Machine Learning (2024) 113:2277–2300
https://doi.org/10.1007/s10994-023-06471-1

Forecasting of mobile network traffic and spatio–temporal analysis using modLSTM

Vidyadhar J. Aski[1,2] · Rugved Sanjay Chavan[1,4] · Vijaypal Singh Dhaka[1] · Geeta Rani[1] · Ester Zumpano[3,5] · Eugenio Vocaturo[3,5]

Received: 3 January 2023 / Revised: 8 August 2023 / Accepted: 17 October 2023 /
Published online: 30 November 2023

Abstract
This paper introduces an innovative system and prediction model for forecasting network traffic in specific geographical locations using historical data. As Internet service providers increasingly rely on data analytics for decision-making, optimized network forecasting faces challenges such as data cleaning and preprocessing. Our approach utilizes an Artificial Recurrent Neural Network-based Modified Long Short-Term Memory model to provide continuous and precise predictions of network traffic. Notably, the proposed model outperforms conventional LSTM models, achieving a 61.9% reduction in Mean Absolute Percent Error. Our approach also integrates an interpolation technique to address the zero-component error. This further enhances the effectiveness and reliability of the model. The model promises to enhance resource utilization and lighten the load on traffic resource provisioning entities, promoting more efficient mobile network traffic management. The low training time of 3.26 min and prediction time of 0.14 s pave the way for real-time implementation of the model for network traffic forecasting and management. The comparative analysis with state-of-the-art models proves the supremacy of the proposed model.

Keywords Network traffic · Prediction · Cellular · Artificial recurrent neural network · Grid

1 Introduction

In today's highly interconnected world, mobile communication has become an integral part of our lives. Ericsson (2021) reported a surge of approximately 21% in mobile data traffic per smartphone per month. This rise is anticipated to triple annually, pushing the envelope of technological advancements and catalyzing the development of more robust, data-driven solutions that can efficiently navigate through this exabyte data era. These solutions are critical not only to satisfy the increasing consumer demand but also to create sustainable

Editors: Dino Ienco, Roberto Interdonato, Pascal Poncelet.

Vidyadhar J. Aski, Rugved Sanjay Chavan, Geeta Rani, Ester Zumpano and Eugenio Vocaturo have contributed equally to this work.

Extended author information available on the last page of the article models that reduce energy consumption and make mobile networks more reliable and effective.

One significant component of this technological ecosystem is network Base Stations (nBS). These stations are at the heart of mobile communication, carrying the responsibility of managing data traffic. With the influx of data, the emphasis on predictive maintenance of these stations has become more pronounced, paving the way for service providers to prepare for future challenges.

Our communication behavior generates a myriad of traffic data captured in Call Detail Records (CDR), encompassing diverse parameters such as square ID, time interval, country code, and various activity data. Also, this complex data structure, a fusion of spatiotemporal data and associated timestamps, presents a unique challenge due to its inherent nonlinearity and complexity (Li et al., 2020). The constant flux in the spatiotemporal features, primarily due to our dynamic lifestyles and varied usage patterns, adds another layer of intricacy.

Capturing these swift changes in data becomes crucial to improve the accuracy of network traffic forecasting. It can serve as a linchpin for effective resource management and load balancing in the mobile network. However, predicting mobile traffic, particularly at a granular level, is not a straightforward task. It needs to address the following challenges.

1. The network demand spectrum of individual cellular towers is broad and unpredictable. This is due to the ever-evolving behavior of users, who engage with the network in different settings such as offices while traveling, and in diverse internet applications.
2. The escalated user mobility introduces random fluctuations in the transmitting and receiving capabilities of cell towers, adding another level of uncertainty.
3. The strategic placement of cellular towers based on population and locality affects the network traffic. The change in network load occurs due to social activities, holidays, land acquisitions etc. These dynamic factors further create complex dependencies on spatiotemporal features.

Several studies have explored these challenges, For example, Barlacchi (2015) meticulously investigated the characteristics of an urban wireless traffic dataset with massive internet data usage. They leveraged graphical representations to analyze traffic intensity across 10,000 square grids of Milan city, calculated the Probability Density Function (PDF) of the traffic, and created time series plots for the identified areas.

Another study by Zhang et al. (2019) includes the Spatial–Temporal Cross-domain Neural Network (STCNet), for segmenting a city into grids. But, the model reports a minimum mean absolute error of 15.85. This leaves scope for improvement. Similarly, other researchers, like Wang et al. (2019) and Zhao et al. (2019), proposed models that included temporal dependency-based learning capabilities or used Machine Learning (ML) methods, but these models failed to minimize the prediction time or reduce the forecasting time.

Despite the valuable insights provided by these studies, a comprehensive yet universally applicable method that can continuously forecast future traffic for individual cellular towers in real-time remains elusive. There is a gap in addressing the spatial dependency of individual cell towers. Also, the models make predictions for specific geographical locations only. The research in this manuscript aims to bridge the above-mentioned gaps by introducing a modified version of the Long Short-Term Memory (LSTM) model to accurately predict network traffic. It offers an integration of

Recurrent Neural Network (RNN) and modLSTM to mitigate the significant drawback of error propagation that occurs in LSTM models. The integrated RNN and modLSTM aim to predict network traffic accurately and continuously for a specific grid from a particular location. The integration of actual data in the forecast aims to enhance the precision of predictions, creating a model adaptable to the continuously changing behavior of users and the dynamically varying traffic patterns. This makes strides in the field of mobile network traffic prediction, creating a foundation for more reliable and efficient mobile network services. This research aims to provide service providers with a robust tool to navigate the challenges of the growing data era, ultimately enhancing the user experience and the sustainability of the mobile communication landscape.

The primary contributions of this research are as follows:

1. Developing a customized LSTM model, 'modLSTM', to predict mobile network traffic based on historical and actual data generated in a specific region.
2. Minimizing the error in forecasting mobile network traffic in real-world scenarios, enhancing the reliability and accuracy of the predictions.
3. Analyzing the spatial dependency of individual cell towers, providing valuable insights into the spatial dynamics affecting mobile network traffic.
4. To introduce a dynamic and universal approach for the characterization and prediction of mobile network traffic, aiming to develop a solution adaptable across different geographical locations.

The remaining paper is organized as follows. Section 2 provides a brief overview of materials and methods, including dataset preparation, data preprocessing, and the architecture of the proposed model. Section 3 presents the results of the proposed ModLSTM model. Finally, Sect. 4 presents the conclusions of the research and suggests future research directions.

2 Materials and methods

This section elaborates on the preparation of datasets and the approach proposed for network traffic characterization and forecasting. The overview of the proposed methodology is shown in Fig. 1.

2.1 Data acquisition and preparation

The experiment utilizes cellular traffic data sourced from reference Harvard (2013). This dataset comprises cellular traffic information for 9999 square grids. These grids are two-dimensional representations of distinct geographical areas as shown in Fig. 2. Each grid is a collection of cells where it is assigned a unique identifier i.e. square ID represents the internet traffic activity data of an area. For example, the square grids visualized in Fig. 2 depict a map of Milan city with an overlay of 9999 square IDs. Each square id corresponds to a specific geographical area of Milan City. It refers to the internet traffic data associated with that geographical area over time. Thus, making it an important parameter in this research.

