Journal of Intelligent Transportation Systems
Technology, Planning, and Operations

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/gits20

Metaheuristic enabled deep convolutional neural network for traffic flow prediction: Impact of improved lion algorithm

Monal Patel, Carlos Valderrama & Arvind Yadav

To cite this article: Monal Patel, Carlos Valderrama & Arvind Yadav (2021): Metaheuristic enabled deep convolutional neural network for traffic flow prediction: Impact of improved lion algorithm, Journal of Intelligent Transportation Systems, DOI: 10.1080/15472450.2021.1974857

To link to this article: https://doi.org/10.1080/15472450.2021.1974857

Published online: 06 Sep 2021.

JOURNAL OF INTELLIGENT TRANSPORTATION SYSTEMS
https://doi.org/10.1080/15472450.2021.1974857

Metaheuristic enabled deep convolutional neural network for traffic flow prediction: Impact of improved lion algorithm

Monal Patel[a], Carlos Valderrama[b], and Arvind Yadav[a]

[a]Parul University, Vadodara, Gujarat, India; [b]University of Mons, Mons, Belgium

ABSTRACT

Traffic flow prediction is a basic aspect to be considered in transportation management and modeling. Attaining precise information on near and current traffic flows has an extensive range of appliances and it further aids in managing the congestion. Numerous conventional models failed at offering precise prediction results due to "shallow in architecture and hand engineered in features". Moreover, the raw traffic flow information contains noise that might lead to the worst prediction results. Therefore, this paper intends to design an enhanced prediction model on traffic flow using Optimized Deep Convolutional Neural Network (DCNN). The input features or the technical indicators subjected to the optimized CNN are Average True Range (ATR), Exponential Moving Average (EMA), Relative Strength Indicator (RSI) and Rate of Change (ROC), respectively. Moreover, for precise prediction, the weights of DCNN are optimally tuned using a new Improved Lion Algorithm (LA) termed as Lion with New Territorial Takeover Update (LN-TU) model. In the end, the betterment of implemented work is compared and proved over the conventional models in terms of error analysis and prediction analysis.

Nomenclature: ARIMA: autoregressive integrated moving average; ATR: average true range; BPNN: back propagation neural network; CNNs: convolutional neural networks; DBN: deep belief network; DCNN: deep convolutional neural network; DNN-BTF: deep NN based traffic flow prediction; DL: deep-learning; DRN: deep residual network; EMA: exponential moving average; FDCN: fuzzy deep convolutional network; GCNs: graph convolution networks; GRCNN: graph RCNN; HA: historical average; LA: lion algorithm; LN-TU: lion with new territorial takeover update; LSTM: long short-term memory network; MAE: mean absolute error; MLE: maximum likelihood estimation; MSE: mean square error; RMSE: root MSE; RNNs: recurrent neural networks; ROC: rate of change; RSI: relative strength indicator; STEF-Net: Spatio TEmporal Fuzzy neural Network; STANN: spatial and temporal attentions; SMA: simple moving average; SVM: support vector machine; VAR: vector autoregressive; WNN: wavelet neural network; 4D: four-dimensional; SE: standard error

ARTICLE HISTORY
Received 28 June 2020
Revised 27 August 2021
Accepted 27 August 2021

KEYWORDS
Congestion; DCNN model; error analysis; LN-TU approach; traffic flow

1. Introduction

Precise prediction of traffic flow is a significant factor to be concerned in present transport systems. It is a boost up for copious appliances that necessitate appropriate information regarding traffic in the future (Lin et al., 2019; Miglani and Kumar, 2019; Wang et al., 2020). For instance, the predicted traffic flow is a noteworthy reference for scheduling the vehicle paths and therefore it can aid travelers to make a better choice regarding routes. Predicting "when and where" congestion occurs is much advantageous for dealing with transportation issues, as experts will be able to assign resources to the roads during the risks of congestion and eventually, it can reduce the traffic congestion. Therefore, due to its massive advantage on different appliances, traffic flow prediction (Deng et al., 2019; Ryu et al., 2018; Tian et al., 2018) has turned out to be a major topic of research in recent times. Generally, predicting the traffic flow makes evaluations of the upcoming state depending on knowledge and experiences extracted from associated chronological data. As a result, methods on transmission, data collection, mining, and storage have a massive impact on prediction techniques (Emami et al., 2020; Kong et al., 2019; Cheng et al., 2017; (Rewadkar and Doye, 2019; Rewadkar and Doye, 2018).

