# A Vision Transformer Approach for Traffic Congestion Prediction in Urban Areas

Kadiyala Ramana, *Member, IEEE,* Gautam Srivastava, *Senior Member, IEEE,* M.Rudra Kumar, Thippa Reddy Gadekallu, *Senior Member, IEEE,* Jerry Chun-Wei Lin, *Senior Member, IEEE,* Mamoun Alazab, *Senior Member, IEEE,* and Celestine Iwendi, *Senior Member, IEEE*

**Abstract**—Traffic problems continue to deteriorate because of the increasing population in urban areas that rely on many modes of transportation, the transportation infrastructure has achieved considerable strides in the last several decades. This has led to an increase in congestion control difficulties, which directly affect citizens through air pollution, fuel consumption, traffic law breaches, noise pollution, accidents, and loss of time. Traffic prediction is an essential aspect of an intelligent transportation system in smart cities because it helps reduce traffic congestion. This article aims to design and enforce a traffic prediction scheme that is efficient and accurate in forecasting traffic flow. Available traffic flow prediction methods are still unsuitable for real-world applications. This fact motivated us to work on a traffic flow forecasting issue using Vision Transformers (VTs). In this work, VTs were used in conjunction with Convolutional neural networks (CNNs) to predict traffic congestion in urban spaces on a city-wide scale. In our proposed architecture, a traffic image is fed to the CNN, which generates feature maps. These feature maps are then fed to the VT, which employs the dual techniques of tokenization and projection. Tokenization is used to convert features into tokens containing Vision information, which are then sent to projection, where they are transformed into feature maps and ultimately delivered to LSTM. The experimental results demonstrate that the vision transformer prediction method based on Spatio-temporal characteristics is an excellent way of predicting traffic flow, particularly during anomalous traffic situations. The proposed technology surpasses traditional methods in terms of precision, accuracy and recall and aids in energy conservation. Through rerouting, the proposed work will benefit travellers and reduce fuel use.

**Index Terms**—Vision Transformers, Deep Learning, Intelligent Transportation System, Long-Short-Term-Memory (LSTM), Traffic Congestion prediction.

## 1 Introduction

Development, urbanization, and private travel [1] have all had a direct effect on the growth of cities, construction, and the environment, which has led to more traffic. Also, travel times get longer and traffic patterns get worse, which can lead to traffic accidents [2], [3], [4]. As a consequence, traffic management studies are very significant in science. Congestion could be lessened by making transportation infrastructure more expensive or by putting in place practical traffic solutions, like letting people know ahead of time how bad the traffic will be at an upcoming location. The better estimates of traffic speed [5] and volume [6] based on time series are more useful than trend analysis, which looks for traffic networks that are always backed up [7], [8], [9], [10]. Among them, prediction of traffic jams make it easier for drivers to choose better routes and for traffic managers to respond more quickly to changes in the transportation network. Table 1 represents the Nomenclature used in this paper.

TABLE 1
Nomenclature

| Acronym | Definition |
|---|---|
| LSTM | Long Short Term Memory |
| CNN | Convolutional Neural Networks |
| VT | Vision Transformer |
| ANN | Artificial Neural Networks |
| NN | Neural Networks |
| GPS | Global Positioning System |
| SVM | Support Vector Machine |
| PSO | Particle swarm optimization |
| CEC | Constant Error Carousel |
| BPTT | Back Propagation Through Time |
| RTRL | Real Time Recurrent Learning |
| iGPT | Image Generative Pre-trained Transformer |
| DeiT | Data-efficient image Transformers |
| CSFD | Congestion State Fuzzy Division |
| TFP | Traffic Flow Prediction |

Identification of congestion in the traffic is gaining research interest day by day because traffic congestion could cause wastage of fuel and reduce transportation perfor-


- *Corresponding Author: Gautam Srivastava*
- *Kadiyala Ramana is with the Chaitanya Bharathi Institute of Technology, Hyderabad, India. E-mail: ramana.it01@gmail.com*
- *Gautam Srivastava is with Department of the Mathematics and Computer Science,Brandon University, Manitoba, Canada Research Centre for Interneural Computing, China Medical University, Taichung, Taiwan as well as with the Dept. of Computer Science and Math, Lebanese American University, Beirut 1102, Lebanon Email: srivastavag@brandonu.ca*
- *M.Rudra Kumar is with G.Pullaiah College of Engineering and Technology, Kurnool, India Email:mrudrakumar@gmail.com*
- *Thippa Reddy Gadekallu is with School of Information Technology and Engineering, Vellore Institute of Technology, India Email:thippareddy.g@vit.ac.in*
- *Jerry Chun-Wei Lin is with Western Norway University of Applied Sciences, Bergen, Norway. Email: jerrylin@ieee.org*
- *Mamoun Alazab is with Charles Darwin University, Australia. Email: alazab.m@ieee.org*
- *Celestine Iwendiis with School of Creative Technology. University of Bolton, United Kingdom. Email: celestine.iwendi@ieee.org*


*Manuscript received xxxxxxxxx*

mance and a lot of pollutants will be released into the environment. Detecting congestion could lead the researchers to develop a template for forecasting congestion in the traffic and also provide a casual decision on the route taken by the vehicle users with which networks associated with roads and public transportation will be enhanced. A lot of proactive strategies are developed for controlling the traffic and for vehicle users live route supervision will be delivered.

Early models are based on predicting only speed, length, and traffic flow on a single route, group of roads, or a restricted road network. Road network capability constrains these initiatives; both commuters and traffic authorities have some issues. Data from one fixed sensor mounted on each road or multiple vehicles operating on each route is used. Since installation, service, and maintenance are costly, and third-party access is difficult, this data is difficult to obtain. Recently, real-time traffic information provided by the Web services such as Bing Map [11], Google Traffic [12], Baidu Map [13] and Seoul Transportation Activity and Information Service (TOPIS) [14] has become available in the public. These services are not well-known, but are public, readily available, and provide traffic information for almost all cities. The only problem is the curse of dimensionality because the issue of prediction is a study of time series, which require several inputs.

To address these common issues, the LSTM recurrent neural network has demonstrated considerable success in the areas of identification [15], time series prediction and translation [16], [17]. LSTM, on the other hand, can be difficult to use and slow to process due to its two-dimensional input and output sequences. Convolutional Neural Networks (CNN) have also established a reputation in spatial learning, most notably in image comprehension, segmentation, and object detection. However, due to characteristics such as local networking, weight sharing, and pooling, CNN has difficulty processing high-resolution multidimensional data. Meanwhile, [18], [19] employs a convolutional encoder to convert the input image to a low-resolution spatial image and a convolutional decoder to restore the latent representation to its original size.

Computer vision stores vision as pixels. Pixel arrays, illegal deep learning processors for computer vision, interpret convolutions. Even if this convention produced good vision models, there are still drawbacks. a) Because each pixel is unique, picture categorization models should prioritize the foreground. Segmentation models should prioritize pedestrians above the sky, terrain, trees, etc. Convolutions treat all image patches equally, regardless of importance. Compute and represent become less valuable. b) Not all pictures show ideas. All natural images have low-level characteristics like corners and edges, thus utilizing low-level convolutional filters. It would be inefficient to employ high-level filters on all photos with high-level attributes, like ear shape. Photos of flowers, cars, sea creatures, and other objects don't reveal dog features. Rarely used filters take a lot of computing power. Convolutions don't link far-flung notions. Each completely convolutional filter works in a tiny area, yet semantic ideas interact over great distances. Previously, connecting ideas required larger kernels, deeper models, or additional procedures like dilated convolutions, global pooling, and non-local attention layers. But within the context of pixel convolution, these solutions at best help alleviate the problem by adding model and computational complexity to make up for convolution's faults.

To fix the real problem with the pixel-convolution paradigm, we introduce the Vision Transformer (VT), shown in Figure 1. This is a new way to represent and process high-level concepts in images. Our first thought is that it is fine to explain high-level ideas in an image with just a few words (or Vision tokens). Later in the network, we stop using the fixed-pixel-array representation and instead use spatial attention to turn the feature map into a small set of semantic tokens. To record interactions between tokens, we send these tokens to a self-attention module called a "transformer." This module is often used in natural language processing [19]. The computed Vision tokens can be used directly for image-level prediction tasks, or they can be re-projected in time to the function map for pixel-level prediction tasks.