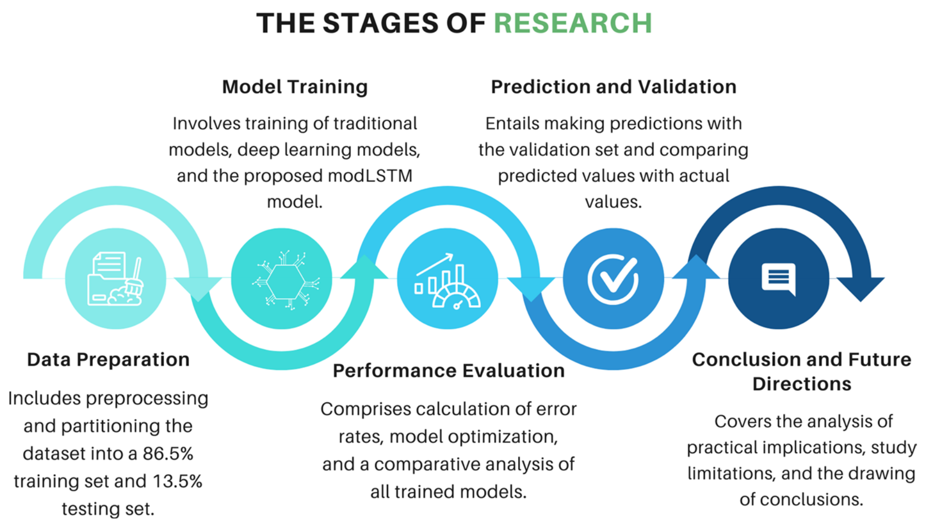

THE STAGES OF RESEARCH
Model Training
Involves training of traditional models, deep learning models, and the proposed modLSTM model.
Prediction and Validation
Entails making predictions with the validation set and comparing predicted values with actual values.
Data Preparation
Includes preprocessing and partitioning the dataset into a 86.5% training set and 13.5% testing set.
Performance Evaluation
Comprises calculation of error rates, model optimization, and a comparative analysis of all trained models.
Conclusion and Future Directions
Covers the analysis of practical implications, study limitations, and the drawing of conclusions.

Fig. 1 Overview of the proposed methodology

These 9999 square grids are divided into smaller units to facilitate data collection and analysis. The dataset includes data on the number of Call Detail Records (CDRs) generated within each square grid over 10-min intervals for a 2 month period from November 2013 to January 2014.

Each file in the dataset is an eight-column text table, with columns representing square ID, time interval, country code, incoming and outgoing SMS activity, incoming and outgoing call activity, and internet traffic activity. The term ‘Internet traffic activity’ in this context refers to the amount of data moving across the Internet. It is the data created,

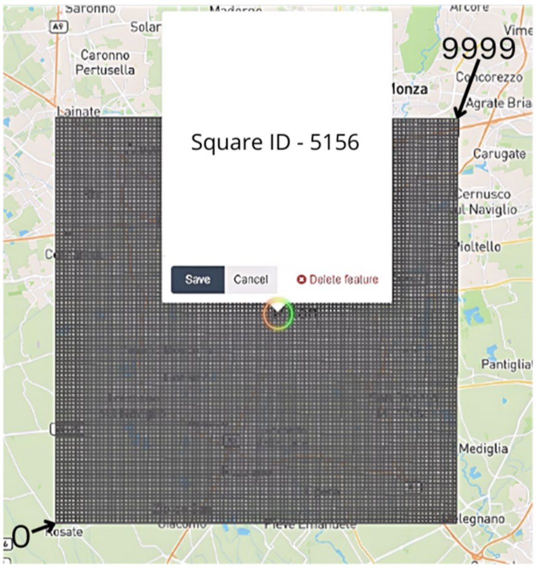


Fig. 2 Geographical representation of Milan city with an overlay of 9999 square IDs

transmitted, and received within a specific time period for a geographical area. A person generates data by using the internet for browsing websites, sending or receiving emails, streaming content, and participating in online chats. This data is divided into small packets sent and received over the Internet. For example, the number of Call Detail Records (CDRs) generated within a given geographical area for a specified duration is a part of the Internet traffic activity of that area. The researchers and network administrators can understand usage patterns, network load, the popularity of certain types of content or services, and peak usage hours by monitoring and analyzing Internet traffic activity.

Despite the richness of this dataset, it presents certain issues such as missing entries and irregular spacing between columns. These irregularities necessitate a meticulous preprocessing phase to ensure the data's usability and integrity.

Initial preprocessing efforts focused on standardizing the column layout. Rows with the correct count of eight columns were retained by using whitespace as a delimiter. Thus ensuring consistent spacing and column integrity. This process was followed for data reduction, in which only the square ID, time interval, and internet traffic activity columns were extracted for subsequent analysis. These columns were consolidated across all 62 files into a single dataset, resulting in a unified table of approximately 65 million rows.

The next phase of preprocessing involved analyzing the condensed dataset to identify areas of high total traffic over the 2 months span. This requires creating a data frame containing the square ID, sum total, and average sum total for each day's total internet traffic for a given grid. It is the sum of all its internet traffic activity values calculated as per Eq. (1):

$$\text{Slots per day} = 24 \times 60/10 = 144 \tag{1}$$

The sum total and average of internet activity for each grid over the 62 days period was calculated. This distribution of average internet activity is visualized in Fig. 3, where a high activity concentration can be observed in the city's center for square IDs between 5000 and 5200.

Following this, we turned our attention toward exploring the probabilistic aspects of our dataset. We began by calculating a Relative Frequency Histogram (RFH) of network traffic across the entire city grid as presented in Eq. (2):

$$T_{\text{avg}}(G_j) = \frac{1}{n \times m} \sum_{i=1}^{n} T_{d_i} \tag{2}$$

In Eq. (2), $T_{\text{avg}}(G_j)$ is the average internet activity for a specific grid j, which ranges from 1 to 9999. T_{d_i} represents the total internet activity of one grid for a day, n signifies the number of days (62 in this case), and m is the number of time intervals in a day (144 intervals of 10 mins each).

To understand the distribution pattern of the data, we also examined the smoothed histogram of frequencies over the 10,000 grid areas for the 62 days period as depicted in Fig. 4. This representation was calculated according to Eq. (3), as detailed in Ref. Barlacchi (2015):

$$P(a \leq x \leq b) = \int_a^b f(x)dx \tag{3}$$

Here, $P(a \leq x \leq b)$ is the probability of x. Its' value lies between the values of a and b. $f(x)$ is the frequency function.

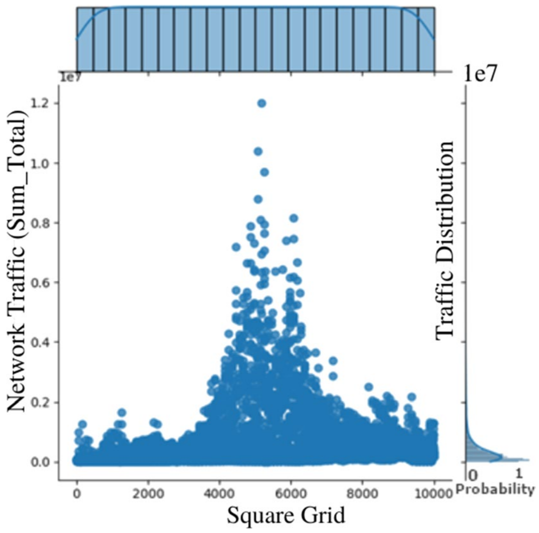


Fig. 3 Distribution of average internet activity for 2 months

Figure 4 shows the normalized histogram of the dataset used in this research. Its' x-axis shows internet traffic activity and the y-axis represents the relative frequency. The relative frequency is calculated by dividing the frequency of each internet traffic activity level by the total number of observations. Figure 4 also demonstrates an observed frequency distribution in different levels of internet traffic activity. It is evident from the figure that, during the 2-month period, only one location manifests the highest network traffic volumes. Thus, the probability of the highest traffic activity is nearly negligible for a randomly selected grid. This probability can be increased by considering a subset of the grids rather than a random grid. But, considering a subset of random grids may provide distorted data usage due to a lack of capturing data usage patterns across the entire city grid. This may lead to incorrect data capturing for a city and hence inaccurate predictions.

