the bulk of Internet traffic. Prediction of network traffic is very important function of any network. Traffic prediction is important to ensure good system efficiency and ensure service quality of IoT applications, as it relies primarily on congestion management, admission control, allocation of bandwidth to the system, and the identification of anomalies. In this paper, a complete overview of IoT traffic forecasting model using classic time series and artificial neural network is presented. For prediction of IoT traffic, real network traces are used. Prediction models are evaluated using MAE, RMSE, and R-squared values. The experimental results indicate that LSTM- and FNN-based predictive models are highly sensitive and can therefore be used to provide better performance as a timing sequence forecast model than the conventional traffic prediction techniques.

1. Introduction

The Internet of Things (IoT) are communicating devices with sensing capability that are connected to the Internet which enables collecting and sharing of data without human intervention. Initially, RFID tags were added to devices to help track their location, and this was one of the first IoT applications. As things are made to communicate with one another, such kind of communication is called Machine-to-Machine (M2M) communication. Communication can be short range with mobile networks like 3G, 4G, LTE, and 5G, such as Wi-Fi, Bluetooth, and broad range [1]. The main tasks to be performed by IoT devices are data collection, M2M communication, and application processing. IoT devices are meant to handle massive amount of data; therefore, its infrastructure needs to implement methods that can manage, hold, and evaluate large data. In order to enable Machine-to-Machine (M2M) communication, IoT platforms are used along with protocols such as MQTT and CoAP [2–4]. IoT devices may include services like monitoring, node managing, data storage and evaluating capabilities, etc. IoT devices are mainly used in applications such as Health Care Systems, Home Appliances, Transportation, Monitoring of Environmental Conditions, Logistics Management, Security services, and many more. The IoT infrastructure consists of (1) IoT nodes which are mainly called the IoT devices which consists of sensors and actuators; (2) servers which are otherwise called Fog nodes that facilitate storage, computing, and networking; (3) data centers called cloud nodes that store data using machine learning techniques and help in data sharing; and (4) IoT applications that make use of the computed data to provide services to the end user. In the Internet of Things (IoT), individuals, systems, data, and objects have been predominantly linked to and connected with the Internet. The effectiveness of any IoT ultimately depends on how easily it can interact with other IoT devices. Their performance is assessed. Speed increase helps to minimize delays and enhance the communication of data between devices. For any IoT system, which relies on real-time updates, having consistent and stable network conditions is extremely essential. The network's ability to deal with more connected devices will help the devices become more reliable.

IoT will contribute to the major part of Internet traffic. Research contributions related to IoT and network traffic are limited to modeling IoT traffic and characterizing its properties. IoT traffic characteristics depend on the types of sources, hardware devices, and services provided as it is the combination of different devices at different environments. Traffic generators, for instance, Iperf and D-ITG, are used in evaluating network performance and characterizing traffic. Different types of interfaces and transport protocols are used by the traffic generators. A traffic generator injects traffic into the network for utilization by other devices. Traffic generator facilitates in evaluating device performance. The IoT usage is rapidly increasing, but a few research studies have been carried out regarding traffic characterization. By visualizing IoT traffic with the t-distributed stochastic neighbor embedding (t-SNE) technique, flowing data analytics to perceive anomalous IoT traffic by visually observing the traffic through sensing devices with the BS graphs are some of the techniques used in traffic characterization [5]. Machine learning algorithms are employed for identifying and classifying IoT devices. Artificial intelligence (AI) is used in the prediction of network traffic for data networking. Traffic size is predicted on the basis of the count of input and output bytes during communication at various network levels. For network operations, management, and other network functionalities and roles, traffic prediction is mandatory. IoT traffic can be broken down into three forms of traffic: deterministic, probabilistic, and facilitated. The traffic from the IoT devices and other communications networks can be transported. Although the traffic from M2M connections was historically smaller than that from end users. Currently, traffic has grown faster than the number of connections as video applications on M2M connections have been deployed quickly, and IoT applications have become more frequent and demand more bandwidth and lower latency.

Devices and their connectivity have increased enormously as well as people and Internet users. This leads to a rise in average per household and per capita devices and connections. Each year, new devices are launched and implemented on the market from a variety of sources and heterogeneous hardware components with increased capacities and intelligence. The development of these devices and connections is greatly influenced by an increasing number of M2M applications. Therefore, the shifting mix of devices and connections must be monitored, and their various ownership affects the traffic patterns differently. One of the main contributors to global mobile traffic growth is the ever-changing combination and increase in wireless devices accessing mobile networks worldwide. Enhancing system capabilities through faster, higher bandwidth, and smarter networks will allow broad experimentation and the expansion of new multimedia applications to help increase mobile and Wi-Fi traffic. The rise in the usage of mobile apps and the growth of mobile access in scope are increasing the number of end users. The need for optimized bandwidth management and new network monetization models has been brought about. This is why the growth of mobile IoT brings individuals, systems, data, and items together to improve the relevance and value of network connections and thus to increase network traffic contribution [6]. Most of the infrastructure of IoT devices consists of sensors, processors, network software, and different vendors' applications. A gateway often comprising products of several suppliers controls the endpoints. The endpoints are different for different applications. The basic functions of each application produce different types of traffic in return. Different variables such as whether a processing gateway is passed through, how much intelligence it includes, and the applications run on the gateway depends on traffic. Intelligent gateway awareness can react locally and process most generated data while dumb devices need more connectivity to a data center or to a cloud service. The characteristics of traffic depend on flow, data collection, transmission of data, and communications amongst actuators and sensors.

There are several IoT devices linked by wireless. Rising online device volumes would require more bandwidth. As IoT grows, ensuring it can handle its devices and traffic through the network is compulsory. As the number of devices increases, IoT can increase its effect on the bandwidth needs. As the technology develops more, the amount of data linked to IoT devices will increase, which in turn will help to improve the bandwidth requirement. Bandwidth will still be sufficient for the end users. The major problem of effective IoT technology deployment is related to the speed and coverage of wireless networks available (Wi-Fi). Another important factor to IT traffic is the rise in broadband speed. The network demands are far higher than IP voice and video. The distribution of bandwidth in a congested network is difficult to handle. Improvements to broadband speed result in increased usage and use of content and software with high bandwidth. The global average speed of broadband is still growing and will double from 45.9 Mbps to 110.4 Mbps from 2018 to 2023. Implementation and use of fiber to the home (FTTH), high-speed DSL, and broadband cable adoption as well as broadband penetration in general are key factors influencing the broadband fixed forecast [6].

As mentioned earlier, IoT is liable for the interaction amongst different devices. With the rise in smartphones, IoT traffic on the Internet has significantly increased, and a lot of research has been done to enhance the end user's connectivity experience. The component to predict traffic has crucial effects on dynamic negotiation of bandwidths. Two independent sections of the method are normally sampled and predicted. To ensure good system efficiency, traffic prediction is vital. To use network resources efficiently, traffic prediction is essential. Congestion management, induction control, network assignment of bandwidth, and detecting malicious applications are based on accurate predictions of traffic at end points. This allows for effective distribution of resources to ensure consistent service quality. Network abuses are preventing and detecting increasingly difficult with increasing traffic and network complexity. The adequacy of traffic forecast is directly linked to this. The quality of service provided by wireless sensor networks depends on the supply of power. A precise traffic prediction is needed to speed up these nodes and to maximize energy savings. With rising requirements for demands in traffic and computational needs, power consumption is increased [7].