### 1.1 Research Gap

When compared to Deep Learning techniques ANN fails to explore the most complex and deeper architectures and deep learning techniques can attain much better performances than the typical methods. Still, deep learning is mainly focusing on the forecast congestion prediction on a simple section of road. Some of the researchers used deep learning techniques to estimate and forecast traffic congestion in the whole transportation network. Most of the time these techniques will not consider spatial correlation but manage to consider temporal correlations at one location.

To overcome this problem, in this article a method is proposed which is based on an image that represents traffic in terms of an image and utilizes deep learning mechanisms like CNN to extract Spatio-temporal features from the image. A CNN is an effective means of extracting features from the image when compared to ANNs. CNN has the following characteristics while features are getting extracted

1) The output neurons of CNN's convolutional layers are locally connected, which means they are connected to neighbouring input neurons.
2) CNN introduces a new layer known as the pooling layer, which selects important features from its receptive zone and reduces the parameters.
3) Normally, completely connected layers are only used at the end of the process.

The contributions of this paper can be summarized as follows:

1) Using the proposed image-based approach and the deep learning architecture of CNNs, spatial dependencies, and network traffic's temporal evolutions are considered and implemented at the same time in predicting traffic-related problems.
2) Spatio-temporal characteristics of network traffic can be extracted automatically with high estimation accuracy when using a CNN.
3) CNN's utilization in the proposed system will help it to be used in the prediction of traffic speed in large-scale problems.

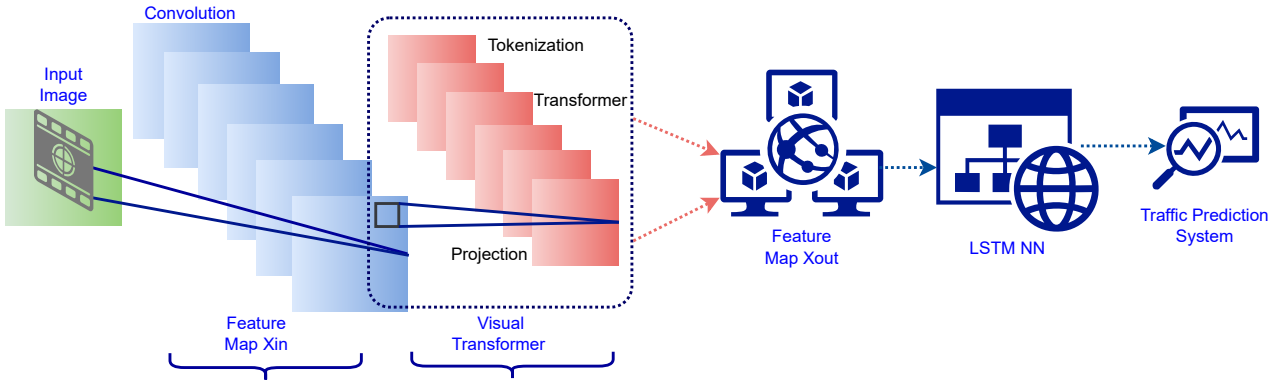


Fig. 1. Overall Framework of the Proposed Architecture

4) A Vision Transformer (VT) is used along with LSTM NN to predict traffic in smart cities.

The gap can be summarized as, CNN's convolutional layer output neurons being locally connected, which implies they are coupled to neighbouring input neurons. CNN introduces the pooling layer, which selects essential information from its receptive zone and decreases the parameters. Typically, completely connected layers are used only at the end of the process.

The contributions are summarized as the proposed image-based technique and the deep learning architecture of CNNs are used to consider and implement network traffic's temporal evolutions and spatial dependencies in traffic prediction issues at the same time. When using a CNN, the spatiotemporal properties of network traffic can be retrieved automatically with high estimation accuracy. Because it uses convolutional and pooling layers, the suggested method can be used to predict the speed of traffic on a large scale while still being easy to train. In smart cities, a Vision Transformer (VT) is utilized in conjunction with LSTM NN to anticipate traffic. It is reducing pollutant emissions; It is reducing traffic congestion so that people's quality of life is not jeopardized.

The remaining paper is framed as follows: Section 2 will present a survey of recent work from the literature. Then in Section 3 Materials and Methodology, we specify common techniques and approaches used to predict traffic, and in Section 4, we present results and related discussions. Finally, in Section 5, we conclude with final thoughts and proposals for directions that future work with our proposed system might take.

## 2 Related Work

In this section, we review various methods and techniques present in the literature that have contributed to predicting congestion in urban traffic.

In [20], the researcher proposed a technique for estimating the congestion in urban traffic using EMA with weights. Then [21] considered GPS location information and estimated the current traffic status using Spatio-temporal information. In [22], the researcher predicted the state of traffic by using information from GPS-equipped vehicle logs, averaging vehicle velocities and related information. Additionally, an improved strategy was introduced in [23] that utilized both time-fluctuating and space-changing data to anticipate metropolitan traffic states dependent on a versatile cubic surface traffic stream model. In this work [24], the researcher introduced a precise answer for forecasting traffic states by removing the Spatio-temporal average speed from an enormous number of GPS test vehicles. This technique depended on a bend-fitting and vehicle-following system. To improve the exactness of such assessments, researchers in [25] determined the mean speed at the street area level from multi-source traffic information and used this to gauge traffic states. In [26], researchers proposed a weighted means of dealing with gauge traffic state, utilizing GPS information by expanding loads of ongoing speed data.

In [27], the author introduced a cross-breed learning system that could suitably join assessment consequences of interstate traffic thickness states from numerous perceptible traffic stream models. Meanwhile, the researchers in this work [28] proposed dealing with gauge blood vessel travel time states by counting Bayesian and Expectation Maximisation calculations utilizing GPS test information. The previously mentioned techniques assessed traffic states by utilizing one explicit boundary like normal speed, travel time, or traffic density. Nonetheless, the vulnerability and intricacy of traffic states have not been adequately addressed by any of these techniques, even the authors in [29], who assessed traffic clog states utilizing a versatile neuro-fuzzy derivation framework.

In another work [30], the author introduced a gridlock assessment framework from video information utilizing physically tuned fuzzy rationale. Nonetheless, vehicle volume and speed were utilized in this strategy without considering the street space data. Elsewhere [31], the author delivered a tracking-based strategy utilizing Pareto's ideal choice hypothesis and extensive fuzzy judgment to appraise the traffic state. In [32], the author investigated the benefits of fuzzy deduction frameworks to assess the degree of street gridlock utilizing traffic density and speed data.

A fusion of SVM and the genetic algorithm was used in [33] to predict congestion in urban traffic. Elsewhere, the Genetic Algorithm was used for parameters optimization, and a support vector machine (SVM) was used to predict traffic based on online learning methodologies. Similarly, the author in [34] proposed a hybrid mechanism using SVM

for prediction and PSO to optimize parameters. Then in [35], the author proposed a traffic prediction mechanism using the Support Vector Regression technique to forecast traffic movement in smart cities using Spatio-temporal information. In [36], the author proposed a support vector Regression approach for traffic congestion forecasting by utilizing a kernel called Gaussian Radial Basis Function, which uses the fusion of simulated annealing and Genetic Algorithm for optimizing the input parameters. In [37], the author proposed a PSO algorithm based on Chaotic Cloud to optimize input parameters; here, too, the fusion of Gaussian Loss function and Support Vector Regression is used for the prediction of traffic in smart cities. Associated works present in the literature are shown in Table 2.

Despite the variety of methods or techniques present in the literature dealing with traffic prediction or forecasting traffic congestion, none have considered the full capacity of this phenomenon or the Spatio-temporal data of roads themselves. Also, almost all the techniques take only one metric to estimate the system's performance, which is a pitfall when we do not find systematic methods that will address stability, instantaneity, and Accuracy simultaneously. To address these issues, we present a novel technique for estimating and forecasting traffic congestion: one that considers all three metrics in its estimation of the system's performance.

The majority of previous research on predicting traffic conditions has concentrated on projecting future traffic flows at a certain site or travel times on a given road segment. Influenced by variables including inhabitants' movement, climate, and traffic control, the urban road traffic flow fluctuates continuously. Simultaneously, urban function design, geographical considerations, and social activities influence the traffic flow in different places and at different times along the same road. It possesses distinct spatiotemporal properties. Additionally, almost all the techniques have taken only one metric to estimate the performance of the system and there is a pitfall that we don't find systematic methods that will address stability, instantaneity, and accuracy simultaneously. To address this, we present a novel technique to estimate and forecast traffic congestion and consider all three metrics for the performance estimation of the system. Table 2 summarizes the related works.

