Multivariate Congestion Prediction using Stacked LSTM Autoencoder based Bidirectional LSTM Model

Vijayalakshmi B[1*], Thanga Ramya S[2], and Ramar K[3]
[1] Department of Computer Science and Engineering, Ramco Institute of Technology,
Rajapalayam-626117, Tamil Nadu, India
[e-mail: bviji.akila@gmail.com]
[2] Department of Computer Science and Design,
R.M.K. Engineering College, Chennai-601206, Tamil Nadu, India
[e-mail: thangaramya@gmail.com]
[3]Department of Computer Science and Engineering, R.M.K. College of Engineering and Technology,
Chennai-601206, Tamil Nadu, India
[e-mail: kramar.einstein@gmail.com]
[*]Corresponding author: Vijayalakshmi B

Received July 28, 2022; revised December 21, 2022; accepted December 31, 2022; published January 31, 2023

Abstract

In intelligent transportation systems, traffic management is an important task. The accurate forecasting of traffic characteristics like flow, congestion, and density is still active research because of the non-linear nature and uncertainty of the spatiotemporal data. Inclement weather, such as rain and snow, and other special events such as holidays, accidents, and road closures have a significant impact on driving and the average speed of vehicles on the road, which lowers traffic capacity and causes congestion in a widespread manner. This work designs a model for multivariate short-term traffic congestion prediction using SLSTM_AE-BiLSTM. The proposed design consists of a Bidirectional Long Short Term Memory(BiLSTM) network to predict traffic flow value and a Convolutional Neural network (CNN) model for detecting the congestion status. This model uses spatial static temporal dynamic data. The stacked Long Short Term Memory Autoencoder (SLSTM AE) is used to encode the weather features into a reduced and more informative feature space. BiLSTM model is used to capture the features from the past and present traffic data simultaneously and also to identify the long-term dependencies. It uses the traffic data and encoded weather data to perform the traffic flow prediction. The CNN model is used to predict the recurring congestion status based on the predicted traffic flow value at a particular urban traffic network. In this work, a publicly available Caltrans PEMS dataset with traffic parameters is used. The proposed model generates the congestion prediction with an accuracy rate of 92.74% which is slightly better when compared with other deep learning models for congestion prediction.

Keywords: Autoencoder, Bidirectional LSTM, Convolutional neural network, Congestion prediction, Spatio-temporal data, traffic flow forecasting.

1. Introduction

The proliferation of rideshare and delivery services, as well as high population density and the -rise of automobiles and associated infrastructure, have all increased traffic congestion. [1] The accuracy of the traffic congestion prediction helps to improvise the transportation management system such as route planning during peak hours that aims to reduce travel time. It also assists travel agencies with travel demand prediction and order dispatching services. Traffic congestion creates the problems such as air pollution [2] due to the emission of gases [3] during the waiting time which increases fuel consumption and time. The prediction of upcoming traffic congestion status is also essential to guide autonomous vehicles by tracking the number of moving vehicles [4] and predicting the traffic flow to avoid congestion. The key to minimizing traffic congestion is gathering and analyzing diverse Intelligent Transportation System (ITS) data and performing a traffic forecast ahead of time.

The data needed for the traffic prediction task includes the road network details such as road intersection, traffic flow, speed of the vehicles on the particular road, number of lanes, travel time, and region-based information like travel demand, regional flow, and trajectory. Apart from these details, the event data such as holidays, festivals, incidents, and metrological data also play a major role. This spatiotemporal data can be classified into static spatial-temporal data (event data), spatial static temporal dynamic data which includes traffic flow, speed, time, and meteorological data, and dynamic spatial-temporal data (trajectory). Here, the Spatial Static Temporal Dynamic data is used for predicting the short-term congestion status.

The average velocity or traffic density of vehicles traveling on a particular road section is known as traffic congestion [1]. The traffic congestion at a particular location is varied depending on the time of analysis. Traffic congestion can be measured by density [5], traffic volume, travel time, vehicle occupancy, and traffic congestion index [6]. All these parameters play a major role in predicting the congestion status. The traffic congestion prediction using multi-parameters has higher accuracy compared with the prediction using a single parameter. Apart from the spatiotemporal parameters, external elements like weather and special events are also the sources of non-recurring traffic congestion [7]. The traffic congestion details for the next 30 minutes can help the users to provide alternate traveling suggestions to avoid the occurrence of congestion.

Predictions of traffic congestion have been the subject of extensive research. Various techniques for predicting traffic congestion have been proposed in the literature. Regarding the variables assessed to forecast the congestion and the information utilized for the forecast, each of the technique varies. The majority of the architectures and models have shallow natures. The performance of the shallow architecture degrades as the amount of data grows. Deep architecture-based solutions are presented for many forecasting applications [8], [9] that include massive data and complicated processing. Many optimization methods [10], [11] were employed in complex applications for optimizing the hyperparameters of this deep architecture to increase the performance metrics.

The proposed model combines traffic flow prediction (TFP) and congestion status detection (CSD) modules for short-term congestion prediction. This modular approach increases prediction accuracy by separating traffic flow forecasting from congestion prediction. Feature extraction (FE) and flow prediction (FP) are two submodules that make up the TFP module. The FE module helps in obtaining weather features with a reduced dimensionality using a stacked autoencoder. Short-term traffic flow is predicted by the FP module, and the module's effectiveness is assessed via rollout cross-fold validation.

The CSD module contains a CNN model that detects the congestion state based on the

results of the TFP module. The temporal dependency in the time series traffic data led to the selection of forward chaining or rollout cross fold validation. By maintaining the temporal order while separating the data into training and testing sets, the forward chaining validation technique helps in measuring the model performance. It performs evaluations with less computational complexity and generates models with higher accuracy, which is a crucial criterion for a system.

The contribution of the proposed research work is given below.

- A multivariate model for predicting the traffic congestion status is designed using a Stacked LSTM AE based BiLSTM deep learning network.
- Autoencoder model is used in the feature extraction module to reduce the dimensionality of the weather data to get latent features that can improve the accuracy of the prediction.
- The BiLSTM model is proposed which considers the spatial correlation from the upstream and downstream stations to find out the long-term dependency of the spatiotemporal traffic data. This traffic data is combined with the encoded weather data to perform the traffic flow prediction for the time range of 5 to 30 minutes. The model parameters are optimized and evaluated using the forward chaining cross fold validation technique.
- Finally, a Convolutional Neural network is used for detecting the congestion status as normal, light, and heavy congestion at a particular station from the predicted traffic flow value.
- The congestion prediction accuracy results exhibit that this model outperforms other methods during the weekdays as well as weekend and peak hours also.

The paper is structured as the literature review is presented in section 2. The introduction of the problem and the proposed methodology are given in Section 3. Section 4 explains the dataset and experimental part. Section 5 describes the result and comparison with the existing approaches. Section 6 elaborates on the conclusion and future work.

2. Literature Survey

In intelligent transportation systems, traffic flow forecasting is a preliminary step for traffic congestion prediction. The major categories of traffic parameter prediction are parametric and non-parametric methods. A detailed description and analysis of various methods used in short-term traffic forecasting and its applications were discussed in [12-14]. Mahmuda Akhtar et al. [15] summarized the machine learning techniques for congestion prediction and described their strengths and challenges. Nishant Kumar et al. [16] presented a detailed analysis of deep learning methodologies and their variants in congestion detection and prediction.

Traffic forecasting started with the naive approaches like Historical average and parametric approaches Box-Jenkins method, ARIMA, SARIMA, Markov chain [17], and Kalman filter [18] which uses statistical methods and requires the predefined model for predicting the traffic forecasting. The nonparametric approaches were introduced to overcome the drawbacks of the naive approaches for predicting traffic parameters. The artificial neural network [19] based forecasting model was developed and it helped to design and represent the non-linearity spatiotemporal nature of the traffic information with prior domain knowledge.

The recent advances in machine learning and deep learning make the deep neural network architecture such as convolutional neural network and recurrent neural network(RNN) variations can be used extensively in the forecasting process. The parameters of the model are optimized with optimization algorithms [20] to improve the accuracy. S. Neelakandan et al. [21] designed a stacked sparse autoencoder based model in which the autoencoder uses traffic

and weather data to perform the traffic flow prediction. C.H. Chou et al. [22] designed a stacked deep ensemble LSTM model to predict the long term traffic time. This model integrated the weather data to predict the traffic time during rush hours.

An approach for determining congestion level was discussed [23] in which the road occupancy rate at a given moment is taken as a key criterion for evaluating traffic congestion. M. Chen et al. [24] proposed a deep CNN model to predict the short term traffic congestion using the local temporal dependencies and multi-grained features. CNN model applies multiple convolution operations to the periodic traffic data to perform the prediction.