The contributions of the paper are listed as follows:

(i) The related techniques for traffic prediction are explored and implemented for comparative study of the existing methods with the proposed LSTM method and feedforward neural networks (FNN) prediction method

(ii) To use the network bandwidth optimally, to reduce the over consumption of the IoT channels, FNN and LSTM are proposed which are more efficient than the existing ARIMA- and VARMA-based techniques

(iii) The problem of insufficient data of IoT devices and unavailability of historical data, a time series-based learning model has also been introduced in this paper. By learning from the accumulated data from similar domains, the LSTM and FNN methods allow the predictive models to get trained from accumulated data and application of time series-based learning model in the local domain

The rest of the paper is structured accordingly. Section 2 outlines the previous work on the forecasting of IoT traffic. In Section 3, the IoT traffic forecasting models used are presented in brief. Section 4 describes the evaluation methodology with the experimental findings. In Section 5, concluded with future trends.

2. Related Work and Research Gap

The rapid growth of network traffic leads to enormous research contributions based on prediction of the network traffic, but the Internet of Things requires improved and efficient approaches to handle huge and dynamic data as shown in some recent study [2, 7]. We have surveyed the different techniques used in network traffic prediction in this section to provide insights into the research contributions made by others in the area of our research work.

The author proposes a model which uses spatiotemporal features from neighboring cellular stations to target base station. Traffic prediction is done with the help of these features over time using deep learning algorithms like 3D convolutional networks. The methodology used showed promising results that outperformed other traffic prediction systems [8]. Essien et al. proposed a model for urban traffic prediction using deep learning architecture which works on features extracted from traffic and weather-related tweets. This model helps in multistep traffic prediction which includes deep Bidirectional Long Short-Term Memory (LSTM) stacked autoencoder (SAE) architecture. The results obtained showed to be effective in improving prediction accuracy when compared to other traditional statistical and machine learning models. This model was found to be useful to reduce frustration due to heavy road traffic, low cost and increased savings for businesses, and environmental sustainability [9].

Volkov et al. proposed an IoT traffic prediction model using NARX neural network with a single step ahead and multistep ahead prediction. The evaluation of prediction was done using Traincgf, Traincgp, and Trainlm neural network training algorithm along with forecasting accuracy measures like mean square error (MSE) and mean absolute percentage error (MAPE) [10]. Meena et al. proposed a tool used for accurate traffic flow prediction. They used machine learning, genetic, soft computing, and deep learning algorithms to build a model which does big-data analysis with reduced complexity for the transportation system [11]. Feng et al. proposed a traffic prediction end-to-end model using deep-learning approaches. The model uses spatial-dependent and long-period cellular traffic for forecasting traffic. The model comprises a feature extractor that models spatial dependencies and encodes the external information. It also models complicated temporal variations using a sequential module. Spatial modeling is done using a correlation selection mechanism, and encoding of external information is done by embedding mechanism. The attention mechanism is applied to the seq2seq model to build the sequential model. The observation of results shows that this model outperforms other traditional prediction models by more than 12.31 percent [12].

Hua et al. proposed the Random Connectivity LSTM, an LSTM-based deep learning model (RCLSTM). RCLSTM is used to forecast traffic and confirm the satisfactory efficiency of the RCLSTM with even 35% neural connectivity, and its prediction accuracy is even higher than the basic LSTM [13]. Wang et al. [14] suggest an optimized model for prediction of space-temporal traffic flow. In comparison to traditional coevolutionary neural networks, the GCN's applications have a more robust and accurate capacity to learn spatial features of the road system and the use of LSTM with the introduction of weather and regular features as additional knowledge increases its strength in temporary feature extraction. The test results showed the model's capacity to capture spatial and temporal features [14]. Zhang et al. suggest a way of bridging the gap between deep learning and mobile and wireless networking research by the adoption of a systematic crossover of the two fields. This survey reveals the issues facing currently and the potential of wireless network research [15].

In the time series NARX feedback neural networks with multistepped ahead prediction, Abdellah et al. proposed an IoT-prediction model traffic time series. The prediction was assessed using reliability assessment functions such as MSE, SSE, MAE, and the mean absolute percentage of error (MAPE) [16]. Essien et al. proposed a method for single-step time series prediction using a combination of deep study methods, such as Wavelet Transforms (WT) and 2-dimensional CNN's and Long Short-Term Memory (LSTM) stacked autoencoders (SAE). In the univariate time series prediction, the model had positive results [17]. Kamble and Kounte proposed to use machine learning algorithms (ML) to identify traffic congestion using parameters such as difficult timing limitations and the speed available via GSP trajection. This paper shows that approaches to machine learning (ML) can be useful for prediction of current and historical data in real-time traffic and future traffic and short-term traffic. There were three separate time slots used for tracking traffic congestion that contributed to the assessment of vehicle average speed [18].

Tang et al. proposed to predict future TL and network involvement with a new deep learning system with a TL prediction algorithm. A new intelligent channel assignment algorithm was considered to be a deep learning-based predictive method with partially overlapping channel assignments. The proposed algorithm smartly prevents future congestion and helps to rapidly allocate appropriate channels for SDN-IoT. It is observed that the results were much better than other traditional channel algorithms [19]. A traffic congestion control system was suggested by Nguyen et al. Different functions of the proposed model include collection of data, building-up of structures, modelling of traffic flow, congestion prediction, and congestion services. The use of a verification method based on the Rankine-Hugoniot condition minimizes false predictions. The experimental model analyses showed that traffic congestion can be effectively tracked in terms of precision and network response time [20].

In deep learning methods, Polson and Sokolov suggested a combined architecture. The architecture combines a linear model that is equipped with $\ell 1$ regularization with a series of tanh layers. In order to capture spatio-temporary effects and short-term traffic forecasts, the proposed model was followed [21]. Huang et al. introduced a multitask learning architecture using mobile traffic prediction deep learning networks. Three traditional algorithms for deep learning, RNN, and 3D CNN have been used. The CNN-RNN algorithm was designed with different purposes to collect mobile traffic resources. The experimental observation shows that the customized CNN and RNN can be used effectively to obtain geographic and time traffic characteristics. The comparative study shows that CNN-RNN reliably track traffic with an accuracy of 70% to 80% [22]. Li et al. introduced a methodology which used traffic big data analysis. There was a systematic analysis of the modelling and estimation of mobile network traffic. A new traffic prediction has been created to incorporate and explore more of these characteristics using a dictionary learning-based alternative direction method system. Experimental results of the established model showed that the modelling and forecasting problems in application-level learning of traffic and prediction were significantly solved [23].

Luo et al. suggested a model with more than 9 layers of a deep convolutive neural network. The proposed models with the support of a deep convolution neural network have shown to be more powerful in terms of prediction accuracy than other typical vehicle recognition systems based on machine learning algorithms [24]. Xu et al. proposed a method for mobile traffic extraction and modelling. It has developed, implemented, and assessed the model using a time series analysis method that decomposes mobile large-scale traffic based on component regularity and randomness. The experimental proves that the regularity variable is highly predictable and does not predict the average component [25].