TABLE 2
Associated Works in the Literature

| Ref. | Approaches Used | Limitations |
|---|---|---|
| [38] | Fusion of SVM and chaos wavelet analysis is used for the kernel selection for the Traffic congestion forecasting | Choice of appropriate kernel function for the practical problem; how to optimize parameters efficiently and effectively |
| [39] | Confirmed that appropriate selection of Support Vector Regression will improve the traffic congestion prediction accuracy | SVMs generally high performance may suggest SVR is likewise appropriate. However, SVRs depend on good parameter selection (PS). |
| [40] | Used SVR for the traffic congestion prediction | Selection of kernel function is a pivotal factor that also determines the performance of SVR |
| [41] | Particle filter technique is used for the traffic congestion prediction | The advantage of particle filter, in which each particle has a prediction value and an associated weight, cannot predict traffic state reliability information. |
| [42] | Artificial neural networks are used for traffic congestion prediction | Fails to predict methodological applications to urban arterials with more significant congestion levels; also fails to account for the effects of traffic signals. |
| [43] | A multivariate spatial-temporal auto regressive Model is proposed for the prediction of congestion and speed of the vehicles | The System slows down in heavy volumes and at high-speed modes |
| [44] | Fusion of Genetic Algorithm and Fuzzy rule-based hierarchical approach is proposed for traffic congestion prediction | Due to its automatic selection and ranking algorithm there is a possibility that lots of features with weight might be left out. |
| [45] | Learning based Statistical algorithms are used for the forecast of congestion in the traffic | Assessment of the statistical learning techniques for the prediction of traffic congestion is not performed |
| [46], [47] | Support vector machine-based least squares approach is used for the prediction of traffic in the urban areas and large-scale taxi traces are also used. | This method doesn't indicate the correlation among different road segments or the influence that certain road segments have on others. |

## 3 Materials and Methods

### 3.1 Feature Extraction Using CNN

CNN has exhibited substantial image-understanding learning capacity because of its unique image-feature extraction methods. CNN differs from conventional deep learning designs in two ways: (a) instead of connecting output neurons to all input neurons, output neurons are selectively connected to adjacent input neurons. Because each layer captures different aspects of the issue to be forecasted, they can eliminate image features efficiently [48]. With these two characteristics, CNN is upgraded to fit the sense of transportation. initially, The model's input images have one lane, which is evaluated by traffic speed, and pixel values range from 0-maximum traffic speed. Three channels will classify photos. Normalize model inputs to prevent training issues from model weights. Second, outputs vary. The model predicts traffic speeds on all road segments, while the classification model provides a labelled image. Third, abstract qualities are ambiguous. In transportation, abstract features produced from convolutional and pooling layers imply road speed connections. Abstract image characteristics for training may be shallow picture edges or deep object outlines. Abstract properties are useful for prediction queries. Fourth, model outputs affect training goals. Intelligent transportation outputs are continuous traffic rates, thus cost functions should be too. Image categorization uses cross-entropy cost functions. Fig. 2 shows the CNN framework for transportation: traffic function extraction, model input, and feature map.

From a transportation network, the images created with Spatio-temporal characteristics are the first model input. Let

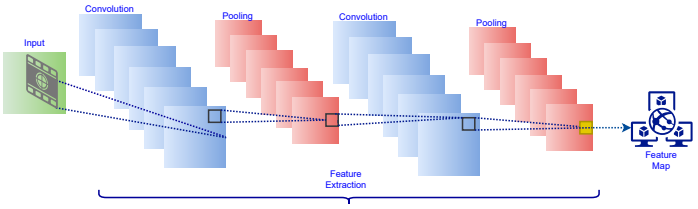


Fig. 2. Feature Extraction

F and P, respectively, be the lengths of input and output time intervals. In this case, the model feedback can be represented by the Equation 1:

$$X^j = [m_i, m_i + 1, m_i + p - 1], i \in [1, N - P - F + 1], \quad (1)$$

where $m_j$ is a column vector that defines, in a transportation network, the speed of the traffic within the one-time unit, where $N$ represents the length of time intervals, and $i$ represents the sample index.

Second, the CNN model's central component is the retrieving traffic attributes, which is a blend of convolutional and pooling layers. The term pool denotes the pooling method, and the depth of CNN is denoted by the L.$x_l^j$ represents input $l^{th}$ layer and $0_l^j$ represents the output of $l^{th}$ layer and ($W_l^j$,$b_l^j$) represents parameters of $l^{th}$, considering the various convolutional filters in the convolutional layer, j is the channel index. $c_l$ is the number of convolutional filters in the $l^{th}$ layer. The first convolutional and pooling layers' output can be written as in Equation 2.

$$0_1^j = pool\Big(\sigma\big(W_1^j x_1^j + b_1^j\big)\Big), j \in [1, c_1] \quad (2)$$

The Action function is referred to as $\sigma$, which will be addressed in the following section. The output of the convolutional and pooling layers in $l^{th}$(l $\neq$ 1,l=1L) can be written as in Equation 3.

$$0_l^j = pool\left(\sigma\Big(\sum_{k=1}^{c_{l-1}} \big(W_l^j x_l^j + b_l^j\big)\Big)\right), j \in [1, c_l] \quad (3)$$

The following are the characteristics of traffic function extraction: (a) Pooling and convolution are performed in two dimensions. This component will learn the Spatio-temporal connections of road parts; (b) Unlike the layers in Fig. 2 with only four convolutions or pooling filters, in the application number of layers is set to 100 seconds, since a CNN can learn hundreds of features; and (c) by using these layers CNN will convert the input model into deep features. In the prediction model, traffic function extraction output is fused with the features learned by the model into a vector containing final features that are high-level of the transportation network provided as input to the model. The dense vector is represented as in Equation 4.

$$0_L^{flatten} = flatten([0_L^1, o_L^2 o_L^j])' j = c_L, \quad (4)$$

where $L$ is the CNN deepness and flatten is the process mentioned above of fusing. Finally, a completely connected layer converts the vector into model outputs. As a consequence, the model output can be written as in Equation 5.

$$\begin{aligned}\widehat{y} &= W_f o_L^{flatten} + b_f \\ &= w_f(flatten(pool\Big(\sigma\Big(\sum_{k=1}^{c_{l-1}} \Big(W_L^j x_L^k + b_L^j\Big)\Big)\Big))) + b_f,\end{aligned} \quad (5)$$

where $W_f$ and $b_f$ are completely connected layer parameters. The predicted network-wide traffic rates are denoted by $\widehat{y}$.

It is necessary to remember that an activation mechanism triggers each layer before going on to the explicit layers. Below are listed some of the advantages of using the activation function: (a) the output of the activation function is limited to a scaled dataset that is used for training the model; and (b) the other layers are combined with the activation function to simulate complex nonlinear processes, allowing the CNN to handle the complexities of an Intelligent transportation network. The Relu function, which is used in this analysis, is defined as represented in Equation 6.

$$g_1(x) = \begin{cases} \chi & if x > 0 \\ 0 & Otherwise \end{cases} \quad (6)$$

Each output neuron is related to every input neuron in a traditional feedforward neural network, and the network is entirely connected, while convolutional layers are not. Over its input layer, then, the CNN applies convolutional filters to achieve local connections, through which local input neurons are linked to the output neurons. The various number of filters are added to the input, with the effects being combined in each layer. Hundreds of filters can extract hundreds of traffic features and One filter can remove one traffic feature from the input layer. Create high-level features of the traffic are created by traffic features merging. Now to create a more abstract range of traffic features and higher-level features combine the features of traffic extracted. This method validates the CNN's compositionality, ensuring that each filter generates high-level features from low-level features using a local path. When the convolutional filter $W_l^\gamma$ is added to the data, the effect is as represented in equation 7.

$$y_{conv} = \sum_{e=1}^{m} \sum_{f=1}^{n} \Big((W_l^Y)_{ef} d_{ef}\Big), \quad (7)$$

where $m$ and $n$ are the filter's two dimensions, $d_{ef}$ is the input matrix's data value at positions e and f, $(W_l^Y)_{ef}$ is the convolutional filter's coefficient at positions e and f, and $y_{conv}$ is the performance.