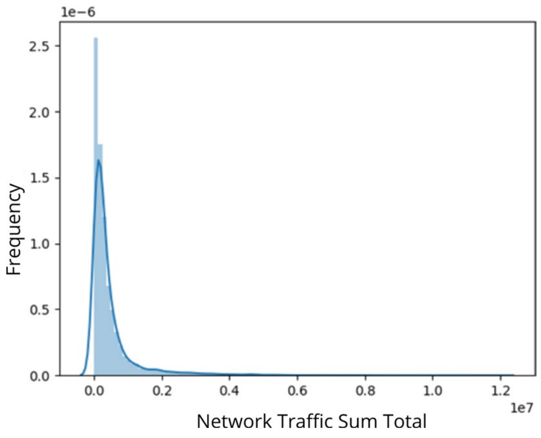


Fig. 4 Smoothed histogram of frequencies for 10,000 samples

2.2 Model selection

Selecting Long Short-Term Memory (LSTM) networks as the foundation of the proposed model 'modLSTM' was driven by the following characteristics of LSTM and requirements of time-series forecasting task.

1. Handling long-term dependencies: The traditional neural networks and other ML models such as K-Nearest Neighbor, and decision trees, have limited ability to effectively predict future values based on long-term dependencies in the data. On the other hand, LSTM networks are specifically designed to remember information for long periods of time. This makes them highly suitable for interpreting sequential data with potentially important temporal dependencies as observed in the cellular traffic dataset used in this research.
2. Memory cell architecture: The unique structure of the LSTM, especially the memory cell with its' forget input and output gates, allows the model to control the flow of information across time steps, including the ability to forget irrelevant previous data and remember useful historical features. This ability to regulate information flow is crucial in effectively learning from time-series data.
3. Mitigating the vanishing gradient problem: The vanishing gradient problem leads to poor learning and difficulty in adjusting the parameters associated with earlier layers of Recurrent Neural Networks (RNNs). LSTMs alleviate this problem with their gating mechanisms and become efficient in learning patterns from the data.
4. Proven performance: LSTM networks have demonstrated strong performance on a wide range of time-series forecasting problems such as stock price prediction, natural language processing, and network traffic forecasting. This proven potential of LSTM in these related domains also motivated us to employ LSTM for this research.

2.3 Forecasting models: LSTM versus modLSTM

The conventional LSTM employs an unsupervised learning approach. It forecasts the future traffic based on the history of training data and predicted values as illustrated in Fig. 5. The standard LSTM model could indeed use the actual values instead of the predicted values for training the model. However, in practice, when LSTM is applied for time series forecasting, then the actual value is not available. In such a scenario, we rely on the predicted values from the model for future predictions. The error in the predicted values is carried to the next iteration which may lead to wrong predictions. Also, the standard LSTM operates on a fixed training dataset without considering new data points once the model is trained. In contrast, modLSTM intentionally incorporates actual data into the model even after initial predictions. It also reduces the error that accumulates over time in a standard LSTM when predictions are based solely on previously predicted erroneous values. The modLSTM is designed specifically to work in the real world to incorporate actual data points back into the model as it becomes available. This gives a significant improvement in prediction accuracy.

It is suitable for predicting network traffic (Cisar & Cisar, 2012; Wang et al., 2019). Hsowever, it demonstrates the marginal variation of actual traffic from the predicted results for successive days. The following data is anticipated based on the previous data and the predicted values. This continues, and the inaccuracy becomes more pronounced.

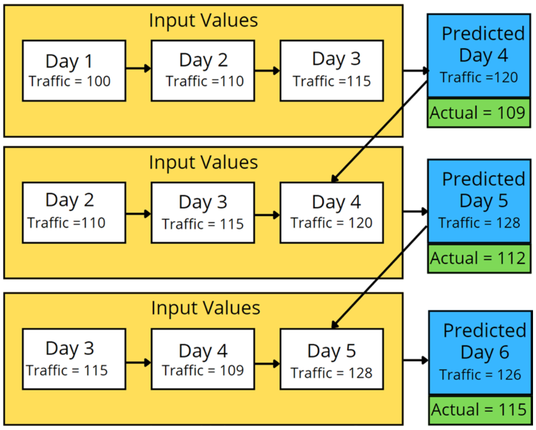


Fig. 5 Forecasting with conventional LSTM architecture

The authors of this manuscript identified the significant drawbacks of LSTM in attributing the error to the next prediction. The error is equal to the difference in the value of the training data and its' corresponding predicted values. It has been observed that conventional LSTM reduces prediction accuracy. Therefore, we modified the conventional LSTM model in terms of input attributes and proposed the novel modLSTM model. It is a special type of Recurrent Neural Network (RNN) with the capacity to remember short time-series data for an extended period. Every neuron in an RNN sends feedback to every other neuron in the network, whereas every neuron in a modLSTM is a memory cell. Its current neuron stores previous data and is used to predict the subsequent state.

The primary step in LSTM, as shown in Fig. 6, is to decide what information will be discarded from the current cell state. This decision is made using the forget gate. The forward propagation of information in modLSTM is similar to an RNN, with input data being a time series of length T. The algorithm employs the parameters specified in Eqs. (4) through (9). These parameters have been adapted from the work proposed by Lu and Yang (2018) and Li et al. (2020). Here, σ is a sigmoid function. Given a series of x and an initial state h_0, the prediction model performs iterative computations for recurrent states starting from h_{t-1}. Here, x_t denotes the function coefficient at a given time interval t. The parameters W_i, W_f, W_o, W_c are weight matrices, while b_f, b_i, b_o, b_c are bias terms.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{4}$$

Equation 4 computes the forget gate values f_t, determining how much of the past state C_{t-1} should be forgotten.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{5}$$

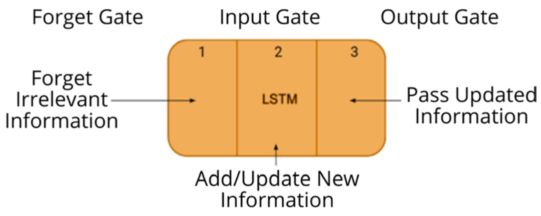


Fig. 6 Structure of LSTM

Equation 5 computes the input gate values i_t, deciding how much of the newly computed state for time t, C_t, should be stored in the cell state.

$$\tilde{C}t = \tanh(W_c \cdot [ht - 1, x_t] + b_c) \tag{6}$$

Equation 6 computes the candidate cell state $\tilde{C}_t$ for the current time step t.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{7}$$

Equation 7 updates the cell state C_t, which is a combination of the past cell state and the candidate cell state, moderated by the forget and input gate values respectively.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{8}$$

Equation 8 computes the output gate values o_t, deciding how much of the cell state C_t should be output at this time step.

$$h_t = o_t \cdot \tanh(C_t) \tag{9}$$

Equation 9 calculates the output hidden state h_t for this time step, which will be used in the next time step. The hidden state is the cell state passed through the tanh activation function and moderated by the output gate.

The modified Long Short-Term Memory (modLSTM) aims to resolve the error margin commonly observed in conventional LSTM by incorporating a different approach to future predictions. Rather than relying solely on predicted results, modLSTM makes use of both the training dataset's history and the actual traffic data. This method effectively eliminates the moving average errors in predicting future values.

As depicted in Fig. 7, the modLSTM's predictions consider both historical and actual data. For instance, the predicted traffic volume for day 4 was based on the data from days 1, 2, and 3. In this case, the expected traffic volume for day 4 was 120, whereas the observed volume was 109. This process ensures that subsequent predictions, like for day 5, also consider both the training data and the actual data, thus minimizing the prediction error.

The proposed cellular traffic forecasting system is bifurcated into two key parts: data characterization and traffic prediction.

The data characterization involves the extraction of required parameters in a suitable format from the large dataset. This process includes pre-processing the data, data cleaning, data transformation, and data loading. The processed data is then forwarded to the prediction part.

The traffic prediction employs modLSTM to make predictions and subsequently optimizes the predicted results through interpolation. This optimization process includes reducing the forecast error and achieving more accurate and reliable predictions.

The architecture of the proposed cellular traffic forecasting system is illustrated in Fig. 8.