Though much research explores different methodologies for predicting, classifying network traffic for the Internet of Things is still to be explored. Traditional prediction and classification techniques to forecast IoT-based traffic and provide data security and integrity will not be able to give satisfying performance and accurate results. This leads to the use of newer AI-based intelligent techniques which are suitable to the IoT environment. Therefore, we are proposing machine learning methods to predict the IoT-based traffic.

3. Data Preparation

Data has to be prepared as per the model selected. The two steps for data preparation are by firstly considering the quality of the data and then organizing the data in a particular way for a time series forecasting context. These two steps of data preparation are elaborated below and process see in Figure 1.

3.1. Data Preprocessing. Abnormal measurements (e.g., due to problems with a measuring sensor) has to be removed. The abnormal measurements might trick the model into learning patterns that it should not, which results in overfitting the training data. Hence, it is imperative to visualize the data in different ways to find abnormal measurements and fix it. Data has to be normalized as per the ML models used. For example, all features of the dataset may to be normalized to a particular scale (e.g., all values between 0 and 1) as shown in

$$x_{\text{std}} = \frac{x - x_{\min}}{(x_{\max} - x_{\min})}, \tag{1}$$

$$x_{\text{scaled}} = x_{\text{std}} * (x_{\max} - x_{\min}) + x_{\min}. \tag{2}$$

Recent studies focus on applying machine learning techniques to time series forecasting through supervised learning. For this purpose, the data has to be prepared before feeding it as input to some machine learning model. Also, predicting a value in a time series is based on many previous values in the same time series, e.g., $y_T = f(x_1, x_2, \cdots, x_{T-h})$. Therefore, in order to give the model some temporal context, the inputs must be sequences of feature vectors and not just a single feature vector. The moving window approach is applied during the learning phase as the machine learning model expects many pairs of inputs and outputs, the number of sets n be contingent on the number of windows the time series is split up into which is evaluated to be $T - s - h + 1$, where the window size is represented as "s" and forecasting horizon as "h." The direct strategy for multistep forecasting is applied by training different models for each value of h [26].

The model will iterate through the mappings window by window and learn the temporal patterns throughout the time series. By the end of the learning phase, the model will be an approximation of the function f as shown below:

$$y(t+h) = f\left(x_{(t-n+1)} x_{(t-n+2)}, \cdots, x_{(t)}\right), \forall t \in \{n, n+1, \cdots, T-h\}, \tag{3}$$

where f requires new inputs to also be sequences of size n . The output similar to the trained sequence during the learning phase will be produced when the new sequence of n inputs is fed into the model. The output will not be accurate if the input sequence does not resemble the learning phase inputs. This shows that the training data which was used

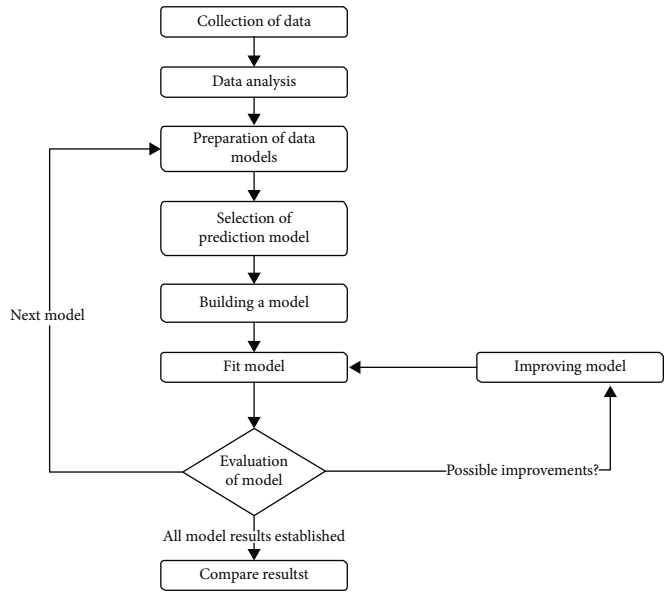

Figure 1: Flow of the proposed model.

for generalizing to the data in the test set has abnormal measurements. Machine learning models depend on many fixed parameters which do not directly belong to the model itself and remain constant throughout the learning phase. These are called hyperparameters. Tuning the hyperparameters helps in avoiding a poorly performing model. Optimization requires a trial-and-error approach which is usually a difficult task. A technique called grid search is often used to automate the process of empirically finding the best combination of hyperparameters [27].

3.2. Forecasting Models

3.2.1. ARIMA. ARIMA is one of the most commonly used time series prediction models and is a widely used statistical technique. Exponential smoothing is also used in time series forecasting in addition to the ARIMA model. Exponential smoothing model relies on trend and seasonality in data while ARIMA is responsible for describing the auto correlation in the data and for predicting future values for a prediction based on past time series values. The Auto-Regressive Integrated Moving Average (ARIMA) model, it is a linear model of regression using its own lags as predictors.

There are two kinds of ARIMA model that can be used for forecasting which are seasonal and nonseasonal. The ARIMA model norm is given as an ARIMA(x, y, z), where x is the order of the AR term, y is the order of the Moving Average term, and z is the number of differentials that are necessary to stabilize the time series [28]. A time series property like mean and time variance is known stationary. The time series must be stationary in the ARIMA model. This concerns the disadvantage of using this prediction model because extremes, typical in traffic data sets, appear to be missed [29]. The previous value must be removed from the current value to make the sequence stationary. The complexity of the series therefore determines how many differences are necessary. Thus, the d value is the minimum number of differences needed to stabilize the series, and if the time series is already stationary, then $z = 0$. The series should be overdifferentiated, because it affects the parameter of the model. The "x" parameter is the lag order that shows the number of lag observations in the model. The parameter "y" is called the order that shows the moving average window size.

This model is defined by the differencing

$$C_t = \phi_1 c_{t-1} + \phi_2 c_{t-2} + \cdots \phi_x c_{t-x} + a_t - \theta_1 a_{t-1} + \cdots - \theta_y a_{t-y}. \quad (4)$$

The model's Equation (4) describes predicted C_t = Constant + Linear grouping Lags of C (upon x lags) + Linear Forecast error grouping (upon y lags),

where c is an overall sequence of times, C_t marks the time series forecast t, $c_{t-1\cdots ct-p}$ is the previous time series $p$values, $\phi_1 \cdots \phi_p$ is the coefficients for model fitting to be defined, $a_t \cdots a_{t-x}$ zero indicates white noise and is the stirring avg. term, $\theta_1 \cdots \theta_{t-x}$ is the coefficient for model fitting to be determined, x is the autoregression term numbers, and y is stirring avg. term numbers.

The combination of AR and MA with their respective x and y parameters defines the operation, known as ARIMA(x, y), and is defined in Equation (5) as follows:

$$C_t = \mathrm{AR}(x) + \mathrm{MA}(y). \quad (5)$$

A pure AR model is where C_t relies only on its own lags, while C_t relies exclusively on the lagged prediction errors.

3.2.2. VARMA. For the production of linear forecasts for sets of time series variables, vector autoregression moving average models (VARMA) are used. ARIMA is generalized to several parallel time series. Linear forecasts are used in the assessment of nonlinear characteristics as a benchmark.