There are several traditional traffic forecasting approaches (Li et al., 2016) that comprise of temporal dependence, together with the HA, ARIMA model, Gaussian process-based, Kalman filtering model, VAR,

CONTACT Monal Patel Parul University, Vadodara, Gujarat 391760, India.

etc, and it also considered multi-source data fusion, along with tensor factorization model, coupled matrix and mixed Gaussian probability model and so on. Nevertheless, the dynamic variations of traffic flows and the outcomes achieved by these techniques remain disappointing in terms of prediction accuracy. Furthermore, improvement in accuracy can directly equate to financial/cost savings for supply chain/logistics companies (Aramini and Fan 2019; Candini et al., 2019; Li et al., 2015; Taghipour and Frayret, 2010; Vosooghidizaji, et al, 2020; Yang and Hu, 2016).

In recent times, deep learning techniques (Guo et al., 2014; Li et al., 2020) have revealed optimistic outcomes in a dynamic prediction model. Significantly, deep learning models namely, RNNs based models, CNNs based models, and GCNs based models have attained much popularity owing to its higher capability of learning varied features of spatial-temporal traffic data (Peng and Xiang, 2020; Wu et al., 2018). The aforesaid approaches consider the spatial dependency and temporal dependency, however, they pay no consideration to the modeling of relationships regarding dynamic traffic stations and past traffic passenger flows. Therefore, consideration of both potential relationships and traffic topological structures amongst stations from past traffic passenger's flow will help in forecasting the traffic flows in an accurate manner (Benocci et al., 2019). In summary, Traffic flow prediction is a fundamental problem in transportation management and the prediction model is done by optimized DCNN. For precise prediction, a novel LN-TU algorithm is introduced to select the weights optimally. Experimental results show that the proposed method has better performance on 33 datasets. The error performance of the proposed LN-TU model has attained minimal error for all measures when compared over the existing schemes in terms of MAE, MSE and RMSE measures. The prediction analysis of the adopted LN-TU model was almost equal to the actual value whereas the conventional methods show poor performance. The main contribution of the presented work is given below:

1. An improved traffic flow prediction model using optimized DCNN is introduced. The weights of the DCNN are fine tuned.
2. For the optimal selection of weights, a novel Lion with New Territorial Takeover Update (LN-TU) is established, which is an improved version of the LA model is introduced to reduce the error.
3. The efficiency of the proposed model is validated using 33 datasets which ensures the effectiveness and versatility of the research. The performance of the proposed research is compared with the other existing researches on the basis of error analysis.

The paper is arranged as follows: The reviews are presented in section II. Section III describes the implemented model for traffic flow prediction and section IV portrays the optimized DCNN based classification using improved lion algorithm. In addition, the results are discussed in section V and the paper is concluded by section VI.

2. Literature review

2.1. Related works

Hou et al., (2019) have established the non-linear WNN technique and linear ARIMA technique for predicting the traffic flow. Subsequently, outputs of these two approaches were examined and united by fuzzy logic, which offered the absolute predicted traffic flow in the network. The outcomes have indicated that the hybridized scheme offered better stability even sporadic conditions and it was also found to be error-free and accurate.

Do et al., (2019) have established a deep learning-oriented model termed as STANN for predicting the traffic flow. The temporal and spatial attention was exploited for finding out the spatial dependencies among temporal dependencies and road segments correspondingly. Finally, the analysis outcomes revealed the betterment of the implemented scheme in terms of prediction accuracy and it also analyzed the exploitation of numerous data resolutions.

Yang et al., (2019) have presented an enhanced scheme, which connected the high-impact value of longer sequence time steps to the present time step. These traffic flow values with higher impact were captivated using the attention strategy. Simultaneously, some data were stabilized from the normal range for attaining improved prediction outcomes. The investigational results proved the enhancement of the implemented model in terms of short-term prediction of traffic flow.

Tang et al., (2019) have determined a prediction technique that combined the SVM scheme and de-noising model for enhancing the prediction accuracy. This method computed the multi-step prediction of approaches with diverse de-noising schemes by deploying the traffic data gathered from the "city of Minneapolis". The results attained from prediction demonstrated that the prediction outcomes including

Table 1. Reviews on conventional traffic flow prediction systems.