Since they only collect critical numbers from a single area, pooling layers are built to downsample and accumulate results. The pooling layers ensure that CNN is locally invariant, ensuring that regardless of feature transitions, rotations, or sizes, the CNN will still derive the same feature from the input [49]. Based on the above, pooling layers will minimize CNN's network size and classify the most popular input layer functionality. The pooling layer can be written as in equation 8, using the full operation as an example in Equation 8.

$$y_{pool} = max(d_{ef}), e \in [1 \dots p], f \in [1 \dots q], \quad (8)$$

where $y_{pool}$ is the pooling output, where p and q are two dimensions of pooling window size, and $d_{ef}$ is the data value of the input matrix at positions $e$ and $f$, respectively.

3D input is given to the model, where each dimension will give you information like input time interval, count of the road sections in Network 1, and another dimension will represent that the input picture has one channel.

## 3.2 LSTM

### 3.2.1 Standard LSTM

A memory block is considered, the most basic unit of the LSTM network hidden layers. A memory block is consisting of at least one memory cell and a couple of gating units that are adaptive and send input to all cells and output to all the cells in the memory block. A constant error carousel is at the core of every memory cell which is a self-connected, recurrent and linear unit and the activation of this core is called as state of the cell. This constant error carousel is responsible for addressing the error vanishing problem and it maintains error backflow consistently. The constant error carousel is protected from backward-flowing from the output gates and forward-flowing activation from the input gates. Noise will never enter the cell even when the gates are in a closed state. The memory block is shown in Fig. 3. By considering three inputs from sources and cell current state, cell state, $S_c$ is updated. Cell input is $net_c$,$net_{in}$ are input to the input gates and $net_{out}$ is the input to the output gates, standard LSTM is shown in Fig. 3.

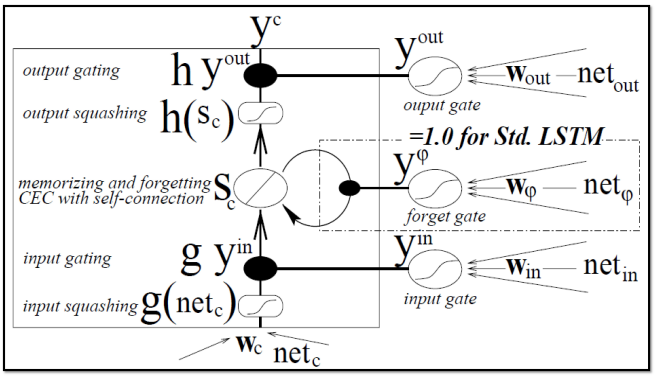


Fig. 3. Standard LSTM

Discrete-time steps $f = 1, 2, \dots$ are considered, A single stage includes upgrading all units (forward pass) and calculating all weights with error signals. Input gate activation $y^{in}$ and activation at the output gate is $y^{out}$ are computed as represented in Equations 9 and 10.

$$net_{out_j}(t) = \sum_m w_{out_j m} y^m(t-1)$$
$$y^{out_j}(t) = f_{out_j}\left(net_{out_j}(t)\right) \quad (9)$$
$$net_{in_j}(t) = \sum_m w_{in_j m} y^m(t-1)$$
$$y^{in_j}(t) = f_{in_j}\left(net_{in_j}(t)\right). \quad (10)$$

Throughout this article, $j$ refers to memory blocks; $v$ refers to memory cells in block $j$ and $c_i^v$ represents the $j^{th}$ memory block, and $v^{th}$ cell;$w_{Im}$ is the link load from unit $m$ to unit $l$. Index $m$ spans the entire source units, as defined by the blueprint of the network. For gates, $f$ is a logistic sigmoid with a range of $[0, 1]$. Net feedback to the cell itself is compressed by $g$, a set of $[-2, 2]$ centered logistic sigmoid functions. Input that is gated and squashed is applied to the preceding stage state $s_c$(f $-$ 1)(f $>$ 0) to determine the memory cell inside state $s_c$(t) as represented in Equation 11:

$$net_{c_i{}^U}(f) = \sum_m w_{c_i{}^U m} y^m(f-1)$$
$$s_{c_i{}^U}(f) = s_{c_i{}^U}(f-1) + y^{in_j}(f) g\Big(net_{c_i{}^U}(f)\Big) \quad (11)$$

With $s_{c_{i^0}}(0) = 0$. By squeezing the interior state $s_c$ with the help of function $h$ of output which is squashed, the cell output $y^c$ is determined and then multiplied by the output gate activation $y^{out}$ is represented in Equation 12.

$$y^{c_i{}^U}(f) = y^{out_j}(f) h\Big(s_{c_i{}^U}(f)\Big). \quad (12)$$

The centered sigmoid is $h$ within range $[-1, 1]$. Finally, assume that the coefficients for the $k$ output units are a layered network topology with a normal input layer, a hidden layer consisting of memory blocks, and a standard output layer as represented in Equation 13.

$$net_k(t) = \sum_m w_{km} y^m(t-1), y^k(t) = f_k\Big(net_k(t)\Big), \quad (13)$$

where $m$ extends to all components that serve the production units, the logistic sigmoid or scale $[0; 1]$, is used again as the squashing function $f_k$. Except for Equation 11, both equations refer to extended LSTM with forget gates.

### 3.2.2 Limitations of standard LSTM

LSTM stores data through random time delays and backdates error indications. LSTM's strengths can also bring problems. Cell states $sc$ tend to stretch linearly across a time sequence. If we provide a constant input stream, cell states will extend forever, saturating the output squashing function at h. The problem's origin suggests that cell states can be reset often, such as at the start of fresh input sequences. Saturation causes the derivative of h to vanish, eliminating incoming errors and causing the cell's output to match the activation of the output gate, rendering the cell unusable as a memory.

Long Short-Term Memory Neural Network is proposed to forecast traffic congestion. [50] suggested this to alleviate time series delays. This system's features make it useful for predicting urban traffic congestion and intelligent transportation. This system will have an input layer-1, a recurrent hidden layer-1, and an output layer-1. The memory block is used as the LSTM NN's hidden layer [51]. The memory block consists of self-connected memory cells and memorizes the temporal state. It also has gating units that regulate the flow of information into the block; the gates are multiplicative and adaptive.

Also, the memory block has one input and one output gate. Memory cells have an occasionally self-associated direct unit-Constant Error Carousel (CEC), whose operation determines cell state. The CEC opens and closes multiplying doors. LSTM NN can prevent disappearing errors by keeping organizational errors stable. A fail-to-remember door was added to the memory block to prevent an unbounded inward cell while managing persistent time arrangement. This approach allows memory squares to reset without external aid when data is outdated. Replaces CEC weight with fail-to-remember door activation. The above system is depicted in Fig. 4.

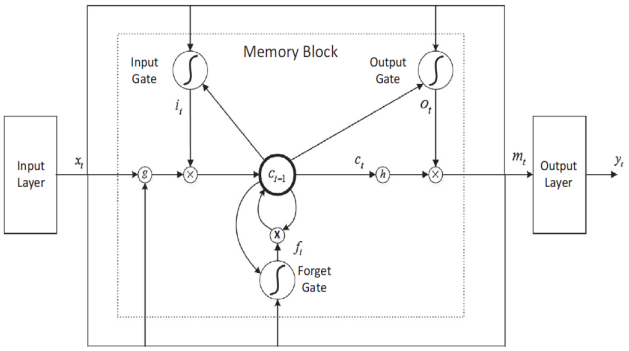


Fig. 4. architecture of LSTM Neural Network

Input for the above said model is given as $x = (x_1, x_2, \ldots, x_T)$ ,and the sequence of output is depicted as $y = (y_1, y_2, \ldots, y_T)$ , period of prediction is depicted with T. In the traffic congestion scenario, the past data is represented with x. The main motive of LSTM NN is to forecast the traffic congestion in the coming step based on the past data without telling the number of steps to be tracked back. To apply this approach, the predictions will be calculated with the equations specified from Equation 14 to Equation 19.