Consider K associated time series, $y_t = (y_{1t}, \cdots, y_{kt})$: the vector autoregressive (VAR) form [30] in Equation (6) can

be a model for the conditional mean of data generation process (DGP) of the observed series.

$$y_t = A_1 y_{t-1} + \cdots + A_p y_{t-p} + u_t, \tag{6}$$

where the coefficient matrices of $A_i (i = 1, \cdots, p)$ is $(k \times k)$ and u_t is a term k-dimensional error. If u_t is time independent, the conditions of y_t are in Equation (7) given past observations

$$y_{t|t-1} = E(y_t \mid y_{t-1} \cdots) = A_1 y_{t-1} + \cdots + A_p y_{t-p}. \tag{7}$$

This model is therefore suitable for y for forecasting a time ahead and repetitively measuring predictions with wider horizons.

The VARMA modelling technique allowed for the modelling of many dependent time series, both cross-relationships and intercorrelations. Assume that z_t is a stationary k-dimensional presentation sequence, then the VARMA(x, y) models used in the forecast for each group can be defined as Equation (8) [31].

$$\phi(B) z_t = \phi_0 + \theta(B) a_t, \tag{8}$$

where "t" represents period and ϕ_0 is a constant vector

$$\phi(B) = I_k - \sum_{i=1}^{q} \phi_i B_i, \tag{9}$$

$$\Theta(B) = I_k - \sum_{i=1}^{q} \theta_i B_i. \tag{10}$$

Two polynomial matrices of the $x > 0$ and $y > 0$ order, respectively, are represented by Equation (7). In Equations (9) and (10), B is an operator of a back-shift defined by $B_{xt} = x_t - x_{t-1}$, and a_t has an independent, equally distributed random sequence with a midnull covariation matrix and a positive-defined matrix $\sum a$.

VARMA can be expressed as Equation (11) for any two-time series z_{1t} and z_{2t},

$$\begin{bmatrix} z_{1t} \\ z_{2t} \end{bmatrix} - \begin{bmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{bmatrix} + \begin{bmatrix} z_{1t-1} \\ z_{2t-1} \end{bmatrix} = \begin{bmatrix} a_{1t} \\ a_{2t} \end{bmatrix} - \begin{bmatrix} \theta_{11} & \theta_{12} \\ \theta_{21} & \theta_{22} \end{bmatrix} * \begin{bmatrix} a_{1t-1} \\ a_{2t-1} \end{bmatrix}. \tag{11}$$

3.2.3. Recurrent Neural Networks/LSTM. Neural networks are a collection of algorithms designed for patterns to be identified by means of the interpretation of sensory data through interpretation, marking or grouping of raw data as shown in Figure 2.

The first phase in an advancing propagation is a neural network. The output is forecast by the network when an input is applied. The second stage is backpropagation, where the real learning takes place. Clanking fits well with conventional feedforward neural networks, but rather than evaluating input sequences, it is restricted to individual instances. Recurrent neural networks are based on neurons like feedforward neural networks, but, as shown in Figure 3, there are additional connections between layers.

The RNN architecture is depicted in Figure 3. There are one or more neurons in each square in the architecture, and each arrow is corresponding to a completely connected layer between the neurons on either side of the arrow. The left side shows the cyclically weighed RNN wheras the right side shows the over time, and the same network unrolls. Recurrent neural networks (RNN) are recurrent, as the functions used for every data input are the same, and the current input output is dependent on the last one. The output is copied into the recurrent network and returned as an input. It takes into account the present input and the output of the previous input. When using tanh as an activation method, RNN does not process very long sequences.

Long-term Short-Term Memory (LSTM) networks are an improved variant of recurring, deliberately designed to prevent the long-term issue of reliance. One of the inconveniences of RNN is also corrected here as a vanishing gradient problem. In view of time lags of uncertain length, LSTM is used to define, process, and forecast time series. In sequence prediction problems, it is able to learn order dependence [32].

LSTM architecture is shown in Figure 4.

The architecture, as shown in Figure 4, includes three cell gates—the forgotten gate, the input gate, and the output gate. These gates represent sigmoid functions which take values within the [0,1] range and transfer information to and from the cell. By descending the gradient, weight of the gates is balanced. Forget gate the sigmoid feature produces a 0 to 1 value which defines the amount of information to keep from the previous hidden state and current input. The sigmoid feature is used. In the cell state, C_{t-1} for each number is the predecessor state (h_{t-1}) and the input (i_t) and the output number between 0 and 1. The input gate selects the new data to be stored in the memory cell. A vector of new candidate values can be applied to the state in the tanh layer. The input gate is combined with the tanh layer to construct a state change. The tanh function weighs values between -1 and 1 to identify their significance. The old state is multiplied by f_i and the information we have forgotten before. Then, we add $i_t * C^t_{\text{new}}$. This is the current values of the candidate by how often each state value has been modified. The performance is finally dependent on the cell condition. The sigmoid function selects the values by [0,1], and the tanh function measures the values, so that their significance can be determined between -1 and 1 and multiplied by the sigmoid output.

Equations (12) to (17) of LSTM are as follows:

$$f_t = \sigma\left(W_f \left[h_{t-1}, X_t\right] + b_f\right), \tag{12}$$

$$i_t = \sigma\left(W_i \left[h_{t-1}, X_t\right] + b_i\right), \tag{13}$$

$$o_t = \sigma\left(Wo \left[h_{t-1}, X_t\right] + b_o\right), \tag{14}$$

$$C_t = f_i * C_{t-1} + i_t * C^t_{\text{new}}, \tag{15}$$

$$h_t = o_t * \tanh * C_t, \tag{16}$$

$$C^t_{\text{new}} = \tanh\left(W_c \left[h_{t-1}, X_t\right] + b_c\right). \tag{17}$$

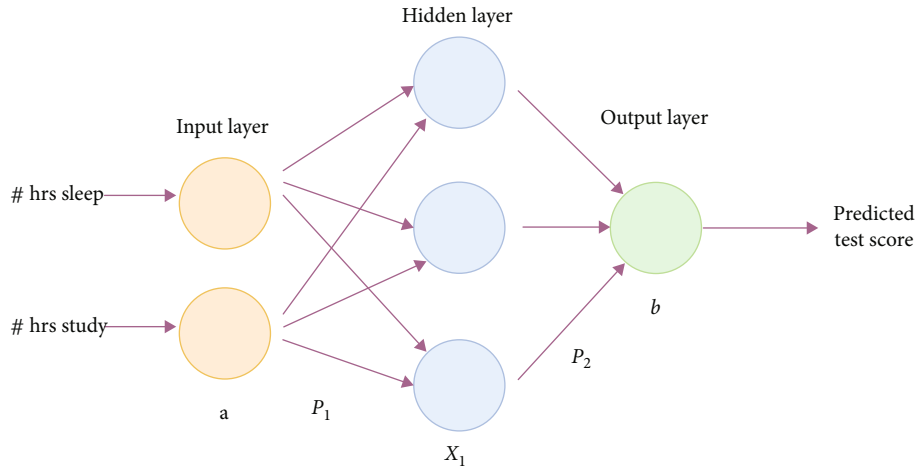

Figure 2: An example of a neural network.