S.Zhang et al. [25] proposed a deep congestion prediction network that uses a feature learning architecture inspired by a deep autoencoder and it predicts the short-term traffic congestion from the temporal correlations from historical traffic congestion data of a transportation network. T.Sun et al. [26] proposed a bidirectional spatial-temporal network using LSTM and convolutional neural network which are utilized to find the temporal aspects of traffic congestion evaluation and extract spatial information respectively. Both weather information and traffic speed were used in this model. The spatial and temporal nature of the traffic data leads to the development of hybrid/ensemble models. There may be a different combination of deep learning architecture such as CNN & RNN or statistical and deep learning approaches. A hybrid model such as CNN-LSTM with attention model [27], and LSTM-XGboost [28] uses a combination of two models for prediction. The summary of various traffic forecasting methods is listed in **Table 1**.

Table 1. Summary of related forecasting techniques

Ref. No	Model Name	Data Set Used	Primary Data	Secondary Data	Predicted traffic parameter
[21]	Stacked sparse autoencoder	China Metro freeways in the Twin Cities	Traffic data	Weather data	Traffic flow
[22]	Deep ensemble stacked long short term memory	Taiwan Dataset	Traffic time & Speed	Weather data	Traffic time
[24]	Deep CNN	China dataset	Vehicle passage records	-	Congestion
[25]	Deep autoencoder	Seattle area traffic congestion status dataset	Traffic congestion map	-	Congestion
[26]	Bidirectional LSTM	China Dataset	Speed	Weather data	Speed
[27]	Attention-based CNN & LSTM	PEMS dataset	Traffic flow & Speed	-	Traffic Flow
[28]	LSTM-XGboost	Shenzhen Cup dataset	Traffic speed	-	Speed
[29]	DBN	PEMS dataset	Traffic Flow	Weather data	Traffic Flow
[30]	Stacked Autoencoder & Radial Basis Function	China metro freeways in the Twin Cities	Traffic flow	Weather data	Traffic Flow
[31]	LSTM and CNN	China Highway traffic dataset	Traffic flow	Weather data	Travel time
[32]	CNN & BiLSTM Model	PEMS dataset	Traffic Flow	-	Traffic flow

There are prediction models in the literature that employ a variety of parameters for traffic forecasts, including speed, flow, and weather information. Based on the analysis of existing approaches, the combination of weather and traffic factors plays a crucial role in traffic forecasting and improves the model accuracy. However, adding weather information to the model training increases the amount of data and lengthens the training process. So the dimensionality reduction can be applied to the meteorological data to increase model robustness and decrease complexity. Therefore, in the proposed model, autoencoders are used to encode the weather features in order to speed up the forecast process.

The proposed model used the stacked LSTM-based autoencoder to perform feature extraction and bidirectional LSTM to forecast the short term traffic flow. The CNN model predicts the congestion state from the predicted traffic flow at a particular location, which is helpful to manage the upcoming traffic congestion status at a particular highway.

3 Proposed Methodology

3.1 Problem Definition

Traffic congestion prediction denotes the forecasting of traffic state features like the average traffic flow density at a particular location. It is a particular case of traffic forecasting in which the problem arises from traffic instability when the maximum flow is reached. The task is to find out the variable which is used to measure the congestion level of traffic and the value can be categorized into multiple levels depending on the threshold value. From the past 12 consecutive 5-minute interval traffic flow values, the upcoming congestion state is predicted. The occurrence of congestion is considered a traffic jam or heavy congestion in a particular road section if the load factor is above the average threshold ($Threshold_{Avg}$) value. The $Threshold_{Avg}$ value is calculated based on the average of every 5-minute interval traffic flow values on the particular link per day.

The predicted value of traffic flow($\hat{y}$) for the next 10 minutes is used to find out the congestion status(CS) in near future. It not only depends on the previous flow values, but it also depends on other external input parameters like weather conditions, the number of lanes on a road, and time stamp parameters (time & day of the week). The input parameters for the traffic flow prediction are the previous traffic flow values and it is represented as $Y_{1..i-1}$, and X_i denotes the weather condition such as temperature, humidity, and overall weather condition given in (1) & (2)

$$Y_{1\ldots i} = [Y_1, Y_2, \ldots . Y_{i-1}] \tag{1}$$

$$X_{1\ldots i} = [X_{1\ldots i-1\ temperature}, X_{1\ldots i-1\ humidity}, X_{1\ldots i-1\ condition}] \tag{2}$$

The output congestion state (CS) can be defined in (3)

$$CS = \begin{cases} Normal,\ if\ \hat{y} < Threshold_{avg} \\ Light\ Congestion, if\ \hat{y} \cong Threshold_{avg} \\ Heavy\ Congestion, if\ \hat{y} > Threshold_{avg} \end{cases} \tag{3}$$

Where $\hat{y}$=f($Y_{1..i-1}$, $X_{1\ldots i-1}$) and $Threshold_{Avg}$ indicates the average traffic flow value of the day.

3.2 Proposed Methodology

The proposed model uses the traffic flow details from the nearby station (upstream and downstream) for predicting congestion at a particular location. The system is designed with two modules such as the TFP module makes use of the stacked LSTM autoencoder for feature extraction and bidirectional LSTM for predicting the traffic flow. The CSD module was designed with CNN for congestion status prediction based on predicted traffic flow for the time range of 5 to 30 minutes. The model proposed for congestion prediction is shown in **Fig. 1**.

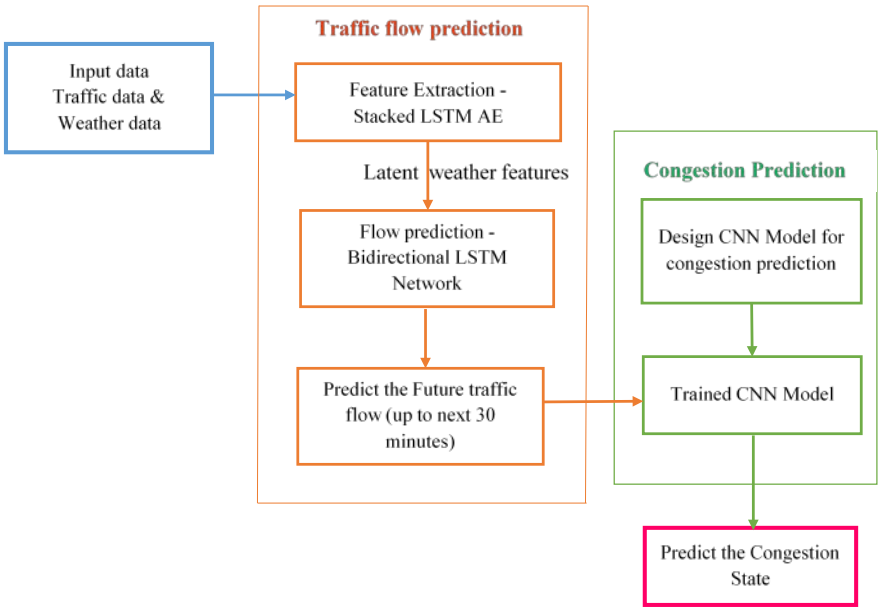


Fig. 1. Proposed SLSTM_AE-BiLSTM Model

3.2.1 Stacked LSTM Autoencoder (SLSTM AE Model)

The LSTM-based autoencoder is an unsupervised learning method that uses exclusively LSTM network for the encoder and decoder. AE learns encoding-decoding from high-dimensional input using a feedforward neural network in which the high-dimensional data is fed into the hidden layer, which encodes it to produce latent low-dimensional data. At the output layer, the decoder network reconstructs the original data.
The encoder in the autoencoder network accepts an input sequence of x_i and encodes it into a hidden representation of h_i using the formula (4).

$$\text{Encoding: } h_i(\mathrm{x}) = \mathrm{E}(w_e \mathrm{x} + b_e) \tag{4}$$

This latent data h_i is fed into the decoder network, which reconstructs the original data from it using (5)

$$\text{Decoding: } \hat{x} = \mathrm{D}(W_d h_i(\mathrm{X}) + b_d) \tag{5}$$

where $h_i(x)$ represents the i^{th} SAE layer's hidden vector output from input data x, and $\hat{x}$ denotes the final output layer's decoded output. The encoding and decoding methods are

denoted by the letters E and D. The weight matrix is denoted by w_e and w_d, while the bias value for the encoder and decoder is denoted by b_e and b_d, respectively.