The algorithm used by modLSTM is illustrated below:

2.4 Experimental setup

The data cleaning (Kotsiantis et al., 2006), processing (Munz & Carle, 2007) (Rani et al., 2022), and prediction tasks (Montgomery et al., 2015) were executed on a workstation equipped with an Intel core i9-10850K processor running at 5.20 GHz, 64 GB of system

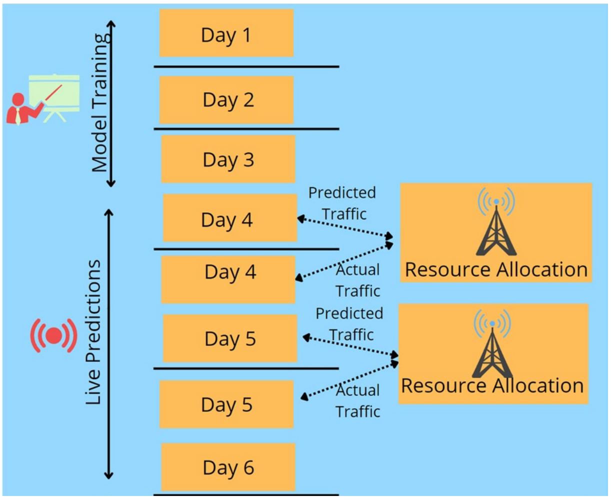
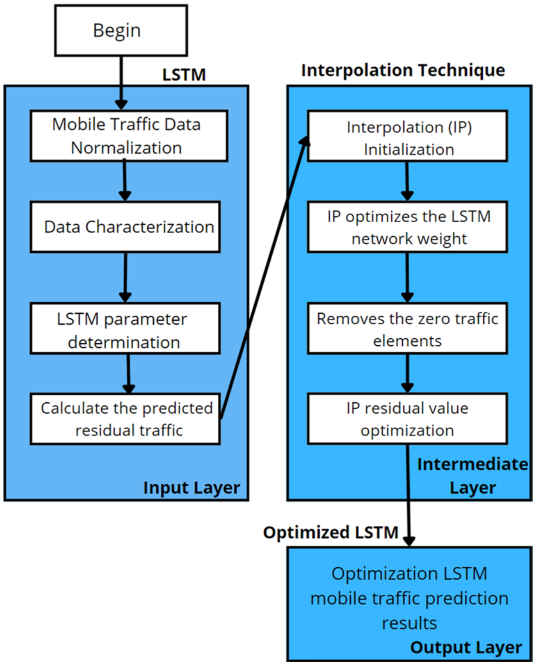


Fig. 7 Forecasting with modLSTM

memory (RAM), and a Gigabyte NVIDIA GeForce RTX 3080 graphics card with 10GB of dedicated GPU memory (NVIDIA, 2020). In terms of storage, the system was provided with a capacity of 3 TB (Patel, 2019). The machine was also fitted with a Realtek RTL8125 2.5 GbE network interface card (NIC), in addition to an Intel Tiger Lake PCH CNVi WiFi

Fig. 8 Flowchart of the proposed cellular traffic forecasting architecture


Begin
LSTM
Mobile Traffic Data Normalization
Data Characterization
LSTM parameter determination
Calculate the predicted residual traffic
Input Layer
Interpolation Technique
Interpolation (IP) Initialization
IP optimizes the LSTM network weight
Removes the zero traffic elements
IP residual value optimization
Intermediate Layer
Optimized LSTM
Optimization LSTM mobile traffic prediction results
Output Layer

Algorithm 1 modLSTM

```
Require: Training dataset $X_t$, actual data $y_t$
Ensure: Traffic prediction
1: Initialize LSTM parameters using equations (4)-(9)
2: for each day $t$ in dataset do
3:     Extract historical data $X_t$
4:     Extract actual data $y_t$
5:     Feed $X_t$ and $y_t$ into the LSTM network
6:     Compute the forget gate $f_t$ using Equation (4)
7:     Compute the input gate $i_t$ and cell state candidate $\widetilde{C}_t$ using Equations (5) and (6)
8:     Update the cell state $C_t$ using Equation (7)
9:     Compute the output gate $o_t$ and hidden state $h_t$ using Equations (8) and (9)
10:    Compute prediction for the next day $\hat{y}_{t+1} = h_t$
11:    Update LSTM state with the actual data $y_t$
12: end for
13: return Predicted traffic
```

NIC, supporting upload and download network speeds of 372.27 and 910.38 Mbps respectively (Intel, 2021).

The operating system running on this workstation was Ubuntu 20.04.3 LTS, a 64-bit linux distribution (Ltd., 2020). All algorithms were implemented using Python 3.8.10 (Foundation, 2020). PyCharm2021.3 (version 213.6461.77) served as the integrated development environment (IDE) for this project (JetBrains, 2021). For data analysis, the pandas' library was employed (McKinney, 2010), while the Matplotlib library was used for data visualization (Hunter, 2007). NumPy was utilized for handling and processing multidimensional arrays (Harris et al., 2020).

2.5 Experiments

The modLSTM model was implemented in the python programming language using the TensorFlow library.

The model is composed of three LSTM layers followed by a dense output layer, as summarized in Table 1. Each LSTM layer has a specific role and number of nodes: the first and second LSTM layers are responsible for extracting temporal dependencies from the input sequence, while the third LSTM layer is employed for integrating the learned features. The final dense layer uses these integrated features for the final traffic prediction.

The hyperparameters of the modLSTM model include the 'timestamp' value and the number of nodes in each LSTM layer. The 'timestamp' represents the number of time stamps the LSTM looks back, in order to make a prediction. Its' value was set to 100. The value was determined by a series of trial experiments. The goal of these trials was to maintain the trade-off between providing sufficient historical information ; and preventing the model from becoming overly complex and time-consuming to train.

Table 1 Summary of LSTM model

Layer (type)	Output shape	Parameters
lstm (LSTM)	(None, 100, 50)	10,400
lstm_1 (LSTM)	(None, 100, 150)	120,600
lstm_2 (LSTM)	(None, 250)	401,000
dense (Dense)	(None, 1)	251

The number of nodes in each LSTM layer was also selected through a process of trial and error experiments. Each LSTM layer was initially given a small number of nodes, and the number was gradually increased until no significant improvement in performance was observed. The optimal performance of the model was observed when the modLSTM architecture comprised the first LSTM layer with 50 nodes, the second layer with 150 nodes, and the third layer with 250 nodes. In addition to these structural hyperparameters, the learning process of the modLSTM model was regulated by a number of additional hyperparameters. The model was trained for 50 epochs. After 50 epochs, the model achieved stability and further training could not yield substantial improvements. Further . The learning rate of the Adam optimizer was set to the default value of 0.001. This choice was based on prior research and the set of trial experiments. At this learning rate, the model maintains a balance between rapid convergence and stability.

During the training process, the model's weights were updated after each batch of 32 samples. This batch size was chosen as a compromise between computational efficiency and the quality of the gradient estimates. Furthermore, a dropout rate of 20% was employed after each LSTM layer to minimize the problem of overfitting. In each training epoch, 20% of the randomly selected nodes of modLSTM were deactivated.

Finally, to evaluate the performance of the modLSTM model, the Mean Absolute Percentage Error (MAPE) was used. This metric was chosen because it provides a scale-independent measure of prediction accuracy, allowing for a fair comparison across different traffic volumes.

2.6 Time series analysis

The time series analysis includes an analysis of the time series plots for network traffic during the first 2 weeks in three areas: the area with square ID 5161 with the highest total traffic during the 2 months, the area with square ID 4556 with average total traffic, and the area with square ID 4159 with less total traffic. This analysis discusses the similarities or differences observed in the temporal dynamics of each area and speculates on their causes. Figure 9 shows the superposed time series plot of the first 2 weeks' traffic in all three areas with the highest traffic for 62 days. In all the graphs shown in Figs. 9, 10, 11, 12, 13, 14, 15, 16, 17, and 18, the X-axis, and Y-axis represent the time slots, and the cellular traffic values respectively. In one hour, there are six-time slots with a granularity of 10 mins each. This provides 144 slots in a day. So, to plot 14 days' traffic, 2016 time slots are required. However, there are multiple time instances where the associated traffic is not recorded in the dataset. Such rows are eliminated during the data cleaning phase, as explained in Sect. 2.