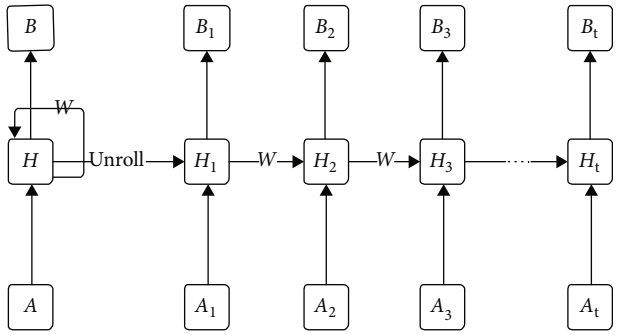

Figure 3: RNN architecture.

LSTM contains an actual memory that is not in the RNN architecture. When backed by depth, LSTM also retains a consistent error. It is anticipated that the LSTM network can yield the best results because it is suitable to time sequence problems. It is proposed that only if simpler conventional approaches fail can LSTM networks apply. LSTM is one of the most common methods for the prediction of time series statistics and machine learning.

3.2.4. Feedforward Neural Network. Any number of layers, units per layer, network inputs, and network outputs can be available for a neural network. Neural feedforward network (FNN) and the related training algorithm for backpropagation (BP). Neurons (i.e., processing units) are placed in the form of layers in a feedforward neural network.

In Figure 5, this is a 3-layer feedforward network, which consists of 4 units in the first (layer A), 3 units in second (layer B), known as hidden layers, and 3 units (layer C) known to be a layer known as output layer. There are four network inputs and one network output in the respective network.

A unit is equivalent to the inputs of network inputs. As shown in Figure 5, weight is altered in each network-input/unit-to-unit and unit-to-unit relation. Each unit has one additional input and one constant. The bias is the weight that modifies this additional data. It is called feeding as all data propagates from the network inputs to the network outputs along the connections [33].

$$Oc = h\text{Hidden}\left(\sum Ic, p\ Wc, p + bc\right), \tag{18}$$

$$h\text{Hidden}(x) = \frac{1}{(1 + e^{-\gamma x})}. \tag{19}$$

In Equations (18) and (19), O_c is the latest hidden layer unit c output, either P is number of units in the past hidden layer or in the number inputs of network, I_c, p is an input to c from either the hidden unit p of the previous layer or the network input p, Wc, p is the weight by which either unit p to unit c or p to unit c is altered, and b_c is the bias. H Hidden(x) is the unit's sigmoid enhancer. The y controls the slope of the feature constantly. Equation (20) gives the network input to the j processing unit

$$\text{net}_j = \sum_i x_i w_{ij} + \theta_j, \tag{20}$$

where $x_i{}'s$ is the output of the previous layer, w_{ij} is the weight, which specifies the position of the sigmoid, of the relation unit i to j, and θ_j is the weight of the bias.

A neural feed network primarily operates with established examples by forming the network. Based on the difference between o_p and y_p, a performance criterion function is defined. The sum of the squared error (SSE) function in Equation (21) is a widely used criterion function

$$F = \sum_p F_p = 1/2 \sum_p \sum_k \left(y_{pk} - o_{pk}\right)^2, \tag{21}$$

where p is the model index and k the output unit index. The output layer error is restored via the network, and the

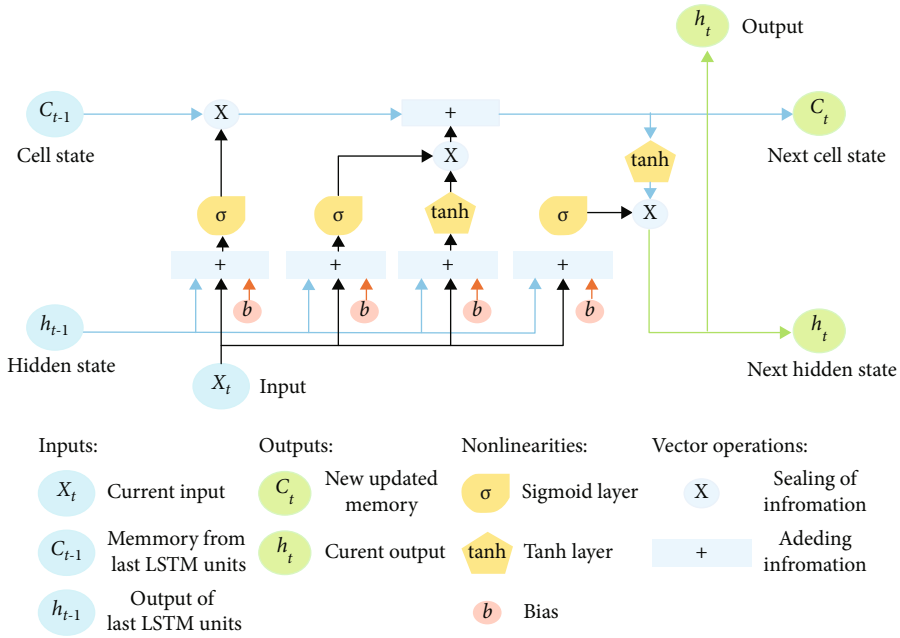

Figure 4: Architecture of LSTM.

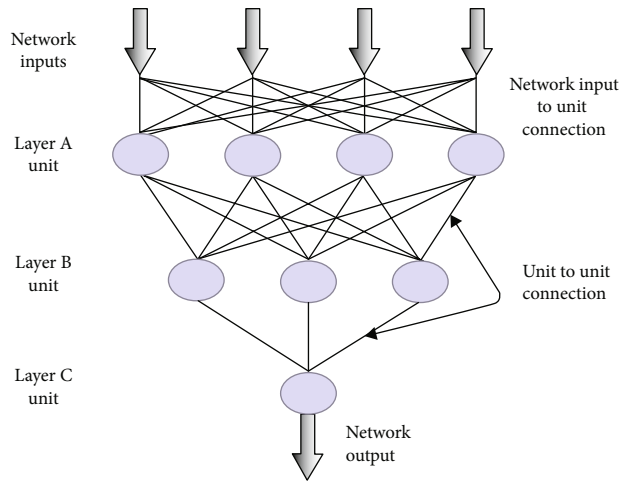

Figure 5: Architecture of feedforward network.

weights (w_{ij}) are updated based on their contribution to the error function shown in

$$\Delta w_{ij} = -\frac{\eta \delta F}{\delta w_{ij}}, \tag{22}$$

where η is called learning rate; the phase size of the weight is determined during the update.

4. Experimental Results Analysis and Discussion

In this section, we carry out experimental research to examine the efficiency in real traffic data sets of ARIMA and LSTM driven prediction models. Network service provider and traffic data from the UCI library were collected in 11 cities.

Table 1: Results of Dickey-Fuller test.

Test statistic	-1.025742*e*+01
p value	4.324537*e*-18
#lags used	3.100000*e*+01
Number of observations used	4.968000*e*+03
Critical value (1%)	-3.431667*e*+00
Critical value (5%)	-2.862122*e*+00
Critical value (10%)	-2.567080*e*+00
dtype	Float64

4.1. Experimental Setup. With varied durations of training sets, lengths of future predictions, feature sets used in learning and prediction, and so on, the suggested methodology is implemented on a single computer using Python. By utilizing Python to create a time series model, we can capture more of the data's complexity and incorporate all of the data pieces that are relevant. Adjustments to different measurements are also made possible which makes it potentially more accurate.