The model needs to learn complicated dynamics within the temporal ordering of input sequences hence LSTM is used in this application which employs memory to maintain the data over lengthy input sequences. It can efficiently capture the temporal connections in multivariate data. Multiple LSTM stacked autoencoders are piled together in an LSTM stacked autoencoder, with the previous layer output feeding into the next layer. Each layer is trained separately, and fine-tuning is done through backpropagation to optimize the overall network performance. A supervised learning technique is employed for fine-tuning to minimize prediction error. The model aims to learn latent features and reduce the reconstruction error. If the reconstruction error calculated by (6) is small, the model has learned to accurately represent the actual data in a latent form. The decoder can now be removed from the model, leaving only the encoder to extract the latent properties of input data. The weather factors are encoded in the proposed work to deliver the best result. **Fig. 2** shows an illustration of the SLSTM AE model.

$$\text{Reconstruction Error} = \|x - \hat{x}\|^2 \tag{6}$$

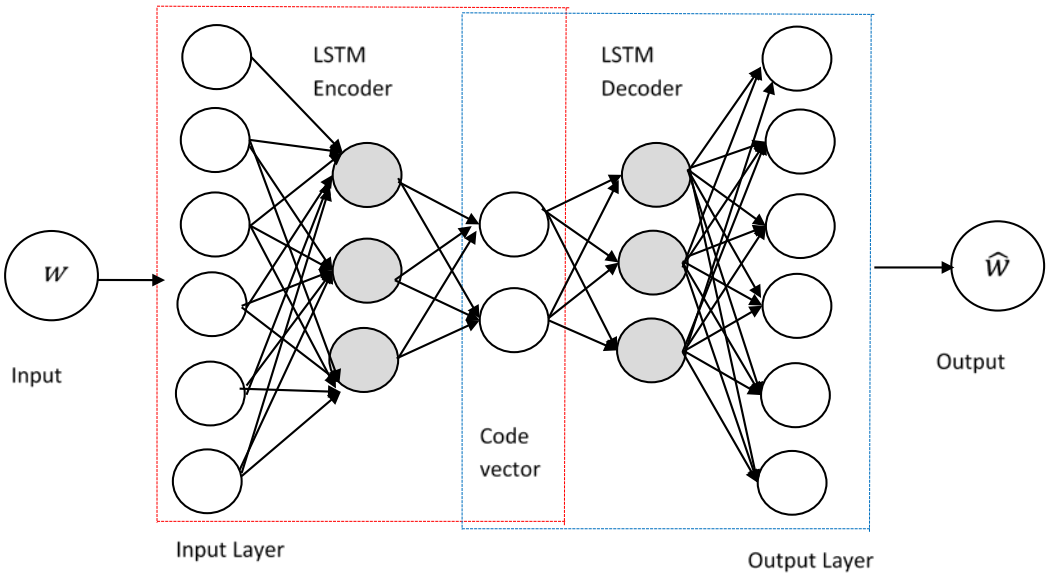


Fig. 2. Stacked LSTM AE Model

3.2.2 Bidirectional LSTM

In many areas, LSTMs outperform traditional feed-forward neural networks and RNNs. This is due to their ability to memorize patterns selectively for long periods and also it will resolve the vanishing gradient problem. The weight updating process for a particular layer in a traditional feed-forward neural network is a multiple of the learning rate, the error term from the previous layer, and the input to that layer. Hence the error term for a particular layer is the product of the errors of all prior levels. When it is toward the starting layer the gradient nearly vanishes, making it harder to train these layers. LSTM is a special kind of RNN to support long-term dependencies. Bidirectional LSTMs provide a long-range context in the input in

both directions is a variance of bidirectional RNN [33]. In a Bidirectional LSTM, there are two models, in which the first model learns the input sequence as provided in the forward direction and the second model learns the input sequence in the reverse direction. This bidirectional flow can preserve the information of both the past and future. The architectures of these models are presented in **Fig. 3**.

In this BLSTM model $\overrightarrow{h_t}$ represents the forward hidden state output which considers the input in the forward direction from layer 1 to T and $\overleftarrow{h_t}$ denotes the backward hidden sequence by the LSTM layers ranging from T to 1 at time t. The values are calculated using (7) & (8). Based on these values the output layer is updated using (9). H represents the tanh activation function.

$$\overrightarrow{h_t} = H(w_{x\overrightarrow{h_t}}\, x_t + W_{\overrightarrow{h_t}\overrightarrow{h_t}}\overrightarrow{h_{t-1}} + b_{\overrightarrow{h_t}}) \tag{7}$$

$$\overleftarrow{h_t} = H(w_{x\overleftarrow{h_t}}\, x_t + W_{\overleftarrow{h_t}\overleftarrow{h_t}}\overleftarrow{h_{t+1}} + b_{\overleftarrow{h_t}}) \tag{8}$$

$$y_t = W_{\vec{h}y}\,\overrightarrow{h_t} + W_{\overleftarrow{h}y}\overleftarrow{h_t} + b_y \tag{9}$$

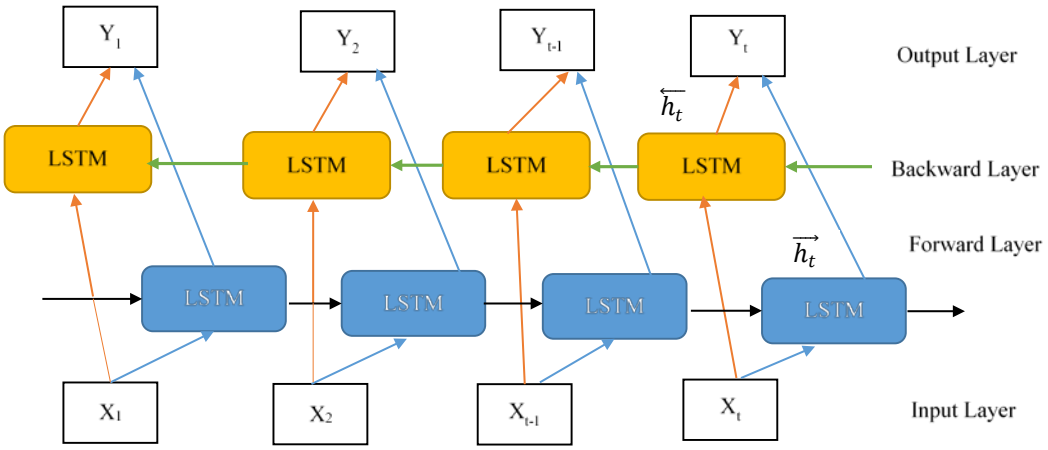


Fig. 3. Bidirectional LSTM

3.2.3 CNN

CNN is well suited to classification tasks involving data with spatial relationships. The congestion on a specific road network will be determined in our task based on the projected traffic flow from adjacent stations as well. Based on the load factor, data from a prediction interval of 5 to 30 minutes in the near future is utilized to determine the level of congestion on the road network. As a result, the Convolutional Neural Network is used in the proposed model to estimate the congestion condition of this road network in the near future. CNN makes use of the threshold value based on node instead of a network-wide threshold.

The CNN network includes a convolution layer, a pooling layer, and a fully connected layer at the end to interpret the features from the previous layer and perform the classification process. The convolution layer does the task of producing a feature map by applying the convolution operation with the input and kernels. The pooling layer of CNN helps to reduce the dimensionality. CNN receives data in the form of a sequence of predicted traffic flow values (time range of 5 to 30 minutes), timestamp information such as time and day, and a threshold value based on the average traffic flow and congestion level indicator of the station

road network. The model is programmed to take into account the flow from upstream and downstream stations that will cause congestion at a specific station. CNN is well-suited for extracting spatial information; it will learn the spatial relationship in the data and accurately forecast the level of congestion.

Algorithm: Traffic Congestion prediction algorithm and Fine-tuning hyperparameters

Input:

Traffic flow values f
Weather parameters W

Initialization of Parameters:

Learning rate, Number of LSTM Autoencoder layer, Number of neurons, Batch size, No of epochs(n), Optimizer, and Loss functions.
Window Size: 12
Step size:5min to 30 min (1 to 6)

Output: Optimized model (Stacked LSTM AE-BiLSTM-CNN) and learnable parameters

1. Design a network model with 2 LSTM Autoencoder stacked networks.
2. Extract features from W and calculate the latent vector W_{latent} using (4)
3. Decode W_{latent} using (5) by the decoder.
4. Calculate the reconstruction loss using (6) and select the encoded W_{latent} features when it is low.
5. Design a Bidirectional LSTM deep neural network model with 2 bidirectional LSTM, 2 LSTM layers with 64,64,32 and 32 neurons.
6. Input the combined W_{latent} and traffic flow(f) values to the model.
7. For i =1 to n do:
    a. Train the model using (7) & (8)
    b. Estimate the error between the actual value and the predicted value.
    c. Use backpropagation and optimize the parameters
8. Calculate $\hat{y}$, the future traffic density value using (9) for the upcoming 5 to 30 minutes (step size - 1 to 6).
9. Apply forward chaining cross-validation to improve the model.
10. Design a CNN model with two convolution layers and a fully connected layer with a softmax activation function.
11. Input the CNN model with traffic flow, average traffic flow value, time, day of the week, and congestion level.
12. Train the CNN model to predict the congestion status.
13. Predict the congestion level (CS) using CNN.