$$i_t = \sigma\Big(W_{ix}x_t + W_{im}m_t1 + W_{ic}c_t1 + b_i\Big) \tag{14}$$

$$f_c = \sigma\Big(W_{fx}x_c + W_{fm}m_c1 + W_{fc}c_c1 + b_f\Big) \tag{15}$$

$$c_c = f_cRejectc_c1 + i_cRejectg\Big(W_{cx}x_c + W_{cm}m_c1 + b_c\Big) \tag{16}$$

$$0_t = \sigma\Big(W_{ox}x_t + W_{0m}m_t1 + W_{oc}c_t + b_0) \tag{17}$$

$$m_c = 0_cRejecth(c_c) \tag{18}$$

$$y_c = W_{ym}m_t + b_y, \tag{19}$$

where Reject signifies product of two vectors with scalar values, and $\sigma$ $(\cdot)$ denotes the standard logistics Sigmoid function defined in Equation 20:

$$\sigma(x) = \frac{1}{\Big(1 + e^{\chi}\Big)} \tag{20}$$

The memory block is laid out in a dished box and comprises an information entryway, a yield door, and a fail-to-remember door, where the yields of three entryways are separately addressed as $i_r$,$0_r$,$f_r$. For each memory block cell activation vectors are denoted as $c_r$ and $m_r$. $W$ is the weight matrix and $b$ is the bias vector both are used for building a connection between the output, input, and memory block g(·) is a centered logistic sigmoid function with range $[-2, 2]$ as represented in Equation 21.

$$g(x) = \frac{4}{1 + e^{\chi}} - 2, \tag{21}$$

where h(·)is a centered logistic sigmoid function with range [-1,1] as represented in Equation 22.

$$h(x) = \frac{2}{1 + e^{\chi}} - 1 \tag{22}$$

Preparing LSTM NN depends on shortened Back Propagation Through Time (BPTT) and a changed rendition of Real-Time Recurrent Learning (RTRL) utilizing the angle plummet enhancement strategy [52]. The basic target work is to limit the number of square mistakes. Blunders are shortened when they show up at a memory cell yield, and afterward, they enter the memory cell's direct CEC, where mistakes can stream back everlastingly, and making mistakes stream outside the cell will in general rot dramatically [53]. This clarifies the motivation behind why LSTM NN has the capacity of handling discretionary delays for time arrangements with long reliance. Because of the broad numerical determinations, the point-by-point execution steps are not canvassed in this part. Fascinating readers may allude to [54] for more data.

### 3.3 Vision Transformer

In this segment, we survey the uses of transformer-based models in PC vision, including picture arrangement, high/mid-level vision, low-level vision, and video preparation. We likewise momentarily sum up the utilization of the self-consideration instrument and model pressure techniques for a proficient transformer.The works that purely use transformer for image classification include iGPT [55], VT [56], and DeiT [57].

The Vision Transformer is shown in Fig. 5. An input image is considered and the input image is given as input to the CNN which will generate several high-level features then a feature map will be given as input to the transformer which is responsible for creating an association between the Vision token which are generated by the process called Tokenization. Finally, these associated Vision tokens are used directly for forecasting or again generate a feature map for forecasting at the pixel level. Vision Transformer will be consisting of three components; they are tokenization and transformer and projections. Tokenization is responsible for creating tokens consisting of Vision information and the transformer's role is to create a semantic association between these tokens [19]. Finally, a projection unit is responsible for generating an augmented feature map. Comparable standards can be found in [58], [59], [60] however with one basic distinction: Previous techniques use many semantic ideas, though our VT utilizes as not many as 16 Vision tokens to accomplish predominant execution.

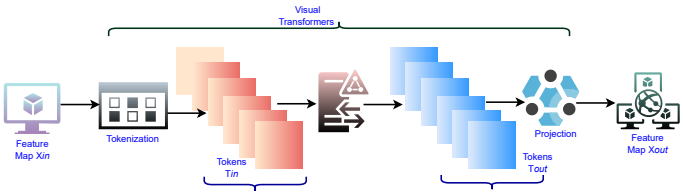


Fig. 5. Vision Transformers

### 3.3.1 Tokenization

A module called Tokenization is introduced which will take feature maps as input and generate tokens with Vision information. Typically feature map which is fed as input will be represented as $X \in \mathbb{R}^{HW \times C}$ and tokens with Vision information will be represented as $T \in \mathbb{R}^{L \times C}$ such that, $<<$ HW. We propose a repeated tokenization process on earlier phase tokens with Vision information. The idea here is to present layer tokens that will be generated by taking supervision from the early stage tokens $T_{in}$, because of this concept only this tokenization process is considered a recurring or repeated tokenization process. Typically we define in Equation 23.

$$W_R = T_{in} W_{T \to R}$$
$$T = SOFTMAX_{HW}\left(XW_R\right)^T X, \tag{23}$$

where $W_{T \to R} \in \mathbb{R}^{C \times C}$. Along these lines, the VT can gradually refine the arrangement of Vision tokens, moulded on previously-processed ideas. Practically speaking, we apply tokenization with intermittent nature beginning from the subsequent VT, since it requires tokens from a past VT. Fig. 6 Will represent tokenization process.

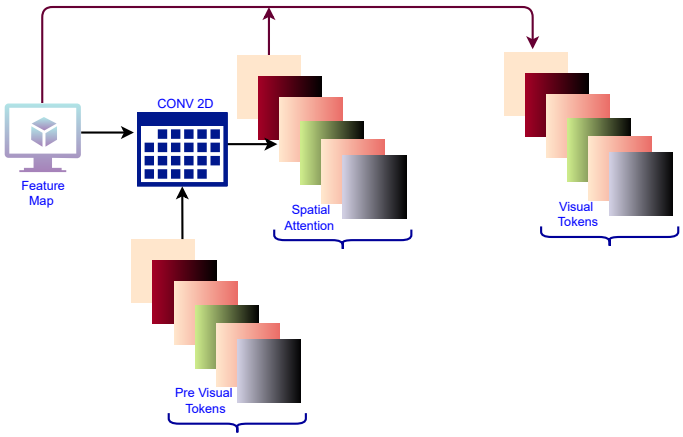


Fig. 6. Tokenization Process in a recurrent manner

### 3.3.2 Transformer

After tokenization, we at that point need to display associations between these Vision tokens. In [58], [59], [60] to create association convolutions based on graphs are utilized. Stable/consistent weights are used while creating association which means that each token while association is bound to some semantic concept. We implement transformers, whose design is based on input weights represented in Equations 24 and 25. We utilize a standard transformer with minor changes.

$$T'_{out} = T_{in} + SOFTMAX_L((T_{in}K)(T_{in}Q)^T)T_{in} \tag{24}$$

$$T_{out} = T'_{out} + \sigma(T'_{out}F_1)F_2, \tag{25}$$

where $T_{in}$,$T'_{out}$,$T_{out} \in \mathbb{R}^{L \times C}$ are the tokens with Vision information Different from graph convolution, in a transformer, weights between tokens are input based and computed as a key query product: $(T_{in}K)\,(T_{in}Q)^T \in \mathbb{R}^{L \times L}$. The transformer structure is shown in Fig. 7.

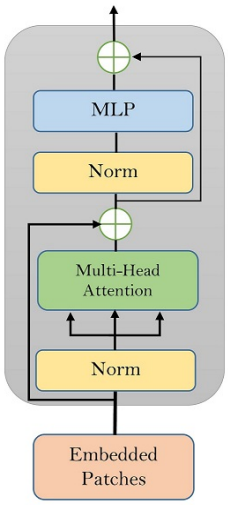


Fig. 7. Transformer Structure

### 3.3.3 Projections

Numerous vision errands require pixel-level subtleties, yet such subtleties are not safeguarded in Vision tokens. Consequently, we meld the transformer's yield with the element guide to refine the element guide's pixel-array portrayal as in Equation 26.

$$X_{out} = X_{in} + SOFTMAX_L\left(\left(X_{in}W_Q\right)\left(TW_K\right)^T\right)T \tag{26}$$

Where $X_{in}$,$X_{out} \in \mathbb{R}^{HW \times C}$ the information and yield are include map. $\left(X_{in}W_Q\right) \in \mathbb{R}^{HW \times C}$ is the question registered from the information include map $X_{in}$.$\left(X_{in}W_Q\right)_p \in \mathbb{R}^C$ encodes the data pixel-p requires from the tokens with Vision.