4.2. Experimental Results. In this section, we present the traffic forecasting performance results in following subsections, including time series collection at regular intervals, converting the time series into stationary time series, converting to supervised and then applying time series algorithm, evaluating the performance accuracy by using root mean square error (RMSE), *R*-squared score, mean absolute error (MAE), and mean absolute percent error (MAPE) performance indicators.

In linear forecasts of fixed time series variables, vector autoregression moving averages (VARMA) can be used. Several time series based on each other are modelled. The

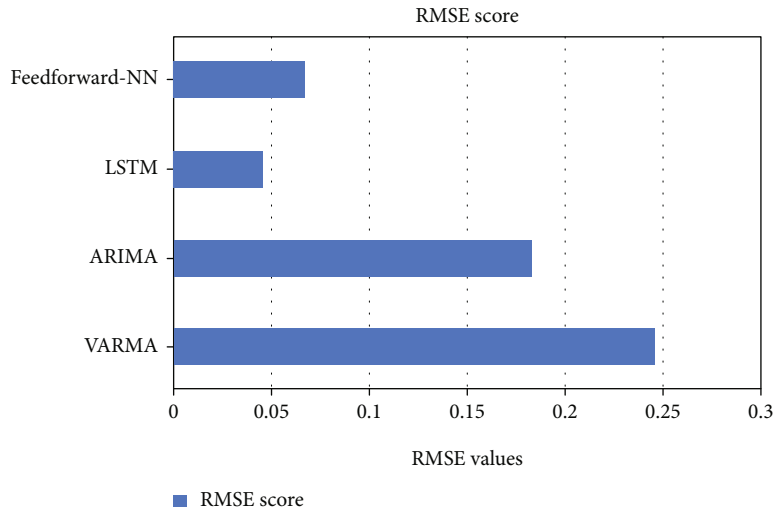

Figure 6: Comparing the RMSE scores of all the applied models.

interrelationships of the series are accounted for both. In the following step, each time series with an ARIMA model is modelled. Therefore, for multivariate stationary sequences, this model can be called the generalization of ARIMA. The vector autoregression moving average (VARMA) is a mixture between VAR and VMA and a generalized ARIMA model for stationary multivariate time sets. It is defined by parameters "x" and "y." It is similar to ARIMA, which is able to behave as an AR model by setting "y" as 0 and "x" as 0 as an MA model. Thus, VARMA is also able of interim like VAR model by setting "y" to 0 and as a VMA model by setting "x" to 0.

When we work with LSTM, the series data can be used as it is or can be converted to supervised learning data using "shift ()" function of the Pandas. Both the methods mentioned above are applied in our project. Hence, we have used two implementations of LSTM.

The DataFrame is created by inserting columns from the front and assigning NaN values to the inserted columns or inserting columns at the end and filling NaN values at the end. This process is accomplished using the shift () function. The shift () function will produce lag observation columns and prediction observation columns for the time series data set in a controlled learning format. Once all the comments are passed down, a new row is added to top. The new inserted row does not have information, so NaN represents "zero." The moved column with the aid of the shift feature is inserted next to our original sequence. New time series problem frameworks are generated using the shift () function on Pandas from the desired input and output sequence length. Pandas are a helpful tool in that it helps us to explore various framings of a time series problem using machine learning algorithms for model performance.

The data points obtained are called time series at constant intervals. Time series are analyzed to assess the long-term patterns in order to predict the future or do some other research. The two factors which distinguish a time series from a normal regression problem are as follows: (1) The linear model of regression is time-independent, and the time series depends on the time. (2) Time series have certain seasonal trends that are unique to a specific timing. By constructing a time series model using Python, we can understand the dynamics of the data more easily and include all the important data elements. Adjustments are often made to various dimensions which can make it more precise.

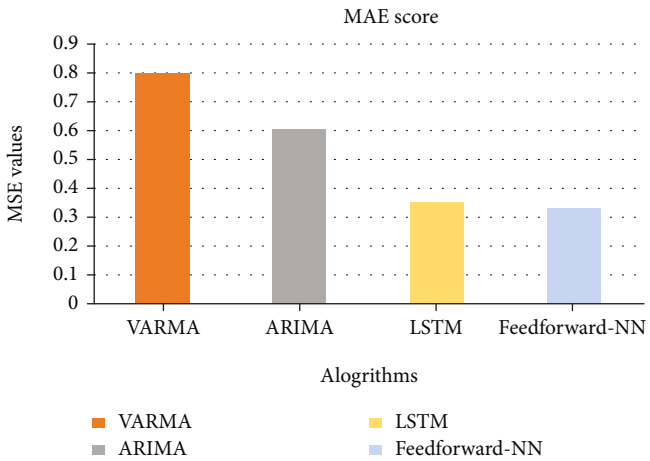

Figure 7: Comparing the MAE scores of all the applied models.

The time series must be converted into a time series (TS). It says that time series is stationary because of their statistical properties, such as average and variation according to their ability to stay continuous overtime. Most models of the time range presume that the time range is stationary [34]. If the time series plans to carry out a specific action over an era, it is likely to be the same in the future. Many studies in connection with stationary series often claim queer and simpler to deploy stationary time series in comparison with other nonstationary series. Stationary behavior is determined using certain fixed criteria's. However, practically the time series is assumed to be stationary if it has constant statistical

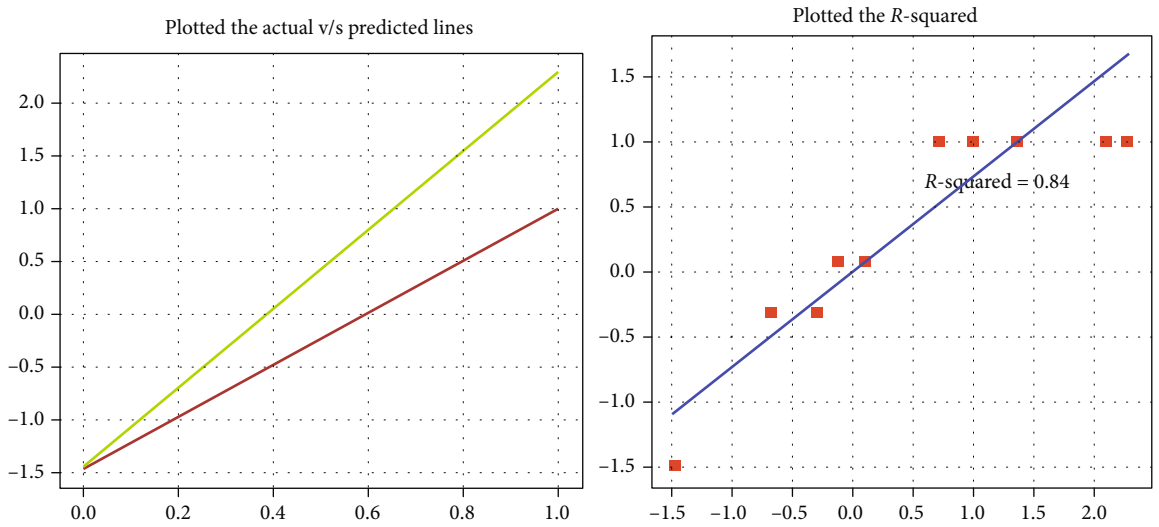

Figure 8: The predicted traffic values and R-squared values for ARIMA.