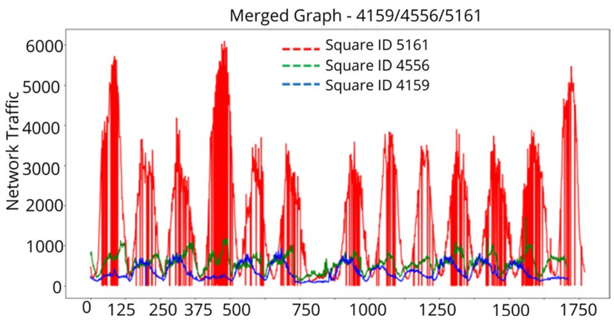


Fig. 9 Superimposed time series plots for square ID 4556, 4159, and 5161

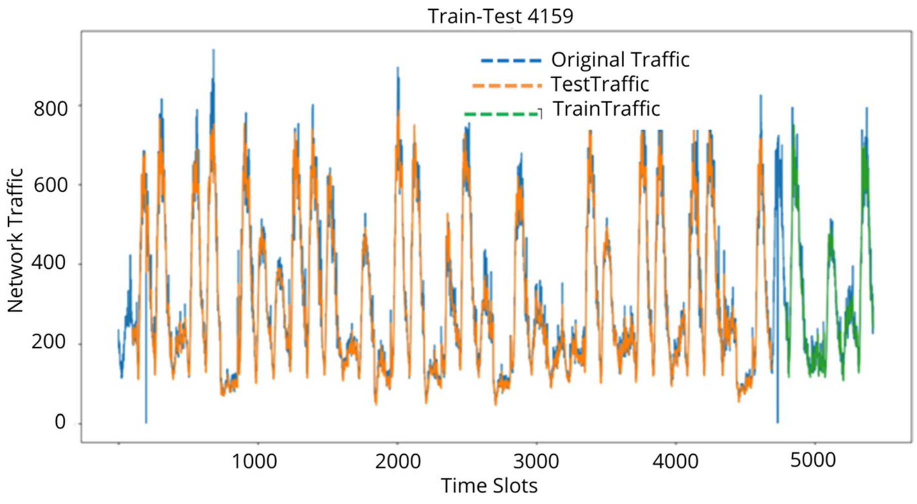


Fig. 10 Superposed time series of original and predicted traffic for area 4159 using modLSTM

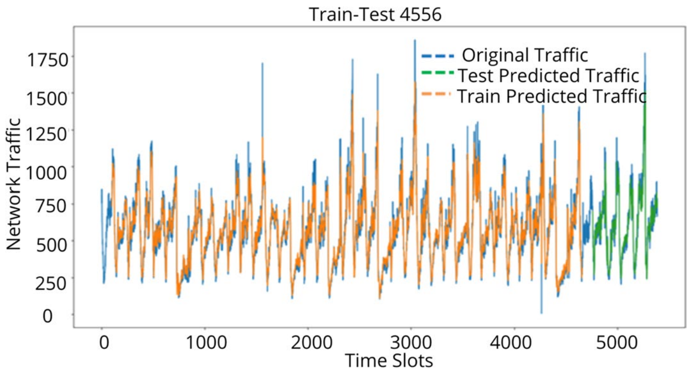


Fig. 11 Superposed time series of original and predicted traffic for area 4556 using modLSTM

Fig. 12 Time series plot of area 4556 using modLSTM before employing interpolation

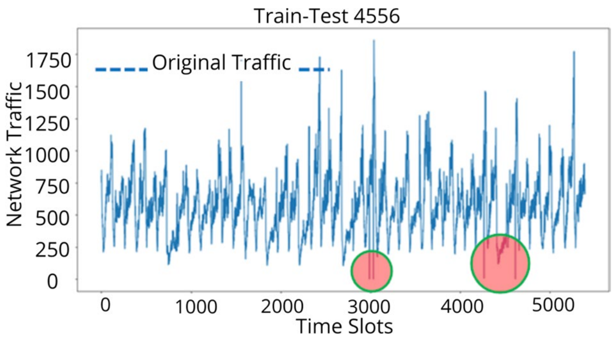
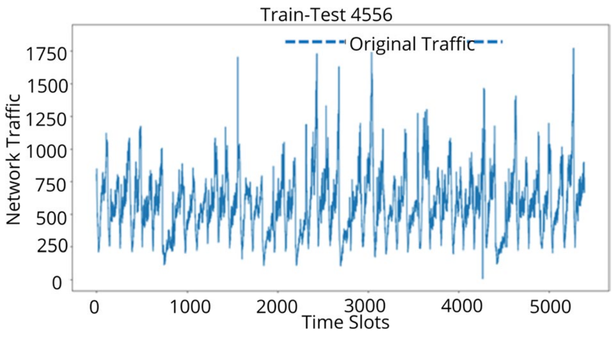


Fig. 13 Time series plot of area 4556 using modLSTM after employing interpolation



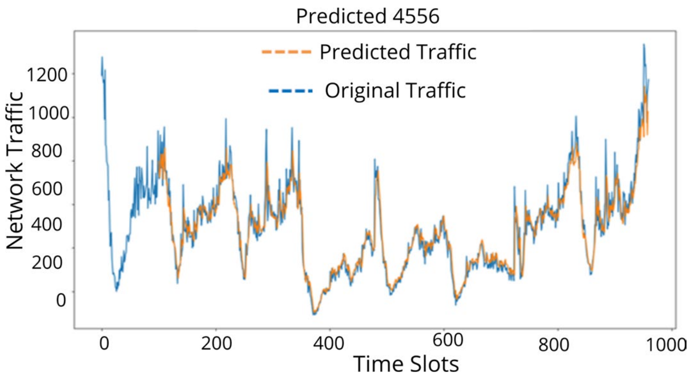


Fig. 14 Superposed time series of original and predicted traffic for area 4556 using modLSTM

The process of finding the area with the highest traffic during the 2 months is described below:

- As specified in Sect. 2, the data frame with 10,000 rows was initially obtained.
- Now, the max function defined in the pandas library was applied to identify the area with the highest traffic for 2 months.
- Next, the associated $square_ID$ was identified from the row having the largest sum_total value. Here, it is 5161 with a total traffic of 11996830.052655682.