Author	Adopted methods	Features	Challenges
(Hou et al., 2019)	WNN	• Reduced error • Reliable performance	• Requires consideration on more informed technique.
(Do et al., 2019)	STANN	• Accurate predictions • Analyses the temporal and spatial features	• Needs consideration on time dependency
(Yang et al., 2019)	RNN	• Smoothens the noise • Minimal error	• Have to concern on space and time features
(Tang et al., 2019)	SVM	• Minimizes the noise • Highly accurate	• Have to focus more on the multi source flow of traffic data
(Zhao et al., 2019)	DBN	• Reduced error • Minimizes the training time	• Parallel computing model is not concerned
(Chen et al., 2018)	FDCN	• Reduced data uncertainty • More accurate	• Requires more consideration on reinforcement learning
(Peng et al., 2020)	Dynamic-GRCNN	• Minimal error value • Reduced time consumption	• Complex and real-time prediction has to be focused more
(Wu et al., 2018)	DNN-BTF	• Offers stable performance • Highly reliable	• Have to focus on deep learning models
Liang *et al.*, (2019)	STEF-Net	• Achieves the smallest RMSE and MAE	• The duration of the network is unknown

de-noising model were better than the predicted outcomes without de-noising approach. This technique also offered important suggestions for electing the suitable de-noising strategy for predicting the traffic flow.

Zhao et al., (2019) have decomposed single dataset into numerous sub-datasets, which were distributed to various computational nodes. In the presented work, learning schemes and algorithms were presented for computing the learning of the DBN framework. Moreover, a "master-slave parallel computing structure" was modeled, in which the features were learned by the slave computing nodes that were then transmitted back to the master computing node. The results have revealed a minimal time consumption of the proposed work when evaluated over the traditional works.

Chen et al., (2018) have proposed a new fuzzy oriented approach for predicting traffic flow. This method was modeled based on the fuzzy approach and the DRN scheme. Here, the fuzzy approach was introduced into the DL model for lessening the impact of data error. Moreover, FDCN model was introduced for enhancing the prediction of traffic flow and accordingly, the temporal and spatial correlation of traffic flow was analyzed.

Peng et al., (2020) have developed a new Dynamic-GRCNN that captured the spatio-temporal features of traffic flow for predicting the traffic flows of urban passengers accurately. Moreover, for utilizing the entire historical flow of passengers, the long-term, medium-term and short-term historical traffic data were sampled during training that captured the trend and periodicity of the passenger flow traffic at diverse stations.

Wu et al., (2018) have introduced a DNN-BTF model for improving the accuracy of prediction. The adopted scheme exploited the temporal-spatial characteristics and daily/weekly periodicity of traffic flow. In addition, an attention oriented approach was established that automatically determined the significance of previous traffic flow. Moreover, RNN and CNN were deployed for mining the temporal features and spatial features of the traffic flow.

Liang *et al.* (2019) have proposed a STEF-Net method to accurately predict passenger demands by combining deep learning with a fuzzy neural network. This method can handle complex input dependencies like spatial and temporal factors. A new feature fusion method with convolution followed by an attention layer was performed to combine the two neural networks into one and keep temporal relations for further temporal relevance modeling. This model outperforms the other methods with a 10% improvement in RMSE.

2.2. Review

Table 1 shows the reviews on the traffic flow prediction systems. Initially, WNN was exploited in (Hou et al., 2019) that offers reduced error with reliable performance, but it requires consideration on more informed technique STANN was deployed in (Do et al., 2019) that offers accurate predictions and it also analyses the temporal and spatial features. Nevertheless, it needs consideration on time dependency. Also, the RNN model was exploited in (Yang et al., 2019), which smoothens the noise and it is highly prone to error. However, it has to concern with space and time features. Likewise, SVM was introduced in (Tang et al., 2019), which offers high accuracy along with minimal noise. However, it has to concern more on the multi-source flow of traffic data. DBN was exploited in (Zhao et al., 2019) that offers

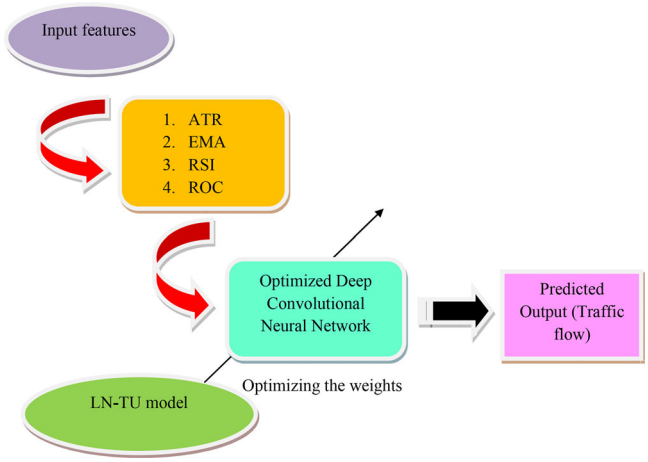

Figure 1. Architecture of proposed model.