4 Experimental Setup

The experimental setup, datasets used, and hyperparameters with their selection criteria in the proposed model are all shown in this section. It also outlines the performance metrics that were used to validate the model. The tests are performed on an NVIDIA GPU-powered Dell Workstation of Dell Precision Tower 7920, 2xIntel Xeon Gold 5118, 12 core processor, Intel C621 Chipset, 256 GB 2666 MHz DDR4 RAM, and 2xNVIDIA Quadro GV100 + NVLink which is available in Research Lab, Ramco Institute of Technology, Rajapalayam. The prediction model was created in Python using Keras and Tensorflow frameworks.

4.1 Dataset

4.1.1 Traffic Data

The data used for the experimental purpose is collected from the Caltrans performance measurement system (PeMS)[34]. It is a California-based traffic data collection system that offers real-time traffic data, including date and time stamps, traffic flow per lane, and aggregated traffic flow, speed, and occupancy for analysis. Depending on the needs, the granularity can also be set to 5 minutes or hourly or weekly, or even monthly. The dataset contains traffic data collected every 30 seconds from the 15,000 detectors installed throughout the city. The data was aggregated into a 5-minute interval by PeMS. The proposed work used the traffic flow value from the PeMS dataset.

The data from the freeway I-405-Northbound of Orange county is taken for this research work which is shown in **Fig. 4**. The traffic data from the PeMS dataset for the 30 stations on the freeway is considered, in which the data from 08/01/2018 to 09/30/2018 is used for training purposes and the data from 10/01/2018 to 10/10/2018 is used as a testing data set. Each day has 288 data points with a 5-minute time interval because the traffic flow value is updated every 5-minute interval. The number of vehicles traveling in multiple lanes during the 5-minute interval is recorded.

Fig. 4. Freeway I-405-Northbound

4.1.2 Weather Data

The weather data for the location of Santa Anna, Irvine is collected for 5 minutes' intervals and it contains parameters such as Temperature, Dew Point, Humidity, Wind, Wind Speed, Pressure, precipitation, and weather conditions. The weather data from 08/01/2018 to 09/30/2018 is used for training the model and the data from 10/01/2018 to 10/08/2018 is used to validate the model's performance and effectiveness.

4.2 Data Preprocessing

The data quality plays an important influence in the deep learning model's success. As a result, before training the model, the data should be cleaned and prepared. Imputing missing values and feature scaling are two of the data preprocessing procedures employed in this proposed model design. In the dataset, feature scaling, also known as normalization, is used to translate the attributes into a single numerical scale, ensuring that the outcome is unaffected. The min-max normalization is used to get the data into the [0,1] range. The weather condition field, which comprises categorical data, is converted to a numerical representation using one-hot

code transformation. The min-max normalization applied is given in (10).

$$F' = \frac{F - \min(F)}{\max(F) - \min(F)} \tag{10}$$

Where F represents the old value of the data attribute and F' denotes the new value. Min(F) and max(F) indicates the minimum and maximum value of the attribute F.

4.3 Feature extraction

Feature extraction is an important task that helps to reduce the dimensionality of the data and thus reduces the computational complexity for processing. The stacked LSTM-based AE for extracting the features from the weather data W is used in this model. The SAE model is designed with two LSTM autoencoder stacked together which have the minimum error values. Hence the encoder output is taken as the latent vector (W_{latent}) and used as input to the prediction model in the next stage. The weather features extracted by this model reduce the processing time and also improve the accuracy of forecasting traffic flow.

4.4 Model design and Hyperparameter tuning

4.4.1 Model design

The traffic flow prediction phase of the proposed work is designed with a bidirectional LSTM model. The feature vector W_{latent} extracted from the SLSTM AE model encoder and the normalized traffic features are given as input to the traffic flow prediction model. BiLSTM model with 4 layer is trained to predict the traffic flow. The output of the hidden layer in both directions is calculated to produce the output of the final layer. The weight and bias values initialized randomly are tuned based on the backpropagation method with the RMSProp (Root Mean Squared Propagation) optimization techniques. The final layer will predict the traffic flow for the next 5 to 30-minute intervals. The BiLSTM model achieves the minimum MAE and MSE and better accuracy compared with other models.

Time and day of the week, traffic flow at intervals of 5 to 30 minutes, average traffic flow, and congestion level indicators are used to train the CNN model. Based on the load factor, the model will produce a congestion level indication. CNN model is built with two Convolution1D layers of 128,64 neurons, followed by a dense layer, and finally a fully connected layer to estimate the level of congestion. The softmax activation function is used in the output layer and categorical cross-entropy loss function is used to find the difference between predicted and actual values, and the adam optimizer is used to optimize the weights and bias value in the network. The average of the predicted value (Y_i) at a particular station for the next 5 to 30-minute interval is given as input to the trained CNN model and short-term congestion status is obtained.

4.4.2 Hyperparameter tuning

Hyperparameter tuning is needed to avoid the model to be overfitted. By altering the number of layers and hidden units, adding dropout, or using regularizers, it was possible to get the model to learn the complex pattern in the training data as opposed to memorizing it.

In the proposed model, the individual hyperparameters are tuned one by one and the model performance is analyzed. The number of LSTM layers and hidden units per layer is decreased to 4 and 64,64,32,32 neurons in each layer respectively. The inclusion of regularizers like kernel, bias, and recurrent regularizers in the model penalizes the complicated model to lessen

overfitting and enhance performance. In the LSTM layers, the L1 and L2 regularization is used to change the weight's value. By adding an L1 penalty, which is an absolute value of the magnitude of the coefficient or weights, the size of the coefficients can be constrained in the L1 regularization (11)

$$Coss\ function = Loss + \frac{\lambda}{2m} * \Sigma\|w\| \tag{11}$$

In L2 regularization, an L2 penalty which is basically a square of the magnitude of the coefficient of weights is added and it is represented in (12).

$$Coss\ function = Loss + \frac{\lambda}{2m} * \Sigma\|w\|^2 \tag{12}$$

The kernel regularizer value is set as 0.01, the recurrent regularizer is set as 0.02 and the bias regularizer is set as 0.01. The dropout rate of 0.2 is given for the LSTM layers.

4.4.3. Forward chaining cross-validation

The forward chaining cross-validation is a suitable method for time series forecasting model evaluation. It maintains the temporal dependency in the traffic data while creating the data folds for measuring performance. In the proposed work, the entire traffic data is separated into five folds, validating the model on each fold, and averaging the performance metrics, forward chaining cross-validation is used to increase the model's efficiency and performance. **Fig. 5** illustrates how the data is separated and how the model is being trained. The model was trained using the first fold of data and tested using the second fold during the first iteration. The first two folds in the next iteration are used for training, and the third fold is utilized for testing. This process does not violate the temporal dependency present in the traffic data. Similar steps are taken for each fold, and for each iteration, the average performance metrics such as MAE, RMSE are calculated. This outcome value demonstrates the model's increased accuracy.

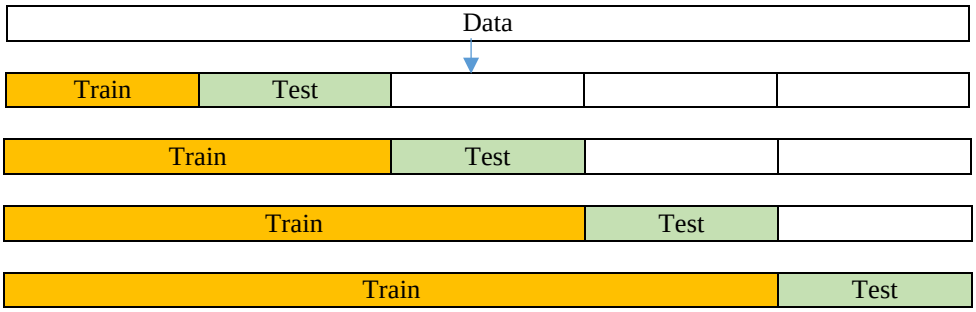


Fig. 5. Forward chaining cross-validation

4.5 Evaluation Metrics

The flow prediction model can be evaluated using the following performance metrics: Mean Square Error(MSE), Root Mean Square Error(RMSE), Mean Absolute Error(MAE), and Mean Absolute Percentage Error (MAPE). It can be calculated using (13) - (16). These metrics reveal the difference between the actual and predicted values. MAE is defined as the average difference between the original and predicted value. The average squared error between the original and anticipated value is known as the MSE. RMSE is the square root of the mean square error value. The accuracy of the regression model is determined by the value of R^2 which is defined as the ratio of variance explained by the model to the total variance.

$$MSE = \frac{1}{n}\sum(y - y')^2 \tag{13}$$

$$RMSE = \sqrt{\frac{1}{n}\sum(y - y')^2} \tag{14}$$

$$MAE = \frac{1}{n}\sum|y - y'| \tag{15}$$

$$MAPE = \frac{100\%}{n}\sum\left|\frac{y-y'}{y'}\right| \tag{16}$$

The congestion detection can be measured with the following performance metrics given in (17)-(19). Accuracy is the proportion of correct predictions. Precision is the proportion of true positives among all the values predicted as positive and recall or Sensitivity is the true positive rate. The classifier performance can be compared using the F1-Score which is the harmonic mean of precision and recall given by (20).