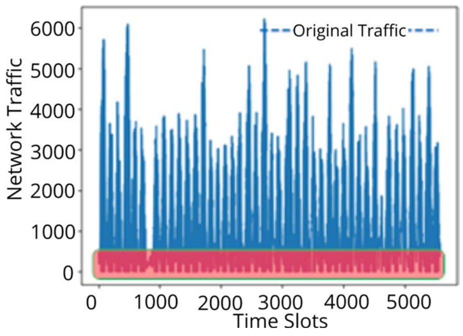


Fig. 15 Time series plot of area 5161 using modLSTM without interpolation

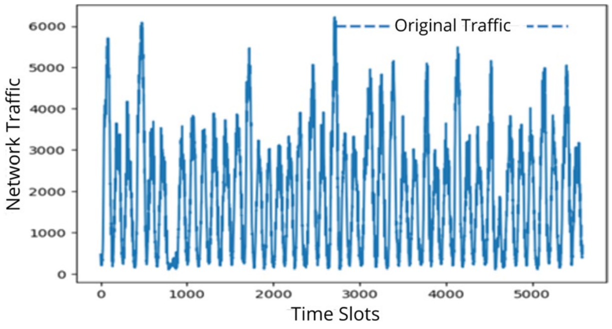


Fig. 16 Time series plot of area 5161 using modLSTM with interpolation

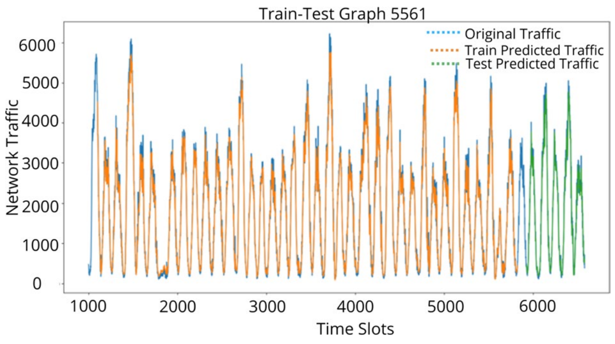
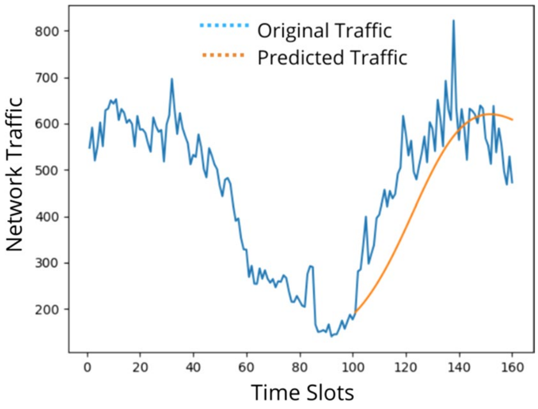


Fig. 17 Superposed time series of original and predicted traffic for area 5161 using modLSTM

- According to an analysis of 10,000 square grids for the entire city of Milan, the location of the square grid with the highest traffic flow for 2 months is in the city's center. This may be due to high population density or the high mobility of the city crowd towards the city center. Additionally, the analysis reveals that the cellular traffic within the correlated locations varies dynamically in relation to temporal changes.
- To further validate the time series plot of 14 days' traffic, the areas with square IDs 4159, 4556, and 5161 are demonstrated in Fig. 9.

The following dynamic temporal observations are drawn from Fig. 9:

Fig. 18 Prediction using conventional LSTM



- The areas with square IDs 4556 and 4159 are spatially related because their 14-day traffic variations are similar. In addition, these areas are distant from the city center. Consequently, lower traffic than grid-5161 was observed in these regions.
- The area with square ID 5161 has the highest traffic density among the 10,000 areas.
- All three areas show the highest peak during the mid-forenoon or post-mid time in the evening. This corresponds to the time slot number between 72 and 82 every day.
- The recurrent lowest traffic for all three areas is observed during the range of 125 to 144 slot numbers every day. These time slots correspond to midnight.
- There is a long-time decline between slot numbers 750 and 900 in all three graphs. The time statistics for training and prediction for IDs 4159, 4556, and 5161 are given in Table 3. It is apparent from the table that the modLSTM takes equivalent time of 3.27, 4.05, and 3.27 mins for training on the dataset captured for square ids 4159, 4556, and 5161 respectively. Also, the model reported lowest prediction time of 0.14 s for square id 4159. The low training and prediction time improves the real-life applicability of the model.

The time series analysis provides insights into the temporal dynamics of cellular traffic in different areas of the city. It helps to identify patterns and variations in traffic flow, enabling a better understanding and predicting network demands.

2.7 Predictive analysis using modLSTM

We run the modLSTM algorithm to forecast traffic in three geographical areas with IDs 5161, 4159, and 4556 during the week from December 16 to 22. At each time t, the algorithm receives history x_t of an area a as input. Here, a is a vector of traffic values in past time intervals up to time t. The algorithm estimates future traffic at time $t + 1$ in an area a, denoted as $x_a(t + 1)$. Here, $x_a(t)$ represents the traffic observed in area a during time interval t.

3 Results

For training, validation, and testing, the initial 44 days dataset was chosen. 86.5% of this data was selected for training and the remaining 13.5% was used for testing. The selection of above-mentioned split is influenced by both standard practices in ML research and specific characteristics of our dataset. A split close to 80–20% is used in ML techniques for training and testing respectively. For our time-series dataset, an allocation of 86.5% dataset for training provides more historical data points to the proposed LSTM model. This enhances the learning of temporal dependencies and improves model performance.

Whereas, 13.5% of the data used for testing includes approximately six days of traffic data over 850 data points. This sample size is substantial enough to provide a rigorous assessment of the model's predictive performance. This test dataset effectively captures the volatility and complex patterns in real-world network traffic data.

Let's consider the example of area 4159, where we had 4690 samples for training and 733 samples for testing. The forecast was made from day 45 (Dec. 15) to day 52 (Dec. 22). To begin the prediction, the algorithm requires a minimum of 100 timestamps. Therefore, 144 timestamps from the previous day (Dec. 15) were used to initiate the prediction on day 46 (Dec. 16). Our model reported the minimum Mean Absolute Percentage Error (MAPE) of 8.80% on the training data and 8.68% on the testing data, suggesting that it can effectively generalize to unseen data. The prediction errors reported for the area 4159, using modLSTM are illustrated in Table 2. Further, the superposed time series of original traffic and the predicted traffic for the training and testing data of area 4159 is shown in Fig. 10.

Next, the time series of original traffic for the area with square ID 4556 is shown in Fig. 11. As highlighted in Fig. 11, there are multiple time instances in which the actual traffic load is zero, which would eventually cause a divide by zero error while calculating MAPE. This resulted in a high MAPE value. To resolve this issue, the interpolation technique was employed. The interpolation technique plays a crucial role in handling instances where the actual traffic load is zero. In our time-series dataset, there are multiple time instances with zero actual traffic load. When these instances are encountered during the calculation of MAPE, a divide by zero error occurs, resulting in an inaccurately high MAPE value.

Table 2 Error reporting in area 4159's prediction using modLSTM

Parameters	Values
Train data size with 4690 samples	
Mean squared error (MSE)	36.2289857598838
Mean Absolute Error (MAE)	24.822641116789057
Mean Absolute Percentage Error (MAPE)	8.806629277908165
Test data with 733 samples	
Mean squared error (MSE)	34.635568813272855
Mean absolute error (MAE)	24.4047072083763
Mean absolute percentage error (MAPE)	8.686576076171716
Predicted data size (16–22 Dec.) with 954 samples	
Mean squared error (MSE)	47.71699444362534
Mean absolute error (MAE)	25.791052422418424
Mean absolute percentage error (MAPE)	9.254709424886051

In such a case, interpolation estimates an appropriate traffic value based on neighboring known data points. This is helpful in smoothing the data and preventing the divide by zero errors while calculating MAPE.

This approach helps to maintain the integrity of the error metrics and ensure the high performance of the model as evidenced by low MAPE values. The calculation of MAPE is done using Eq. (10).

$$MAPE = \frac{1}{n} \sum \left| \frac{x_t - X_i}{x_t} \right| \tag{10}$$

In Eq. 10, variable n represents the number of times the summation iteration occurs, x_t and X_t are the actual and forecasted values respectively. Zero components were eliminated by interpolating between two sequences of values.

3.1 Time statistics

We also calculated the training time and prediction time as shown in Table 3. For instance, if there is a series of 12, 4, 5, 0, 1 with zero in between the series, then it is converted to 12, 4, 5, 3, 1 using interpolation. Figures 12 and 13 show the time series plot of the area with ID 4556 before and after employing the interpolation technique, respectively. Interpolation is performed to eliminate internet traffic volume values less than 5. Now, the superposed time series plot of original traffic and projected traffic for the area with square ID 4556 is depicted in Fig. 14. Various error values for the traffic prediction in the area with ID 4556 are shown in Table 4.

Similarly, with the same configurations of the LSTM model, the prediction of the area with square ID 5161 was made. The interpolation for square ID 5161 was done to eliminate the values less than 100. Figures 15 and 16 demonstrate the time series plots of square ID 5161 without and with interpolation, respectively. The graph shown in Fig. 16 demonstrates that the interpolation technique resolves the divide-by-zero error. The superposed time series of original traffic and predicted traffic for the training and test data of area 5161 is shown in Fig. 17. Various error values for the traffic prediction in area 5161 are shown in Table 5. Similarly, the errors reported for the area with square ID 4159 by the conventional LSTM trained for five epochs with a batch size of 10 are illustrated in Table 6

3.2 Comparative analysis

We compared the performance of the conventional LSTM model with the proposed ’modLSTM’ architecture. The performance of LSTM is depicted in Fig. 18. The prediction was made on day 45 (Dec. 15) using 160-time slots. Here, 100-time stamps were used for training, and the remaining 60 were projected.