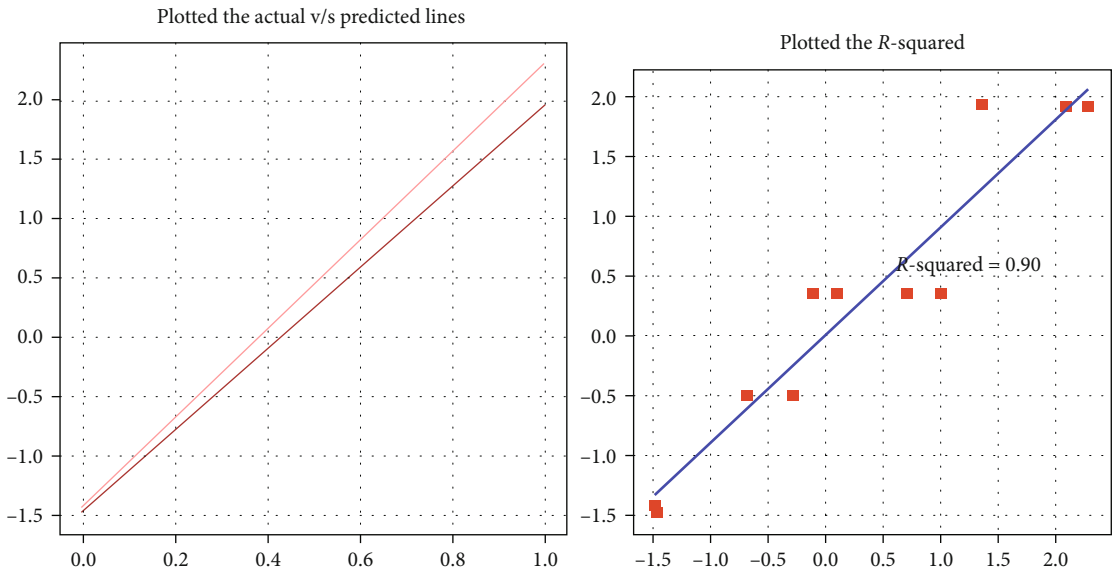

Figure 9: The predicted traffic values and R-squared values for LSTM.

properties that does not change overtime and that does not depend on time. In order to determine the statistical properties considered are as follows: (1) constant mean, (2) constant variance, and (3) self-covariance. Consequently, most time series used in the forecast models are stationary, or the time series must be stationary in order to suit a model. Visualization and the Augmented Dickey-Fuller (ADF) test are the two typical methods for the control of the time series stationarity.

As shown in Table 1, according to an interpretation of the results of the time series statistical analysis, there have been even fewer differences in the mean and standard deviation in magnitude in the time series listed. From the observation, it is shown that the test statistic is smaller than the 1% critical value. There are no missing values in the data observed as all values from starting are given weights. Therefore, this data will work even with fresh values.

4.3. Statistical Performance Evaluation. The accuracy of the traffic prediction models has been evaluated by mean absolute error (MAE), rooted mean squared error (RMSE), and R-squared values.

$$\begin{aligned} \text{MAE} &= 1/\xi \sum_{i=1}^{\xi} |y_i - Y_i|, \\ \text{RMSE} &= \sqrt{1/\xi \sum_{i=1}^{\xi} (y_i - Y_i)^2}, \end{aligned} \tag{23}$$

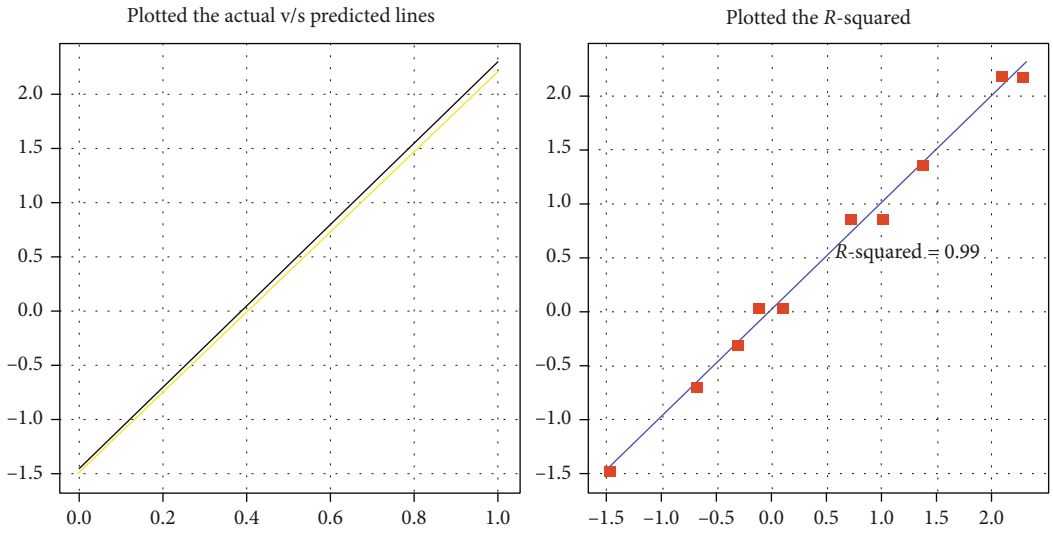

Figure 10: The predicted traffic values and R-squared values for feedforward-NN.

where yi and Yi denote the predicted value and the ground truth of region i for predicted time step and ξ is the total number of samples.

Figure 6 shows the RMSE values attained by various algorithms taken up for the research study for prediction of IoT traffic. The scores reveal that LSTM and feedforward-NN give better accuracy over ARIMA and VARMA.

The prediction results clearly show that the accuracy of prediction strongly depends on the training dataset, standardization of data, statistical evaluation, and the feature set used in the learning scheme. The results in Figure 7 indicate that the reframed LSTM and feedforward-NN are performing significantly better than other forecasting models such as VARMA and ARIMA.

It can be observed from Figures 8–10 that the highest accuracy in predicted and actual values is given by feedforward-NN, then at slight difference by reframed LSTM over the accuracy produced by ARIMA.

5. Conclusion and Future Scope

Prediction of IoT traffic in the recent years has attracted an insightful attention for enhancing resource and bandwidth utilization. This paper is focusing on the study of the problem of IoT traffic prediction by employing machine learning, deep learning, and statistical time series-based prediction methods including LSTM, ARIMA, VARMA, and feedforward neural networks. In the different traffic intervals and architecture parameters, a comparative study of prediction models on the basis of statistical learning has been performed. In the prediction of futuristic IoT-based traffic, the LSTM model and FNN model have shown the considerable accuracy over the conventional models such as VARMA and ARIMA. Our experiment results demonstrate that the LSTM and feedforward neural networks (FNN) exhibit more accurate predictions as compared to the other methods considered for comparative study on the basis of statistical parameters such as RMSE score, MAE score, and R-sqaured values. The future scope of the research is to design algorithms which can consider all dynamic parameters of IoT environment and can predict the upcoming traffic with more accuracy.