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \tag{17}$$

$$Precision = \frac{(TP)}{(TP+FP)} \tag{18}$$

$$Recall = \frac{(TP)}{(TP+FN)} \tag{19}$$

$$F1Score = \frac{(2*Precision*Recall)}{(Precision+Recall)} \tag{20}$$

where TP, TN, FP, & FN denotes the True Positive, True Negative, False Positive, and False Negative respectively. In the case of an imbalanced dataset, the accuracy measured does not reflect the correctness of the model. In such a case, Balanced Accuracy(BAC) can be measured which reflects the model performance. It is calculated by taking the mean of the true positive rate of the individual classes in multi-class classification.

5. Experimental Results

The traffic flow in each lane and aggregated flow values of the consecutive 30 stations are taken for processing. **Fig. 6** shows the average traffic flow in one week. From this, there is an inference that weekdays experience higher traffic volumes than weekends.

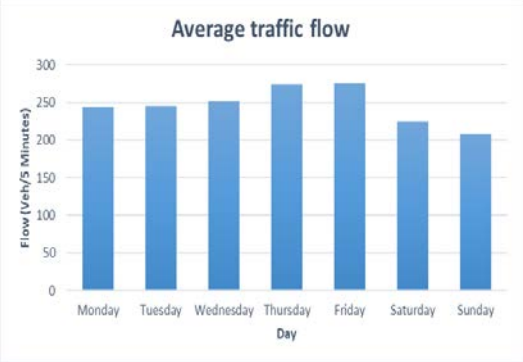


Fig. 6. Average traffic flow in a week

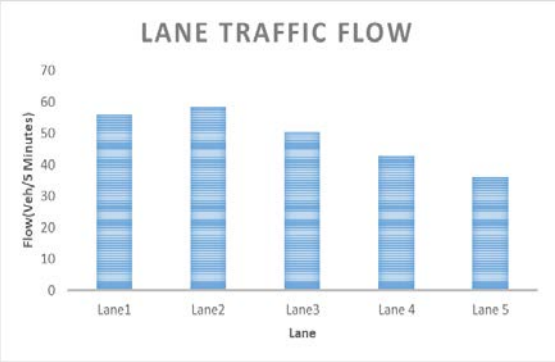


Fig. 7. Average Lane traffic flow

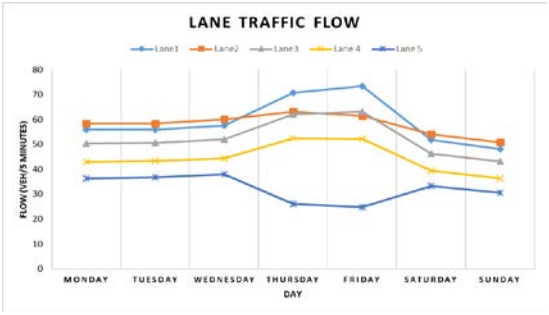


Fig. 8. Lane occupancy for a week

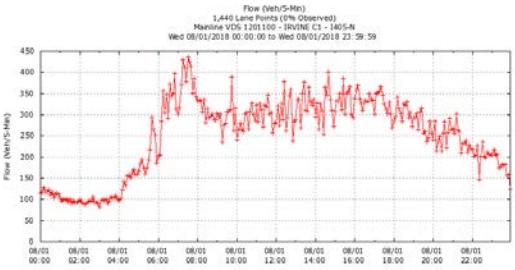


Fig. 9. Traffic flow value for a day

Fig. 7 shows the typical lane-by-lane average traffic flow for a single day. **Fig. 8** depicts how each lane was used over the course of a week. This shows that lanes 1 and 2 are frequently used while lane 5 is rarely used. **Fig. 9** the traffic flow value of a single day. According to this, the morning hours between 06 and 09 am the daily traffic flow value reaches a peak.

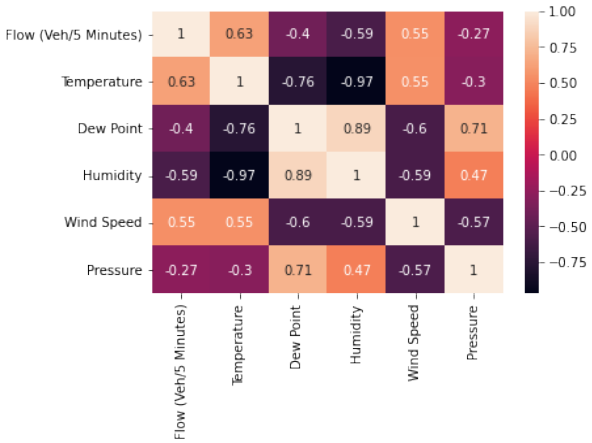


Fig. 10. Correlation between traffic flow and weather features

The correlation between the weather characteristics and the traffic flow value is calculated and shown in **Fig. 10**. It depicts that there is a strong relationship between traffic volume and temperature. The number of motorists on the road decreases as the temperature rises. The correlation between feature pressure and traffic flow is also not very strong. Temperature is one of the key characteristics of the weather that can be determined from this association, and it is given top attention while encoding. In **Fig. 11**, represents the actual traffic flow vs the predicted traffic flow value of the proposed work. The actual data (number of vehicles) for a week from October 1, 2018, to October 07, 2018 are depicted as a blue line, and the predicted values of the proposed model are depicted as an orange line. **Fig. 12** plots the prediction values from the baseline models, including LSTM [35], GRU[35], Stacked LSTM[36], and Stacked GRU[37] against the actual data. The baseline models are implemented using Tensorflow and Keras framework.

The baseline model details are given below:

1. LSTM- Long Short-Term Memory Network- A single LSTM layer with 64 neurons.
2. GRU- Gated Recurrent Unit- A variant of LSTM with a simpler structure- A single GRU layer with 64 neurons.
3. Stacked LSTM: Stacking of 4 LSTM layers with neurons of 128,64,64 and 64 respectively.