Table 3 Time statistics for training and prediction using modLSTM

Area square ID	Training time	Prediction time (train and test)	Actual prediction time (16–22 Dec.)
4159	0:03:27.203367	0:00:01.138399	0:00:00.145075
4556	0:04:05.532771	0:00:01.058851	0:00:00.158297
5161	0:03:26.402917	0:00:01.063905	0:00:00.150319

Table 4 Error reporting for area 4556's prediction plot using modLSTM

Parameters	Values
Train data size with 4661 samples	
MSE	81.15021750878684
MAE	52.92640253669787
MAPE	12.94365888659903
Test data with 728 samples	
MSE	83.22860782347028
MAE	54.41319143043707
MAPE	8.63025116665602
Predicted data size (16–22 Dec.) with 960 samples	
MSE	65.00400051726389
MAE	42.8617247216744
MAPE	8.577069840272993

Table 5 Error reporting for area 5161's prediction plot using modLSTM

Parameters	Values
Train data size with 4827 samples	
MSE	212.61319849905377
MAE	145.19385903463774
MAPE	11.290565010773158
Test data with 754 samples	
MAE	207.65285264956663
MAE	144.87526282651507
MAPE	10.218367759524535
Predicted data size (16–22 Dec.) with 1012 samples	
MAE	167.82082759231199
MAE	117.33325957676803
MAPE	10.31944166159958

Table 6 Error reporting for area 4159's prediction Plot using LSTM

Parameters	Values
MSE	14.35393857041777
MAE	12.083134326977998
MAPE	21.01

Before delving into the specific advantages of the proposed modLSTM approach, it is beneficial to establish a context by understanding the strengths and weaknesses of traditional and DL-based time-series prediction models. Models such as random forests, linear regression, and ARIMA are applied for time-series prediction. These models have certain limitations, especially when compared with LSTM-based models. For example, random forest lacks in considering the sequential nature and potential temporal dependencies in time-series data (Breiman, 2001). This can lead to suboptimal predictions when dealing with data in which previous values significantly influence future values. Such data on cellular traffic has been used in this manuscript. The inherent limitation of random forests is represented in Eq. 11:

$$Y = f(X) + \epsilon \tag{11}$$

Here Y is the output, X is the input, f represents the random forest function, and ϵ is the error term. This equation shows that the random forest model does not account for any temporal information between different instances of the input data. Similarly, linear regression models lack in dealing with non-linear patterns in the data and may require a transformation of the data to handle trends and seasonality (Hastie et al., 2019). Moreover, the model assumes independence of errors, which is not applicable to the time-series data. This can be visualized from the general equation of linear regression 12:

$$Y = aX + b + \epsilon \tag{12}$$

where Y is the output, X is the input, a and b are model parameters, and ϵ is the error term. As can be seen from the equation, linear regression does not consider any temporal dependency between different instances of X.

The ARIMA model is effective for time-series data but it lacks in dealing with volatile data with complex temporal structures (Box et al., 2015). The model also works on the assumption of linearity in the dataset. Thus, its' performance degrades for non-linear datasets. The functionality of the ARIMA model is shown in Eq. 13:

$$Y_t = c + \sum_{i=1}^{p} \phi_i Y_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} + \epsilon_t \tag{13}$$

In Eq. 13, Y_t is the output at time t, c is a constant, ϕ_i and θ_i are the AR and MA parameters respectively, and ϵ_t is the error term at time t. Among DL models, Gated Recurrent Units (GRUs) are known for their efficiency in handling time-series data (Cho et al., 2014). However, GRUs lacks in dealing with very long sequences due to their gating mechanism. This may lead to poor performance in learning long-term dependencies (Cho et al., 2014). In contrast, LSTM and the proposed LSTM (modLSTM) are designed to handle time-series data with long-range dependencies (Hochreiter & Schmidhuber, 1997) and can learn complex temporal structures. The model modLSTM incorporates actual data into the predictions, reduces the propagation of error (Hochreiter & Schmidhuber, 1997), and proves more robust than the traditional LSTM. Based on the above discussion, it is apparent that the proposed approach offers an effective solution for this cellular traffic forecasting task. The working of modLSTM is demonstrated using Eq. 14.

$$\tilde{C}t = tanh(W_C \cdot [ht - 1, x_t] + b_C + r_t) \tag{14}$$

In Eq. 14, r_t represents the actual data at time t. This data is incorporated into the LSTM structure to correct errors and improve the robustness of the model.

To validate the superiority of our approach, we conducted extensive experiments comparing the performance of the modLSTM with the aforementioned models on the cellular traffic dataset. The results consistently demonstrated the higher accuracy and reliability of the modLSTM model, which consistently outperformed the other models in terms of error rates, as shown in Table 6 and The comparative performance of modLSTM against these conventional and DL-based approaches demonstrated in Table 6.

The above discussion and results presented in Table 6, prove the supremacy of the modLSTM model over LSTM model applied for time-series forecasting. Thus, it is beneficial for performing challenging tasks such as predicting cellular traffic activity.

3.3 Impact, practical implications, and limitations

The modLSTM architecture extends its utility far beyond network traffic prediction. Its' unique capacity to tackle time-series prediction tasks with remarkable accuracy and adaptability make it a promising tool for a diverse range of applications.

For instance, in the financial sector, accurate and reliable time-series predictions can be a game-changer. The proposed modLSTM model potentially revolutionizes market trend forecasting and risk assessment. It also finds promising applications in meteorology, where it may enhance the accuracy of weather predictions (Chavan et al., 2023), which plays a crucial role in agriculture, disaster management, and tourism. Further, it can accurately predict energy demand patterns, aiding in optimal resource allocation and planning. Moreover, it can be employed in healthcare to predict disease outbreak patterns. Thus, provides assistance in proactive public health measures and resource management.

Although the proposed model has potential applications, its' limitations leave scope for further improvements. First and foremost, the performance of the model is heavily contingent on the quality and structure of the input data. The presence of unstructured data or a high volume of null values may potentially undermine the accuracy of the model. This can be overcome by incorporating robust data preprocessing methods.

Additionally, the modLSTM model was primarily tested on data sourced from an urban area. Consequently, its' performance may exhibit variability when applied to rural or less densely populated areas. We encourage future research to explore the model's adaptability to diverse demographic and geographical contexts, which would further bolster its universal applicability.

Finally, the RNN-modLSTM architecture represents a significant stride forward in the domain of time-series prediction. Its ability to handle complex patterns and dependencies, combined with its adaptability to different application scenarios, makes it a promising avenue for future research and practical applications.