Performance Analysis of the proposed model in comparison with state of art techniques in terms of Recall is shown in Fig. 9.

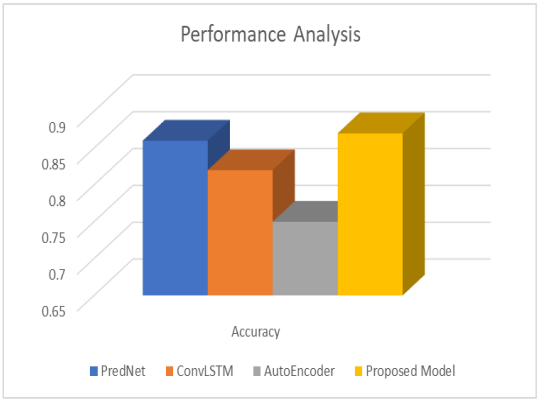


Fig. 8. Performance Analysis concerning Accuracy

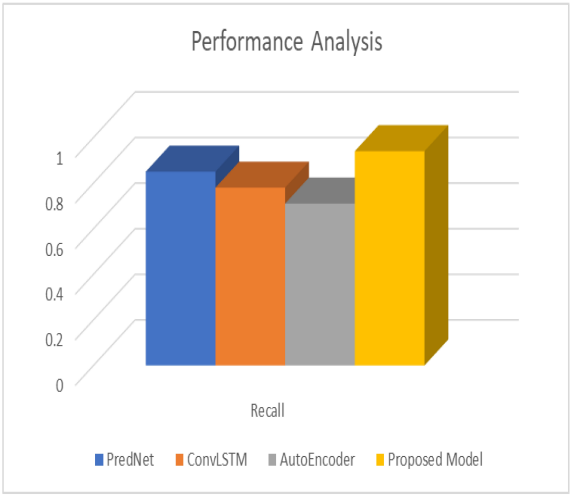


Fig. 9. Performance Analysis concerning Recall

### 3.4 Prediction of traffic flow:

By figuring out how bad traffic is, we hope to lower travel costs and stop it from getting worse. To reach this goal, a method for predicting congestion is used. This method includes predicting traffic flow and fuzzy division of the congestion state. Because of this, we run trials based on these two things.

In this part, we look at how well linear and RBF kernel functions work. When we look at how well these kernel functions work, we don't use any optimization techniques. As we did in the last section, we divide one day into five parts so we can get a more accurate picture of the results. The same indices are used for accuracy before morning peak (`accbefmor`), during morning peak (`accmor`), between morning peak and evening peak (`accbetmoev`), during evening peak (`acceve`), and after evening peak (`accafteve`). Table 3 shows the results of the congestion state forecast for different times. From Figures 10 and 11 and Table 3, we can see that linear and RBF kernels have about the same overall accuracy. This means that they can both be used to deal with congestion in the real world. The linear kernel also does better than the RBF kernel during the morning, evening, and after-evening peak times. It means that each kernel has its advantages when it comes to predicting congestion states. Because of this, we use the multi-kernel function to improve the accuracy of our predictions.

TABLE 3
Congestion State Prediction Performance of Kernel functions in different periods

| Metrics | Linear | RBF |
|---|---|---|
| accuracy | 77.55% | 77.55% |
| acc_befmor | 69.87% | 71.06% |
| acc_mor | 78.04% | 72.45% |
| acc_betmoev | 73.06% | 76.54% |
| acc_eve | 86.66% | 82.91% |
| acc_afteve | 87.45% | 86.27% |

## 4 Results and Discussion

In this section, we first experiment and evaluate the performance of the proposed traffic congestion prediction technique.

By forecasting the congestion present in the traffic in urban areas, we target to reduce the cost required for travel and avoid congestion creating situations from spreading further. To achieve this traffic congestion forecasting technique is implemented and used, which comprises the flow of the traffic forecast and the fuzzy splitting of the congestion state. This way experimentation is performed by keeping these 2 aspects in mind they are speed forecasting in the traffic and volume of the traffic results are discovered in this experimentation.

### 4.1 Performance indexes

From CSFD and TFP, performance indexes are considered and these concepts are explained in the following sections.

### 4.2 Traffic flow prediction indexes

Equations 27 to 30 show the accuracy indexes of the prediction and it is made up of root mean square error, mean square error, mean relative absolute error, and mean absolute error.

$$MAE = \frac{1}{N}\sum_{t=1}^{N}|P_{redict}(t) - R_{eal}(t)| \tag{27}$$

$$MRAE = \frac{1}{N}\sum_{t=1}^{N}\frac{|p_{redict}(t) - R_{eal}(t)|}{R_{eal}(t)} \tag{28}$$

$$MSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}\left(p_{redict}(t) - R_{eal}(t)\right)^2} \tag{29}$$

$$MSRE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}\left(\frac{p_{redict}(t) - R_{eal}(t)}{R_{eal}(t)}\right)^2} \tag{30}$$

where $P_{redict}$ denotes the predicted value and $R_{eal}$ denotes the real value.

To evaluate the proposed method, performance metrics such as Accuracy, Precision and Recall were used and represented in Equations 31 to 33.

$$\text{Accuracy} = \frac{\text{Detected Results}}{\text{Total no.of iterations}} \tag{31}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive+True Negative}} \tag{32}$$

$$\text{Recall} = \frac{\text{True Negative}}{\text{True Positive+True Negative}} \tag{33}$$

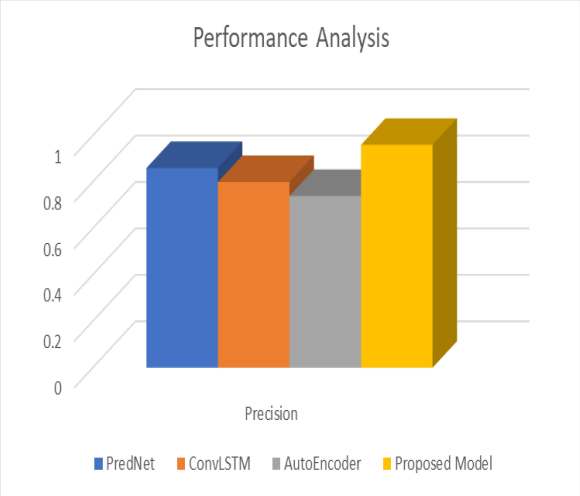


Fig. 10. Performance Analysis concerning Precision

The real-time performance indexes include the time for training model ($tr_{time}$) and traffic flow prediction ($pre_{time}$). The stability performance indexes are related to the process of the punish coefficient selection and the prediction accuracy.

### 4.3 Congestion state division indexes

Here, the Accuracy achieved by the model is given by the state of congestion predicted divided by the state of congestion in reality. The performance of the proposed model regarding traffic volume forecast and an average speed forecast is given below in Tables 4, 5, and 6, which reveal the performance of our proposed model as compared to others available in the literature.

TABLE 4
Performance of the Proposed Model

| Metrics | Traffic Volume Prediction | Average Speed prediction |
|---|---|---|
| maerr | 0.5219 | 0.1475 |
| marerr | 21.8798 | 4.8637 |
| mserr | 30.2642 | 6.7025 |
| msrerr | 1.3565 | 0.2253 |
| $tr_{time}$ | 0.1112s | 0.0821s |
| $pre_{time}$ | 0.0330s | 0.0085s |

The comparison between various models from the literature and our proposed model is shown in Table 6. Performance Analysis of the proposed model in comparison with

TABLE 5
Performance of the Proposed Model about Traffic Congestion forecasting

| Metrics | Proposed Model | $GA_{SVM_{RL}}$ | $PSO_{SVM_R}$ |
|---|---|---|---|
| Accuracy | 0.87 | 0.6725 | 0.7422 |
| Instantaneity | 0.621s | 6.094s | 0.701s |
| Stability | Yes | No | No |