4. Stacked GRU: Stacking of 2 GRU layers with 64 neurons in each layer.

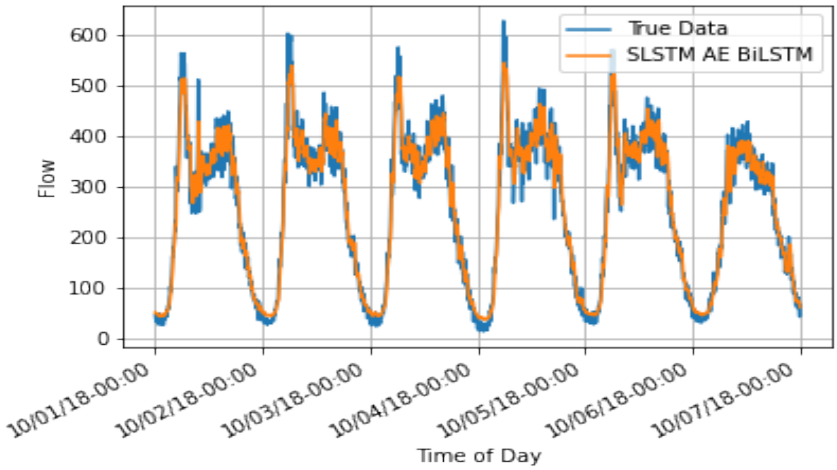


Fig. 11. Actual data vs Predicted data

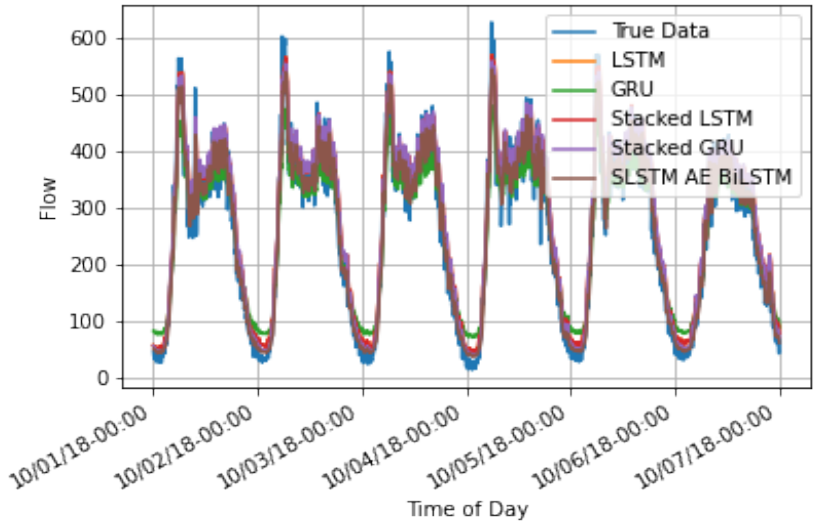


Fig. 12. Comparison of Prediction Results with Other Models

4.5.1. Evaluated Hyperparameters

In this work, the following hyperparameters are evaluated:
Optimizer: The neural network's objective function must be minimized, which is accomplished by the optimizer. When compared to stochastic gradient descent (SGD), the chosen optimizer, RMSProp, demonstrated superior efficiency and effectiveness for large data sets.
Dropout: Dropout is a prominent strategy for dealing with overfitting in neural networks. In the proposed model, the dropout rate of 0.1 to 0.2 for the LSTM layers is applied and evaluated the performance of the model.

Design a deep model with a number of recurrent neurons: Designed and evaluated the model with the number of 1,2 and 4 stacked BiLSTM layers. The number of neurons is decided from the choice of {24,32,64,128} and chosen as a 4-layer model with each layer having a neuron of 64,64,32 and 32 respectively.
Regularizers: On each LSTM unit, the kernel regularizer regularizes the input connections. The bias regularizer can be used to weight regularize the bias connection within the LSTM nodes. The recurrent regularizer regularizes the recurrent connections on each LSTM unit. For the kernel regularizer, the proposed model utilized L1 regularization with a value of 0.01; for the bias and recurrent regularizers, used L2 regularization with values of 0.01 and 0.02, respectively.

The model performance in terms of loss value is shown in **Fig. 13** before and after applying the parameter tuning. The performance metrics of the proposed model with and without hyperparameter tuning are shown in **Table 2**.

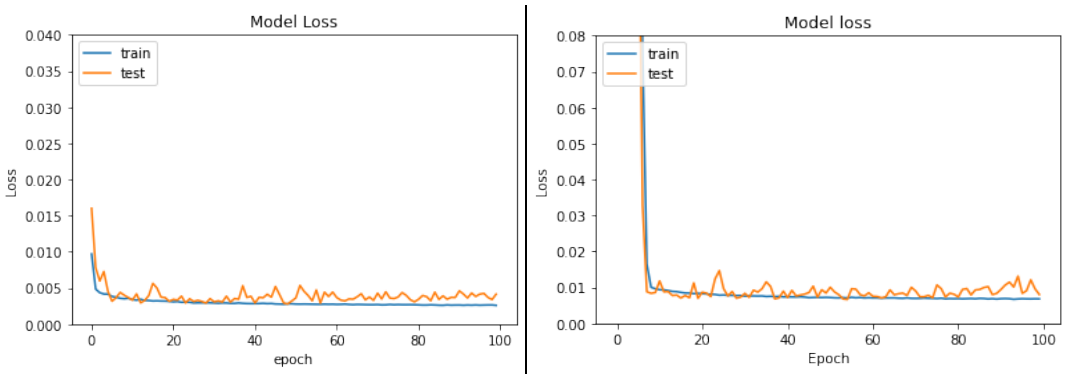


Fig. 13. Loss value before and after hyperparameter tuning.

Table 2. Comparison of different hyperparameters used in the proposed model

Hyperparameter	Before Tuning				After Tuning			
	Parameter Value	**MAE**	**RMSE**	**R^2 Score**	**Parameter Value**	**MAE**	**RMSE**	**R^2 Score**
Kernal_regularizer	**L1=0.0**	37.36	46.73	86.98	**L1=0.01**	33.66	45.63	90.27
Bias_regularizer	**L2=0.0**	33.6	45.63	90.27	**L2=0.01**	26.91	38.07	91.37
Recurrent_regularizer	**L2=0.01**	35.97	46.71	86.12	**L2=0.02**	27.48	39.26	91.05
Dropout	**0.1**	41.55	53.90	81.24	***0.2***	40.74	53.23	83.20
Optimizer	**SGD**	33.35	43.36	88.53	***RMSProp***	30.25	41.88	89.54
No. of Layers	**2**	*41.83*	*57.143*	*84.75*	***4***	37.49	50.97	87.86

Fine-tuning each hyperparameter leads to an improvement in model performance by lowering the error values can be seen in **Table 2**. The LSTM layer does not perform well when dropout is applied. Hence, the drop-out was not used in the LSTM layer. The regularizers were adjusted to prevent the model from overfitting. The set of values that generate better performance metrics is determined after the combination of the various hyperparameters is examined. Rolling cross-validation is used to evaluate this model using optimized hyperparameters. **Table 3** includes the hyperparameter values and the final performance of the fine-tuned model.

Table 3. Hyperparameter Value

Hyperparameters	Values Assigned	Resultant Performance Metrics
Number of layers	Bidirectional LSTM-2 LSTM-2	**MAE- 26.91 RMSE- 38.07 R^2 Score-91.37%**
Number of neurons in each layer	64,64,32,32	
No. of Epochs	100	
Learning rate	0.02	
Regularizer		
Kernal Regularizer	0.01	
Bias Regularizer	0.01	
Recurrent_regularizer	0.02	
Optimizer	RMSProp	
Batch size	100	
Loss function	MSE	
Activation function	relu	

The proposed model is compared with the baseline deep learning architecture LSTM, GRU, Stacked LSTM, and Stacked GRU in terms of performance measures such as MAE, RMSE, and R^2 values respectively. The evaluation measures are given in **Table 4** and **Fig. 14**. From the results, it is observed that the error rate of the proposed model is low compared with others in terms of MAE and RSME of 26.91 and 38.07 respectively. The proposed model uses the traffic data along with encoded meteorological data, so there is a reduction in the error rate. This leads to the efficient utilization of our model for the traffic flow prediction task.

Table 4. Comparison of MAE, RMSE, and R^2 values

Model	Traffic Data+ Weather data (Encoded)		
	MAE	**RMSE**	**R^2 Score**
LSTM	38.96	43.28	88.84
GRU	27.91	38.50	91.06
Stacked LSTM	40.61	48.97	85.71
Stacked GRU	27.51	38.61	91
SLSTM_AE-BiLSTM (Proposed)	**26.91**	**38.07**	**91.37**

The prediction results are measured in terms of mean absolute error, root mean squared error, and R^2 values for various time intervals from 5 min to 30 min which is depicted in **Table 5.** This leads to the conclusion that better results of short-term traffic flow prediction were achieved for the prediction horizon of 20-minute intervals.

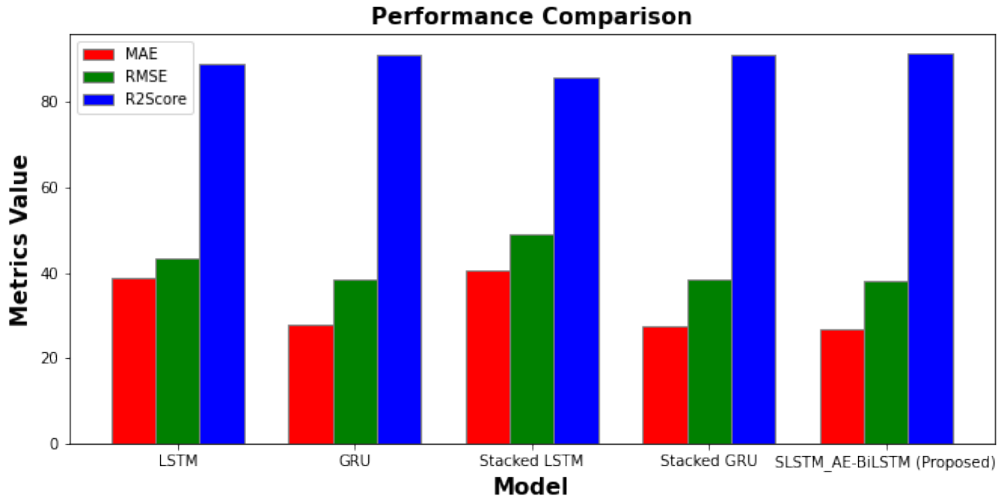


Fig. 14. Performance Comparison

Table 5. Comparison of proposed model output with a different time interval

Model: SLSTM_AE-BiLSTM			
Prediction interval	**MAE**	**RMSE**	**R^2 Score**
5 Min	***26.91***	***38.07***	***91.37***
10 Min	***26.95***	***38.97***	***91.87***
15 Min	***26.95***	***39.04***	***91.91***
20 Min	***29.96***	***39.34***	***92.05***
25 Min	30.20	42.73	92.22%
30 Min	30.30	41.67	92.82%

The forecasted traffic flow from the BiLSTM model is given as input for the various classifier models. The result for the various congestion detection algorithm in terms of accuracy, precision, recall, F1 Scores, and balanced accuracy scores are given in **Table 6**. When compared to other models, the Random forest model has a high accuracy rating of 94%, but its balanced accuracy rating is 64% which is quite poor. The CNN model generates a more accurate congestion detection model with a balanced accuracy score of 70% as shown in **Fig. 15**.