4 Conclusion

In this manuscript, the RNN and LSTM based modLSTM architecture is developed. Firstly, the data preprocessing is performed to remove missing entries from the dataset and resolve the divide by zero error. Now, RNN-modLSTM was applied to predict the network traffic. The proposed architecture reported the minimum MAPE of 8.68% for 733 testing samples from the dataset collected for the square ID 4159. The low error proves the potential of modLSTM to accurately predict the internet traffic for the specified grids from a specific location. It correctly analyzes the continuously varying spatiotemporal network traffic data generated in an urban area using wireless networks.The model reported 61.9% reduction in MAPE reported by the LSTM model. Moreover, the interpolation technique was introduced to handle the multiple instances of zero traffic values-captured in the dataset. Interpolation further improved the prediction accuracy, is useful in network optimization and hence important for operators in judicious resource allocation. The efficacy of modLSTM was evaluated in three different scenarios. In the first scenario, as shown in grid 4159, there was a smooth curve of internet traffic due to low variation in data usage. In the second scenario, as illustrated in grid 4556, there was moderate variation in data usage, resulting in less smooth curves. In the third scenario, as shown in grid 5161, there was a huge variation in internet usage, resulting in rough curves. Based on the experimental results, it is concluded that the modLSTM accurately predicted the traffic in all three scenarios with MAPE of 8.68, 8.63, and 10.21 for testing samples of square IDs 4159, 4556, and 5161 respectively. Thus, prove the robustness of the proposed architecture. The lowest training time of 3.26 mins and prediction time of 0.14 s make the modLSTM useful for real-time network traffic prediction. The high adaptability and predictive accuracy make the modLSTM a versatile tool for any scenario where time-series prediction is essential. It is efficient in handling time-series data for financial forecasting, weather prediction, energy demand prediction, and disease prediction. But the performance of the model depends significantly on the quality and structure of input data. The model may report poor performance for unstructured data or data containing many null values. For this research, the model is evaluated using data from a specific urban area. Its' performance may vary when applied to rural or less densely populated areas. Presently, we are working on developing a generalized model for unstructured, structured, dense, and sparse datasets.

Acknowledgements We acknowledge Manipal University Jaipur for providing the IoT and Machine Learning Lab (114-2AB) for completing the research work presented in this manuscript.

Author contributions All the authors have equal contributions to this research and the preparation of the manuscript.

Funding The research presented in this manuscript is supported by SIMPATICO_ZUMPANO-Progetto di Ricerca POR SIMPATICO: Sistema Informativo Medico Patologie Complesse 3D.

Availability of data and material The dataset used in this research is available online. The references and citations are provided in this manuscript.

Code availability The code is available with the authors. They will upload the modular and executable code to an open-source platform after the completion of the review process.

Declarations

Conflict of interest The authors declare that there is no conflict of interest associated with this research.

Ethical approval The experiments were performed on data not related to living beings. No human beings and animals were involved in conducting the experiments. Thus, approval from an ethical committee is not required.

Consent to participate No living beings are involved in conducting the experiments for this research. Thus, consent to participate is not required.

Consent for publication The authors involved in conducting this research give their consent for the publication of the article titled "Forecasting and Spatio–Temporal Analysis of Mobile Network Traffic using modLSTM".

References

Barlacchi, G. (2015). A multi‾source dataset of urban life in the city of Milan and the province of Trentino. *Scientific Data, 2*, 1–15.

Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time Series Analysis: Forecasting and Control*. Hoboken: Wiley.

Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32.

Chavan, R. S., Srivastava, G., & Pradhan, N. (2023). Advance plant health monitoring and forecasting system using edge-fog-cloud computing and lstm networks. In Mathur, G., Bundele, M., Tripathi, A., Paprzycki, M. (eds.) *Proceedings of 3rd International Conference on Artificial Intelligence: Advances and Applications. Algorithms for Intelligent Systems*. Springer, Singapore . https://doi.org/10.1007/978-981-19-7041-2_26.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y. (2014). Learning phrase representations using rnn encoder–decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

Cisar, P., & Cisar, S. M. (2012.) Fitting univariate distributions to computer network traffic data using gui. In *2012 IEEE 13th International Symposium on Computational Intelligence and Informatics (CINTI)*, (pp. 285–288).

Ericsson. (2021). Mobile data traffic outlook. https://www.ericsson.com/en/reports-and-papers/mobility-report/data-forecasts/mobile-traffic-forecast.

Foundation, P. S. (2020). Python 3.8.10 . https://www.python.org/downloads/release/python-3810/.

Harris, C. R., Millman, K. J., Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., & Smith, N. J. (2020). Array programming with numpy. *Nature, 585*(7825), 357–362.

Harvard. (2013). *URL* : *HarvardDataverse*. https://dataverse.harvard.edu/dataset.xhtml?.

Hastie, T., Tibshirani, R., & Friedman, J. (2019). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Berlin: Springer.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.

Hunter, J. D. (2007). *matplotlib* : *A2dgraphicsenvironment*. *Computing in Science & Engineering, 9*(3), 90–95.

Intel: Intel Tiger Lake (2021). https://www.intel.com/content/www/us/en/products/docs/processors/core/11th-gen-processors-brief.html.

JetBrains. (2021) . *PyCharm*2021.3. https://www.jetbrains.com/pycharm/.

Kotsiantis, S., Kanellopoulos, D., & Pintelas, P. (2006). Data preprocessing for supervised learning. *International Journal of Computer Science, 1*(2), 111–117.

Li, M., Wang, Y., Wang, Z., & Zheng, H. (2020). A deep learning method based on an attention mechanism for wireless network traffic prediction. *Ad-Hoc Network, 107*, 1–11.

Ltd., C. (2020). *Ubuntu*20.04.3*LTS*. https://releases.ubuntu.com/20.04/.

Lu, H., & Yang, F. (2018). A network traffic prediction model based on wavelet transformation and lstm network. In *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)*, (pp. 1–4).

McKinney, W. (2010). Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, (pp. 56–61).

Montgomery, D. C., Jennings, C. L., & Kulahci, M. (2015). *Introduction to Time Series Analysis and Forecasting*. Hoboken: Wiley.

Munz, G., & Carle, G. (2007). Traffic analysis of ipv6 packets in a large scale ipv6 test network. In *10th IEEE/IFIP Network Operations and Management Symposium*.

NVIDIA. (2020). NVIDIA GeForce RTX 3080. https://www.nvidia.com/en-us/geforce/graphics-cards/30-series/rtx-3080/.

Patel, P. (2019). Hard disk drive (hdd) technology. *International Journal of Scientific & Engineering Research, 10*(1), 1614–1617.

Rani, G., Thakkar, P., Verma, A., Mehta, V., Chavan, R., Dhaka, V. S., Sharma, R. K., Vocaturo, E., & Zumpano, E. (2022). *kub* − *unet* :segmentation of organs of urinary system from a kub x-ray image. *Computer Methods and Programs in Biomedicine, 224*, 1–14. https://doi.org/10.1016/j.cmpb.2022.107031

Wang, S., Zhuo, Q., Yan, H., Li, Q., & Qi, Y. (2019). A network traffic prediction method based on lstm. *ZTE Communications, 17*(2), 19–25.

Zhang, C., Zhang, H., Qiao, J., Yuan, D., & Zhang, M. (2019). Deep transfer learning for intelligent cellular traffic prediction based on cross–domain big data. *IEEE Journal on Selected Areas in Communications, 37*(6), 1389–1401.

Zhao, S., Chen, S., Sun, Y., Cai, Z., & Su, J. (2019). Identifying known and unknown mobile application traffic using a multilevel classifier. *Security and Communication Networks*. https://doi.org/10.1155/2019/9595081

Authors and Affiliations

Vidyadhar J. Aski[1,2] · Rugved Sanjay Chavan[1,4] · Vijaypal Singh Dhaka[1] · Geeta Rani[1] · Ester Zumpano[3,5] · Eugenio Vocaturo[3,5]

✉ Geeta Rani
geetachhikara@gmail.com

Vidyadhar J. Aski
vidyadharstjit@gmail.com

Rugved Sanjay Chavan
rugvedm12@gmail.com

Vijaypal Singh Dhaka
vijaypalsingh.dhaka@jaipur.manipal.edu

Ester Zumpano
e.zumpano@dimes.unical.it

Eugenio Vocaturo
e.vocaturo@dimes.unical.it

[1] Computer and Communication Engineering, Manipal University Jaipur, Jaipur, Rajasthan 303007, India

[2] Emerging Technology CoE, Enterprise Applications, Mahindra Digital Engine, Mahidnra and Mahindra Ltd, Mumbai, Maharashtra, India

[3] Department of Computer Engineering, Modeling, Electronics and Systems (DIMES), University of Calabria, Rende(Cosenza), Italy

[4] Computer Science, University of Virginia, Charlottesville, USA

[5] National Research Council, Institute of Nanotechnology (NANOTEC), Rende(Cosenza), Italy