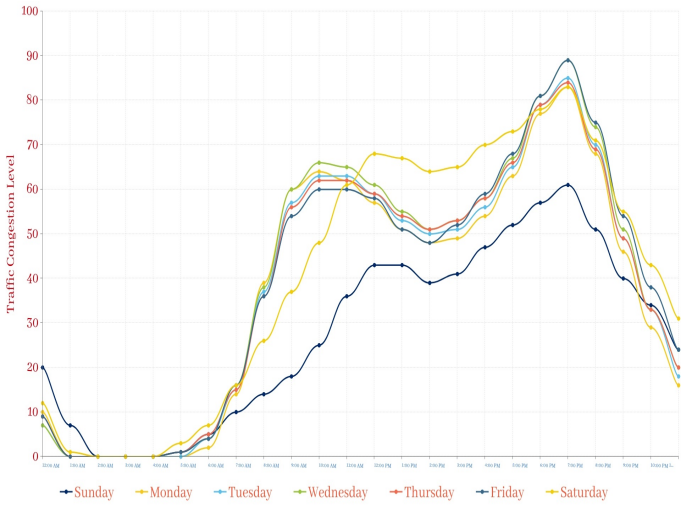


Fig. 11. Estimation of Traffic Congestion during a Week

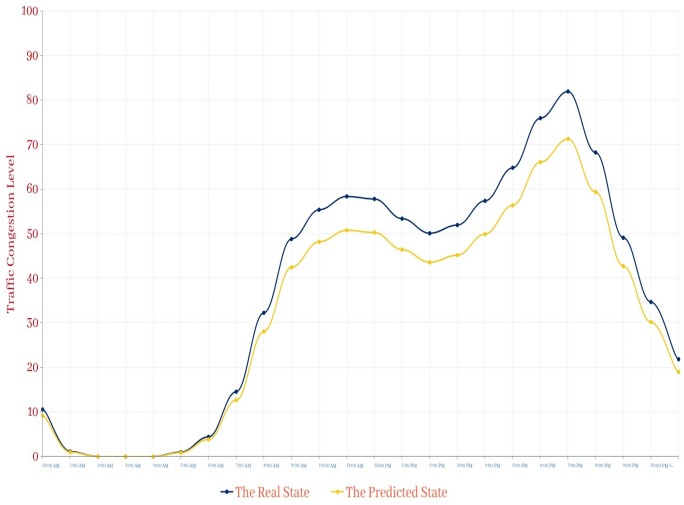


Fig. 12. Comparison between the real congestion and predicted congestion state

other techniques in terms of Accuracy is shown in Fig. 8. Performance Analysis of the proposed model in comparison with state of art techniques in terms of Precision is shown in Fig. 10.

### 4.4 Congestion Estimation

Even though the parameters for traffic flow have been collected, it is clear that traffic congestion can't be described correctly. This is because different roads have different sizes and can hold different amounts of traffic. Google Earth,

TABLE 6
Comparision between state-of-art work and the proposed model in terms of accuracy, recall and precision

| Prediction Model | Accuracy | Recall | Precision |
|---|---|---|---|
| PredNet | 0.86 | 0.85 | 0.86 |
| ConvLSTM | 0.82 | 0.78 | 0.80 |
| AutoEncoder | 0.75 | 0.71 | 0.74 |
| Proposed Model | 0.87 | 0.94 | 0.96 |

the length and number of lanes on a section of road, and the amount of traffic can all be used to figure out the traffic density. There are eight lanes in the 1.127-square-kilometre test area. Road saturation can also be figured out by how much and how much traffic there is. The traffic capacity comes from the Indian National highway capacity manual, which says that a multi-lane highway with a speed limit of 80 km/h can hold up to 1800 pcu/h of traffic per lane. In other words, the road under study can handle a maximum of 150 pcu/5 min/lane. Then, in our suggested method, the congested states are split up by road saturation, traffic density, and average traffic speed. As a result, $Wa = [0.43, 0.27, 0.3]$ and $Wb = [0.23, 0.17, 0.6]$ are given to the weight sets. Figure 11 and 12 depicts our method's traffic congestion estimation over a week (from Monday to Sunday). Fig. 11 depicts an estimation of the congestion of traffic in an urban area during a week. On commuter roads, weekday traffic is often significantly higher than weekend travel. (Weekend traffic is heavier in regions where recreation, tourism, or shopping predominate.) Figure 11 illustrates this variation dramatically for Detroit freeways. It also indicates that there is some variation between weekdays: Thursdays and Fridays are often the busiest days during this period. We have considered Fig. 12 shows the comparison between the predicted congestion state and the real-time congestion. In addition, our suggested method accurately predicts morning peak, evening peak, and after-evening peak traffic congestion.

## 5 Conclusion

In this research, a novel approach is proposed for the estimation and forecast of congestion in urban traffic. In this approach CNN are used where the image is given as input and low-level features are extracted and they will be further converted into high-level features and the fully connected layer used for the prediction is removed from CNN to overcome the problems with CNN and the feature map is given as input to the Vision Transformers, which will convert input feature map into the tokens with Vision information and these tokens are given as input to the transformer which is responsible for creating the association between the tokens and then these tokens with the association are further projected into feature map and this feature map is given as input to LSTM NN which is responsible for the prediction of traffic congestion. This approach is capable to overcome the problems with CNN and also delivers enhanced performance when compared to the state-of-art literature. In the future, various feature selection algorithms can be utilized in this approach which might enhance the performance of this technique and the computational efficiency in traffic congestion forecasting. This work has limitations, such as its inability to predict the speed of the vehicles after the congestion is reduced. Once congestion is reduced to limit $CO_2$ emission vehicles will go at extremely high speeds which is not safe for travellers. So this system fails to predict what is the possible speed of the vehicles which will limit $CO_2$ emission and simultaneously provide safety.

## References


[1] C. Onyeneke, C. Eguzouwa, and C. Mutabazi, "Modeling the effects of traffic congestion on economic activities-accidents, fatalities and casualties," *Biomedical Statistics and Informatics*, vol. 3, no. 2, pp. 7–14, 2018.

[2] C. Wang, M. A. Quddus, and S. G. Ison, "Impact of traffic congestion on road accidents: a spatial analysis of the m25 motorway in england," *Accident Analysis & Prevention*, vol. 41, no. 4, pp. 798–808, 2009.

[3] P. Hao, C. Wang, G. Wu, K. Boriboonsomsin, and M. Barth, "Evaluating the environmental impact of traffic congestion based on sparse mobile crowd-sourced data," in *2017 IEEE Conference on Technologies for Sustainability (SusTech)*. IEEE, 2017, pp. 1–6.

[4] S. Ye, "Research on urban road traffic congestion charging based on sustainable development," *Physics Procedia*, vol. 24, pp. 1567–1572, 2012.

[5] F. Rempe, G. Huber, and K. Bogenberger, "Spatio-temporal congestion patterns in urban traffic networks," *Transportation Research Procedia*, vol. 15, pp. 513–524, 2016.

[6] L. Xu, Y. Yue, and Q. Li, "Identifying urban traffic congestion pattern from historical floating car data," *Procedia-Social and Behavioral Sciences*, vol. 96, pp. 2084–2095, 2013.

[7] J. Park, D. Li, Y. L. Murphey, J. Kristinsson, R. McGee, M. Kuang, and T. Phillips, "Real time vehicle speed prediction using a neural network traffic model," in *The 2011 International Joint Conference on Neural Networks*. IEEE, 2011, pp. 2991–2996.

[8] X. Ma, Z. Dai, Z. He, and J. Ma, "J., y. wang, y. wang," learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, 2017.

[9] H. Chang, Y. Lee, B. Yoon, and S. Baek, "Dynamic near-term traffic flow prediction: system-oriented approach based on past experiences," *IET intelligent transport systems*, vol. 6, no. 3, pp. 292–305, 2012.

[10] S. Zhang, Y. Yao, J. Hu, Y. Zhao, S. Li, and J. Hu, "Deep autoencoder neural networks for short-term traffic congestion prediction of transportation networks," *Sensors*, vol. 19, no. 10, p. 2229, 2019.

[11] GOOGLE, "Google maps," 2021, last accessed 16 March 2021. [Online]. Available: https://www.google.com/maps/place/Delhi

[12] BING, "Bing maps," 2021, last accessed 16 March 2021. [Online]. Available: https://www.bing.com/maps/traffic

[13] S. TOPIS, "Seoul transport opearation & informationservice center," 2021, last accessed 16 March 2021. [Online]. Available: https://topis.seoul.go.kr/prdc/openPrdcMap.do

[14] BAIDU, "Baidu maps," 2021, last accessed 16 March 2021. [Online]. Available: https://map.baidu.com/13036895.494262943

[15] A. Makkar and N. Kumar, "An efficient deep learning-based scheme for web spam detection in iot environment," *Future Generation Computer Systems*, vol. 108, pp. 467–487, 2020.