Data Availability

Data are available on request.

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] I. A. T. Hashem, V. Chang, N. B. Anuar et al., "The role of big data in smart city," *International Journal of Information Management*, vol. 36, no. 5, pp. 748–758, 2016.

[2] N. Naik, "Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP," in *2017 IEEE International Systems Engineering Symposium (ISSE)*, pp. 1–7, Vienna, Austria, 2017.

[3] G. K. Saini, H. Chouhan, S. Kori et al., "Recognition of human sentiment from image using machine learning," *Annals of the Romanian Society for Cell Biology*, vol. 25, no. 5, pp. 1802–1808, 2021.

[4] A. Dogra, A. Kaur, and M. Shabaz, "Data collection for classification in IOT and heart disease detection," *Annals of the Romanian Society for Cell Biology*, vol. 25, no. 4, pp. 2954–2964, 2021.

[5] N. Hung, S. Thomas, Y. Taku, and M. Takumi, "Generating IoT traffic: a case study on anomaly detection," in *Proceedings of the IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, pp. 1–6, Orlando, FL, USA, 2020.

[6] U. Cisco, *Cisco Annual Internet Report (2018–2023), White Paper*, 2020.

[7] 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, 11 pages, 2019.

[8] J. Mejia, A. Ochoa-Zezzati, and O. Cruz-Mejía, "Traffic forecasting on mobile networks using 3D convolutional layers," *Mobile Network and Applications*, vol. 25, no. 6, pp. 2134–2140, 2020.

[9] A. Essien, I. Petrounias, P. Sampaio, and S. Sandra, "A deep-learning model for urban traffic flow prediction with traffic events mined from Twitter," World Wide Web, 2021.

[10] A. Volkov, A. R. Abdellah, A. Muthanna, M. Makolkina, A. Paramonov, and A. Koucheryavy, "IoT Traffic Prediction with Neural Networks Learning Based on SDN Infrastructure," in *Distributed Computer and Communication Networks. DCCN 2020*, V. M. Vishnevskiy, K. E. Samouylov, and D. V. Kozyrev, Eds., vol. 12563, pp. 64–76, Springer, 2020.

[11] G. Meena, D. Sharma, and M. Mahrishi, "Traffic prediction for intelligent transportation system using machine learning," in *2020 3rd International Conference on Emerging Technologies in Computer Engineering: Machine Learning and Internet of Things (ICETCE)*, pp. 145–148, Jaipur, India, 2020.

[12] 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.

[13] Y. Hua, Z. Zhao, Z. Liu, X. Chen, R. Li, and H. Zhang, "Traffic prediction based on random connectivity in deep learning with long short-term memory," in *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pp. 1–6, Chicago, IL, USA, 2018.

[14] Y. Wang, Y. Guo, Z. Wei, Y. Huang, and X. Liu, "Traffic flow prediction based on deep neural networks," in *2019 International Conference on Data Mining Workshops (ICDMW)*, pp. 210–215, Beijing, China, 2019.

[15] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: a survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.

[16] A. R. Abdellah, O. A. K. Mahmood, A. Paramonov, and A. Koucheryavy, "IoT traffic prediction using multi-step ahead prediction with neural network," in *2019 11th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pp. 1–4, Dublin, Ireland, 2019.

[17] A. Essien and C. Giannetti, "A deep learning framework for univariate time series prediction using convolutional LSTM stacked autoencoders," in *2019 IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA)*, pp. 1–6, Sofia, Bulgaria, 2019.

[18] S. J. Kamble and M. R. Kounte, "Machine learning approach on traffic congestion monitoring system in Internet of Vehicles," *Procedia Computer Science*, vol. 171, pp. 2235–2241, 2020.

[19] F. Tang, Z. M. Fadlullah, B. Mao, and N. Kato, "An intelligent traffic load prediction-based adaptive channel assignment algorithm in SDN-IoT: a deep learning approach," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 5141–5154, 2018.

[20] D.-B. Nguyen, C.-R. Dow, and S.-F. Hwang, "An efficient traffic congestion monitoring system on Internet of Vehicles," *Wireless Communications and Mobile Computing*, vol. 2018, 17 pages, 2018.

[21] N. G. Polson and V. O. Sokolov, "Deep learning for short-term traffic flow prediction," *Transportation Research Part C: Emerging Technologies*, vol. 79, pp. 1–17, 2017.

[22] C. Huang, C. Chiang, and Q. Li, "A study of deep learning networks on mobile traffic forecasting," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–6, Montreal, QC, Canada, 2017.

[23] R. Li, Z. Zhao, J. Zheng, C. Mei, Y. Cai, and H. Zhang, "The learning and prediction of application-level traffic data in cellular networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 3899–3912, 2017.

[24] X. Luo, R. Shen, J. Hu, J. Deng, L. Hu, and Q. Guan, "A deep convolution neural network model for vehicle recognition and face recognition," *Procedia Computer Science*, vol. 107, pp. 715–720, 2017.

[25] F. Xu, Y. Lin, J. Huang et al., "Big data driven mobile traffic understanding and forecasting: a time series approach," *IEEE Transactions on Services Computing*, vol. 9, no. 5, pp. 796–805, 2016.

[26] G. Bontempi, S. B. Taieb, and Y. A. L. Borgne, "Machine Learning Strategies for Time Series Forecasting," in *European business intelligence summer school*, M. A. Aufaure and E. Zimányi, Eds., vol. 138, pp. 62–77, Springer, 2013.

[27] A. Géron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow*, O'Reilly Media, Inc., 2017.

[28] K. Li, C. Zhai, and J. Xu, "Short-term traffic flow prediction using a methodology based on ARIMA and RBF-ANN," in *2017 Chinese Automation Congress (CAC)*, pp. 2804–2807, Jinan, China, 2017.

[29] C. Giannetti, A. Essien, and Y. O. Pang, *A Novel Deep Learning Approach for Event Detection in Smart Manufacturing*, CIE49 proceedings, 2019.

[30] L. Helmut, "Chapter 6 Forecasting with VARMA Models," *Handbook of Economic Forecasting*, vol. 1, pp. 287–325, 2006.

[31] P. Aboagye-Sarfo, Q. Mai, F. M. Sanfilippo, D. B. Preen, L. M. Stewart, and D. M. Fatovich, "A comparison of multivariate and univariate time series approaches to modelling and forecasting emergency department demand in Western Australia," *Journal of Biomedical Informatics*, vol. 57, pp. 62–73, 2015.

[32] F. I. Facultit, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," in *A Field Guide to Dynamical Recurrent Neural Network*, IEEE Press, 2001.

[33] Z. Tang and A. F. Paul, "Feedforward neural nets as models for time series forecasting," *INFORMS Journal on Computing*, vol. 5, no. 4, pp. 374–385, 1993.

[34] M. Abbasi, A. Shahraki, and A. Taherkordi, "Deep learning for network traffic monitoring and analysis (NTMA): a survey," *Computer Communications*, vol. 170, pp. 19–41, 2021.