Table 6. Accuracy of Congestion classification algorithms based on single-step traffic flow prediction

Congestion detection Algorithms	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced Accuracy**
Logistic Regression	0.924	0.92	0.93	0.92	0.676
SVC	0.923	0.92	0.92	0.92	0.68
Decision tree	0.920	0.93	0.92	0.92	0.69
Random forest	0.94	0.91	0.94	0.92	0.64
CNN(Proposed)	**0.925**	**0.93**	**0.93**	**0.93**	**0.7**

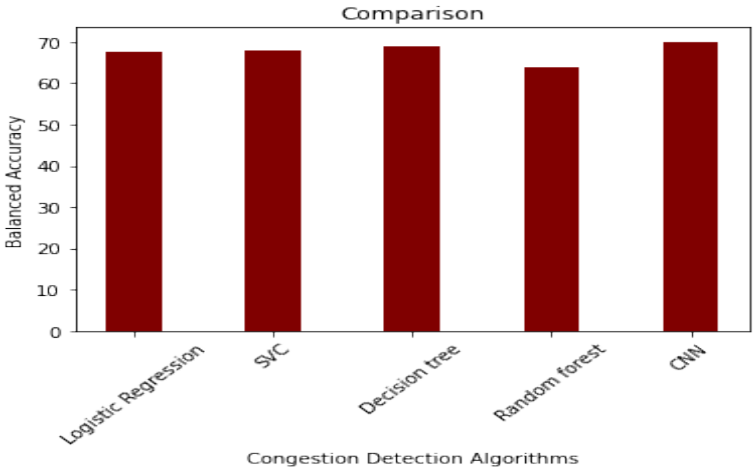


Fig. 15. Balanced Accuracy Comparison

The performance of CNN model based congestion state detection algorithm for various traffic flow prediction horizons (time range of 5 to 30 minutes) in terms of accuracy is given in **Table 7**. The accuracy of the congestion detection is increased over time since the CNN model will receive the average value of the projected traffic flow during the multi-step prediction process. The accuracy is 92.74% for the prediction horizon of 20 minutes which is high compared with the other prediction interval time range of 5 to 30 minutes. It is represented in **Fig. 16**.

Table 7. Accuracy of CNN model in various prediction interval

Congestion detection Algorithm	**Traffic flow Prediction Interval**					
	5 Minutes	**10 Minutes**	**15 Minutes**	**20 Minutes**	**25 Minutes**	**30 Minutes**
CNN-Accuracy	0.925	0.926	0.9225	***0.9274***	0.9243	0.926

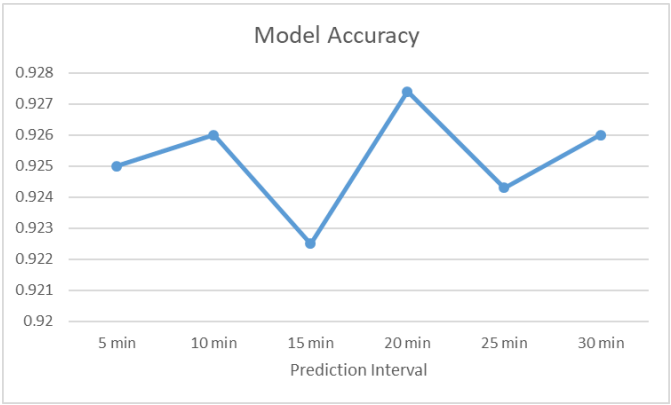


Fig. 16. Accuracy of CNN Model in various Prediction intervals

6. Conclusion and Future Work

A model for traffic congestion prediction based on short-term traffic flow predictions for periods of 5 to 30 minutes was designed in this work. It uses the data from the nearby stations to analyze the congestion propagation in a road network. It uses a Stacked LSTM-based Autoencoder for feature extraction from weather data, a Bidirectional LSTM for traffic flow prediction, and CNN for congestion prediction. This proposed model makes use of the multivariate input for prediction, and its congestion state prediction accuracy is 92.74% which is better than the other models. Forward chaining cross-validation is used to evaluate the model performance. The model performed well both during peak and off-peak hours with various traffic densities. The proposed model was developed to detect congestion using data from a single source, but in the future, it might be enhanced by incorporating data from additional sources. It can be focused on identifying irregular traffic congestion and how it spreads throughout the entire network.

Acknowledgement

We thank the management of Ramco Institute of Technology, Rajapalayam for having permitted us to carry out the research work in AI server of RIT Research Lab.

References

[1] T.Reed and J.Kidd, “Global traffic scorecard,” INRIX Research Altrincham, UK, 2019. [online] Available: https://trid.trb.org/view/1456836

[2] Supriya Raheja, S.Mohammad Obaidat, Balqies Sadoun, Sahil Malik, Anuj Rani, Manoj Kumar and Thompson Stephan, “Modeling and simulation of urban air quality with a 2-phase assessment technique,” *Simulation Modelling Practice and Theory*, vol.109, May 2021, Art. no. 102281. Article (CrossRef Link)

[3] A.Choudhary and S.Gokhale, “Evaluation of emission reduction benefits of traffic flow management and technology upgrade in a congested urban traffic corridor,” *Clean Technologies and Environmental Policy*, vol. 21, no. 2, pp. 257–273, Mar. 2019. Article (CrossRef Link)

[4] M. Gomathy Nayagam, K. Ramar, K. Venkatesh and S. P. Raja, “Moving object detection and tracking algorithm using hybrid decomposition parallel processing,” *Intelligent Automation & Soft Computing*, vol. 33, no. 3, pp. 1485–1499, Mar. 2022. Article (CrossRef Link)

[5] K.Ragavan, K. Venkatalakshmi, K. Vijayalakshmi, “Traffic video-based intelligent traffic control system for smart cities using modified ant colony optimizer,” *Computational Intelligence*, vol. 37, no. 1,pp. 538– 558, Feb. 2021. Article (CrossRef Link)

[6] T.Afrin and N.Yodo, “A survey of road traffic congestion measures towards a sustainable and resilient transportation system,” *Sustainability*, vol. 12, no.11, Jun. 2020, Art. no. 4660. Article (CrossRef Link)

[7] J.C.Falcocchio and H.S.Levinson, “Managing nonrecurring congestion,” in *Road Traffic Congestion: A Concise Guide, Springer Tracts on Transportation and Traffic*, Springer, Cham, vol.7, pp. 197–211, 2015. Article (CrossRef Link)

[8] Akarsh Aggarwal, Mohammed Alshehri, Manoj Kumar, Osama Alfarraj, Purushottam Sharma and Kamal Raj Pardasani, “Landslide data analysis using various time-series forecasting models,” *Computers & Electrical Engineering*, vol. 88, pp.106858, Dec. 2020. Article (CrossRef Link)

[9] S.Kumar, R.Viral and V.Deep, “Forecasting major impacts of COVID-19 pandemic on country-driven sectors: challenges, lessons, and future roadmap,” *Personal and Ubiquitous Computing*, Mar. 2021. Article (CrossRef Link)