[16] R. Fu, Z. Zhang, and L. Li, "Using lstm and gru neural network methods for traffic flow prediction," in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. IEEE, 2016, pp. 324–328.

[17] W. Wei, H. Wu, and H. Ma, "An autoencoder and lstm-based traffic flow prediction method," *Sensors*, vol. 19, no. 13, p. 2946, 2019.

[18] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[19] A. Makkar and N. Kumar, "Protector: An optimized deep learning-based framework for image spam detection and prevention," *Future Generation Computer Systems*, vol. 125, pp. 41–58, 2021.

[20] W. Pattara-Atikom, P. Pongpaibool, and S. Thajchayapong, "Estimating road traffic congestion using vehicle velocity," in *2006 6th International Conference on ITS Telecommunications*. IEEE, 2006, pp. 1001–1004.

[21] J. Yoon, B. Noble, and M. Liu, "Surface street traffic estimation," in *Proceedings of the 5th international conference on Mobile systems, applications and services*, 2007, pp. 220–232.

[22] Y. Chen, L. Gao, Z.-p. Li, and Y.-c. Liu, "A new method for urban traffic state estimation based on vehicle tracking algorithm," in *2007 IEEE Intelligent Transportation Systems Conference*. IEEE, 2007, pp. 1097–1101.

[23] W. Shi, Q.-J. Kong, and Y. Liu, "A gps/gis integrated system for urban traffic flow analysis," in *2008 11th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2008, pp. 844–849.

[24] Q.-J. Kong, Q. Zhao, C. Wei, and Y. Liu, "Efficient traffic state estimation for large-scale urban road networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 398–407, 2012.

[25] Q.-J. Kong, Z. Li, Y. Chen, and Y. Liu, "An approach to urban traffic state estimation by fusing multisource information," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 3, pp. 499–511, 2009.

[26] J.-D. Zhang, J. Xu, and S. S. Liao, "Aggregating and sampling methods for processing gps data streams for traffic state estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1629–1641, 2013.

[27] L. Li, X. Chen, and L. Zhang, "Multimodel ensemble for freeway traffic state estimations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 3, pp. 1323–1336, 2014.

[28] Y. Feng, J. Hourdos, and G. A. Davis, "Probe vehicle based real-time traffic monitoring on urban roadways," *Transportation Research Part C: Emerging Technologies*, vol. 40, pp. 160–178, 2014.

[29] J. Lu and L. Cao, "Congestion evaluation from traffic flow information based on fuzzy logic," in *Proceedings of the 2003 IEEE International Conference on Intelligent Transportation Systems*, vol. 1. IEEE, 2003, pp. 50–53.

[30] P. Pongpaibool, P. Tangamchit, and K. Noodwong, "Evaluation of road traffic congestion using fuzzy techniques," in *TENCON 2007-2007 IEEE Region 10 Conference*. IEEE, 2007, pp. 1–4.

[31] Y. Chen and Y. Liu, "A new method for gps-based urban vehicle tracking using pareto frontier and fuzzy comprehensive judgment," in *2007 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2007, pp. 683–686.

[32] H. Shankar, P. Raju, K. R. M. Rao *et al.*, "Multi model criteria for the estimation of road traffic congestion from traffic flow information based on fuzzy logic," *Journal of Transportation Technologies*, vol. 2, no. 01, p. 50, 2012.

[33] H. Su and S. Yu, "Hybrid ga based online support vector machine model for short-term traffic flow forecasting," in *International Workshop on Advanced Parallel Processing Technologies*. Springer, 2007, pp. 743–752.

[34] C. Cao and X. Jianmin, "Improved particle swarm optimized svm for short-term traffic flow predication," in *2007 Chinese Control Conference*. IEEE, 2007, pp. 6–9.

[35] Y. Xu, B. Wang, Q.-J. Kong, Y. Liu, F.-Y. Wang, Y. Xu, and F. Wang, "Spatio-temporal variable selection based support vector regression for urban traffic flow prediction," in *Proceeding of the 93rd Annual Meeting of the Transportation Research Board, Washington, DC*, 2014, pp. 14–1994.

[36] W.-C. HONG, Y. DONG, F. ZHENG, and S. Y. WEI, "Hybrid evolutionary algorithms in a svr traffic flow forecasting model," *Applied mathematics and computation*, vol. 217, no. 15, pp. 6733–6747, 2011.

[37] M.-W. Li, W.-C. Hong, and H.-G. Kang, "Urban traffic flow forecasting using gauss–svr with cat mapping, cloud model and pso hybrid algorithm," *Neurocomputing*, vol. 99, pp. 230–240, 2013.

[38] J. Wang and Q. Shi, "Short-term traffic speed forecasting hybrid model based on chaos–wavelet analysis-support vector machine theory," *Transportation Research Part C: Emerging Technologies*, vol. 27, pp. 219–232, 2013.

[39] F. Wang, G. Tan, C. Deng, and Z. Tian, "Real-time traffic flow forecasting model and parameter selection based on $\varepsilon$-svr," in *2008 7th World Congress on Intelligent Control and Automation*. IEEE, 2008, pp. 2870–2875.

[40] G. Jun, Q. Li, L. Mingyue, and C. Xiuyang, "Forecasting urban traffic flow by svr," in *2013 25th Chinese Control and Decision Conference (CCDC)*. IEEE, 2013, pp. 981–984.

[41] H. Chen, H. A. Rakha, and S. Sadek, "Real-time freeway traffic state prediction: A particle filter approach," in *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2011, pp. 626–631.

[42] S. Dunne and B. Ghosh, "Regime-based short-term multivariate traffic condition forecasting algorithm," *Journal of Transportation Engineering*, vol. 138, no. 4, pp. 455–466, 2012.

[43] W. Min and L. Wynter, "Real-time road traffic prediction with spatio-temporal correlations," *Transportation Research Part C: Emerging Technologies*, vol. 19, no. 4, pp. 606–616, 2011.

[44] X. Zhang, E. Onieva, A. Perallos, E. Osaba, and V. C. Lee, "Hierarchical fuzzy rule-based system optimized with genetic algorithms for short term traffic congestion prediction," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 127–142, 2014.

[45] R. Herring, A. Hofleitner, S. Amin, T. Nasr, A. Khalek, P. Abbeel, and A. Bayen, "Using mobile phones to forecast arterial traffic through statistical learning," in *89th Transportation Research Board Annual Meeting*, 2010, pp. 10–14.

[46] P. S. Castro, D. Zhang, and S. Li, "Urban traffic modelling and prediction using large scale taxi gps traces," in *International Conference on Pervasive Computing*. Springer, 2012, pp. 57–72.

[47] X. Zhou, W. Wang, and L. Yu, "Traffic flow analysis and prediction based on gps data of floating cars," in *Proceedings of the 2012 International Conference on Information Technology and Software Engineering*. Springer, 2013, pp. 497–508.

[48] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[49] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

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

[51] H. Fatemidokht, M. K. Rafsanjani, B. B. Gupta, and C.-H. Hsu, "Efficient and secure routing protocol based on artificial intelligence algorithms with uav-assisted for vehicular ad hoc networks in intelligent transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4757–4769, 2021.

[52] S. R. Sahoo and B. B. Gupta, "Multiple features based approach for automatic fake news detection on social networks using deep learning," *Applied Soft Computing*, vol. 100, p. 106983, 2021.

[53] F. Gers, "Long short-term memory in recurrent neural networks," Ph.D. dissertation, Verlag nicht ermittelbar, 2001.

[54] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," 1999.

[55] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever, "Generative pretraining from pixels," in *International Conference on Machine Learning*. PMLR, 2020, pp. 1691–1703.

[56] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[57] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," *arXiv preprint arXiv:2012.12877*, 2020.

[58] Y. Chen, M. Rohrbach, Z. Yan, Y. Shuicheng, J. Feng, and Y. Kalantidis, "Graph-based global reasoning networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 433–442.

[59] X. Liang, Z. Hu, H. Zhang, L. Lin, and E. P. Xing, "Symbolic graph reasoning meets convolutions," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 1858–1868.

[60] S. Zhang, X. He, and S. Yan, "Latentgnn: Learning efficient non-local relations for visual recognition," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7374–7383.