[10] K.Lakshmanna, N.Subramani, Y. Alotaibi, S.Alghamdi, OI.Khalafand and AK.Nanda, "Improved Metaheuristic-Driven Energy-Aware Cluster-Based Routing Scheme for IoT-Assisted Wireless Sensor Networks," *Sustainability*, vol. 14, no. 13, Jun. 2022, Art. no. 7712. Article (CrossRef Link)
[11] Anuradha, Durairaj, Neelakandan Subramani, Osamah Ibrahim Khalaf, Youseef Alotaibi, Saleh Alghamdi and Manjula Rajagopal, "Chaotic Search-and-Rescue-Optimization-Based Multi-Hop Data Transmission Protocol for Underwater Wireless Sensor Networks," *Sensors*, vol. 22, no. 8, Apr. 2022, Art. no.2867. Article (CrossRef Link)
[12] I. Lana, J. Del Ser, M. Velez and E. I. Vlahogianni, "Road traffic forecasting: Recent advances and new challenges," *IEEE Intelligent Transportation Systems Magazine*, vol. 10, no. 2, pp. 93-109, Apr. 2018. Article (CrossRef Link)
[13] S.George and A.K.Santra, "Traffic prediction using multifaceted techniques: A survey," *Wireless Personal Communications*, vol. 115, pp. 1047-1106, Jul. 2020. Article (CrossRef Link)
[14] H.Yuan and G.Li, "A survey of traffic prediction: from spatio-temporal data to intelligent transportation," *Data Science Engineering*, vol. 6, pp. 63–85, Mar. 2021. Article (CrossRef Link)
[15] Mahmuda Akhtar and Sara Moridpour, "A review of traffic congestion prediction using artificial intelligence," *Journal of Advanced Transportation*, vol. 2021, Jan. 2021, Art. no. 8878011. Article (CrossRef Link)
[16] Nishant Kumar and Martin Raubal, "Applications of deep learning in congestion detection, prediction and alleviation: A survey," *Transportation Research Part C: Emerging Technologies*, vol. 133, pp. 103432, Dec. 2021. Article (CrossRef Link)
[17] Y. Zheng, Y. Li, C.M. Own, Z. Meng and M. Gao, "Real-time predication and navigation on traffic congestion model with equilibrium markov chain," *International Journal of Distributed Sensor Networks*, vol. 14, no. 4, Apr. 2018. Article (CrossRef Link)
[18] I. Okutani and Y. J. Stephanedes, "Dynamic prediction of traffic volume through kalman filtering theory," *Transportation Research Part B: Methodological*, vol. 18, no. 1, pp. 1–11, Feb. 1984. Article (CrossRef Link)
[19] B.L.Smith and M.J.Demetsky, "Short-term traffic flow prediction: neural network approach," *Transportation Research Record*, No. 1453, pp. 98–104, 1994. Article (CrossRef Link)
[20] Subramani Neelakandan, Perumal Santhosh, Kallimani Jagadish, Ulaganathan Sakthi, Bhargava Sanjay and Meckanizi Sangeetha, "Controlling energy aware clustering and multihop routing protocol for IoT assisted wireless sensor networks," *Concurrency and Computation: Practice and Experience*, vol. 34, no. 21, Jun. 2022. Article (CrossRef Link)
[21] S. Neelakandan, M.Prakash, S.Bhargava, K.Mohan, N.R.Robert and S.Upadhye, "Optimal Stacked Sparse Autoencoder Based Traffic Flow Prediction in Intelligent Transportation Systems" *Virtual and Augmented Reality for Automobile Industry: Innovation Vision and Applications*, pp. 111-127, Feb. 2022. Article (CrossRef Link)
[22] C.H. Chou, Y. Huang, C.Y. Huang and V.S. Tseng, "Long-term traffic time prediction using deep learning with integration of weather effect," in *Proc. of Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 123-135, Mar. 2019. Article (CrossRef Link)
[23] Xu Yang, Shixin Luo, Keyan Gao, Tingting Qiao and Xiaoya Chen, "Application of data science technologies in intelligent prediction of traffic congestion," *Journal of Advanced Transportation*, vol. 2019, Apr. 2019, Art. no. 2915369. Article (CrossRef Link)
[24] M.Chen, X.Yu and Y.Liu, "PCNN: Deep convolutional networks for short-term traffic congestion prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 11, pp. 3550-3559, Nov. 2018. Article (CrossRef Link)
[25] S.Zhang, Y.Yao, J.Hu, Y.Zhao, S.Li et al., "Deep autoencoder neural networks for short term traffic congestion prediction of transportation networks," *Sensors*, vol. 19, no. 10, pp. 2229, May 2019. Article (CrossRef Link)
[26] T.Sun,C.Yang,K.Han, W.Ma and F.Zhang, "Bidirectional Spatial–Temporal Network for Traffic Prediction with Multisource Data," *Transportation Research Record*, vol. 2674, no. 8, pp. 78-89, Aug. 2020. Article (CrossRef Link)

[27] B.Vijayalakshmi, K.Ramar, N.Jhanjhi, Sahil Verma,M.Kaliappan, K.Vijayalakshmi, S.Vimal and Uttam Ghosh, "An attention-based deep learning model for traffic flow prediction using spatiotemporal features towards sustainable smart city," *International Journal of Communication Systems*, vol.34, no.3, e4609, Feb. 2021. Article (CrossRef Link)

[28] X.Zhang and Q.Zhang, "Short-term traffic flow prediction based on LSTM-XGBoost combination model," *CMES-Computer Modeling in Engineering & Sciences*, vol.125, no.1, pp. 95–109,Sep. 2020. Article (CrossRef Link)

[29] A. Koesdwiady, R. Soua and F. Karray, "Improving traffic flow prediction with weather information in connected cars: A deep learning approach," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 12, pp. 9508–9517, Dec. 2016. Article (CrossRef Link)

[30] Yue Hou, Zhiyuan Deng, and Hanke Cui, "Short-Term Traffic Flow Prediction with Weather Conditions: Based on Deep Learning Algorithms and Data Fusion," *Complexity, Hindawi*, vol. 2021, Jan. 2021, Art. no. 6662959. Article (CrossRef Link)

[31] Z. Zheng, Y. Yang, J. Liu, H.-N. Dai, and Y. Zhang, "Deep and embedded learning approach for traffic flow prediction in urban informatics," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3927–3939, Oct. 2019. Article (CrossRef Link)

[32] W.Zhuang and Y.Cao, "Short-Term Traffic Flow Prediction Based on CNN-BILSTM with Multicomponent Information," *Applied Sciences*, vol. 12, no. 17, Aug. 2022, Art. no. 8714. Article (CrossRef Link)

[33] M. Schuster and K.K.Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673-2681, Nov. 1997. Article (CrossRef Link)

[34] Caltrans PEMS dataset: [Online]. Available: https://dot.ca.gov/programs/traffic-operations/mpr/pems-source

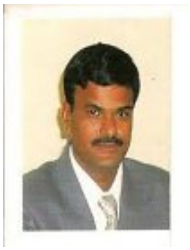
[35] R. Fu, Z. Zhang and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *Proc. of 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 324-328, Nov. 2016. Article (CrossRef Link)

[36] Yu, Rose, Li, Yaguang, Shahabi, Cyrus, Demiryurek, Ugur, Liu and Yan, "Deep learning: a generic approach for extreme condition traffic forecasting," in *Proc. of SIAM*, pp. 777–785, 2017. Article(CrossRef Link)

[37] M. Xia, H. Shao, X. Ma and C. W. de Silva, "A Stacked GRU-RNN-Based Approach for Predicting Renewable Energy and Electricity Load for Smart Grid Operation," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 10, pp. 7050-7059, Oct. 2021. Article (CrossRef Link)

B.Vijayalakshmi, is working as an Assistant Professor(SG) in the Department of Computer Science and Engineering, Ramco Institute of Technology. She has completed her B.E & M.E Degree in Computer Science and Engineering from PSR Engineering College, Anna University. Her areas of interest are image processing, data science and deep learning. She has published 13 papers in journal & International conferences. She is a Life Member of ISTE.

Dr.S.Thanga Ramya, B.E, M.S(by Res), Ph.D, is working as a Professor & Head in the Department of Computer Science and Design, R.M.K. Engineering College, Chennai. She obtained her B.E(CSE) from Dr.Sivanthi Aditanar College of Engineering and M.S by Research (ICE) from Anna University, Chennai. She has obtained her Ph.D in Information and Communication Engineering from Anna University, Chennai, in 2017. She has been in the teaching profession for the past 20 years and published more than 30 papers in various International Journals and Conference. Her areas of interest include programming languages, database management, cloud computing and data mining. She has got Oracle java international certification, IBM RAD,TIVOLI and DB2 certification and obtained her PRP certification from Wipro. She is the life member of ISTE.



Dr.K.Ramar, currently working as a Principal in , R.M.K. College of Engineering and Technology, Chennai. He obtained his Bachelor's Degree in Electronics and Communication Engineering from Government College of Engineering, Tirunelveli and Post Graduation in Applied Electronics from PSG College of Technology, Coimbatore. He obtained his Doctoral Degree in Computer Science Engineering from Manonmaniam Sundaranar University, Tirunelveli in 2001. He has 34 years of teaching experience including 29 years of Research experience. His extensive research experience has produced 46 Ph.Ds so far and 5 Scholars are pursing Ph.D. He has published 132 research articles in various International Journals and 91 of them are Scopus Indexed.