reduced error and it also minimizes the training time, however, it needs consideration on the parallel computing model. Moreover, the FDCN model was deployed in (Chen et al., 2018), which reduces the data uncertainty with high accuracy; however, it needs consideration on reinforcement learning. Dynamic-GRCNN was implemented in (Peng et al., 2020), which offers minimal error value and it reduces the time consumption, however complex and real-time predictions are not concerned. Finally, the DNN-BTF model was suggested in (Wu et al., 2018) that offers stable performance high and it also presents better reliability. However, it has to focus on deep learning models.

3. Implemented model for traffic flow prediction

Figure 1 shows the pictorial representation of the presented traffic flow prediction model. For predicting the optimal flow of traffic, the presented model concerns four input features namely, ATR, EMA, RSI, and ROC. These features are provided as inputs to an optimized DCNN framework that directly predicts the respective traffic flow. More particularly, to make the prediction more precise, the CNN model is optimized by a novel LN-TU algorithm by optimally tuning the weights. This optimization logic ensures accurate prediction outcomes.

3.1. Feature extraction

Feature extraction is the primary phase, where features such as ATR, EMA, and RSI are extracted. The arithmetical formulation for each feature is specified in this section:

ATR: The ATR computes the size of the period's range, and it also computes the gaps from the close of the prior period [25].(https://www.tradingtechnologies.com/xtrader-help/x-study/technical-indicator-definitions/list-of-technical-indicators). It is modeled as per Eq. (1), where, TR refers to the true range.

$$ATR = Aver(TR, n) \tag{1}$$

EMA: The average of traffic flow is denoted by EMA [25].(https://www.tradingtechnologies.com/xtrader-help/x-study/technical-indicator-definitions/list-of-technical-indicators). It is formulated as per Eq. (2), in which, R indicates the traffic flow for the current interval, D denotes the smoothening constant that will be equal to $2/(nu+1)$, nu represents the count of traffic flow in SMA that is estimated by EMA and EMA_u symbolize the EMA for prior traffic flow.

$$EMA = (R - EMA_u) * D + EMA_u \tag{2}$$

RSI: RSI (https://www.tradingtechnologies.com/xtrader-help/x-study/technical-indicator-definitions/list-of-technical-indicators) is modeled as shown in Eq. (3), where RA refers to the proportion of smoothened average.

$$RSI = 100 - (100/1 - RA) \tag{3}$$

These features are then provided as input to the DCNN model for classification.

4. Optimized DCNN based classification using improved lion algorithm

4.1. Optimized DCNN classification

"DCNN is a biologically inspired trainable architecture that can learn invariant features". All stages in a DCNN include filter bank, certain non-linearity, and

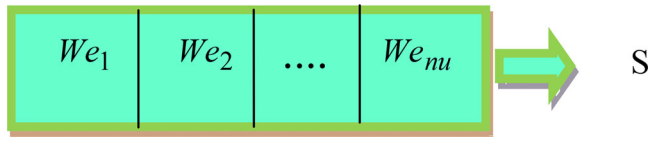

Figure 2. Solution encoding.

feature pooling layers. A DCNN could learn multi-level hierarchical features with numerous stages. In DCNN (Cun et al., 2010), the combined input featuresfe is described by a function as revealed in Eq. (4), such that fe is assigned with a size of$m_1 \times m_2$ and A is the 8-bit channel where the set range from $\{0, \ldots\ldots, 255\}$.

$$\mathrm{Im}_{Se} : \{1, \ldots m_1\} \times \{1, \ldots m_2\} \rightarrow A \subseteq \Re, (i,j) \mapsto fe_{i,j} \tag{4}$$

Consider filter$L \in \Re^{2g_1+1 \times 2g_2+1}$ and for optimal image featuresfe, the discrete convolution (*) with filter L is denoted by Eq. (5), where L is modeled as per Eq. (6).

$$(fe_i * L)_{p,r} := \sum_{v=-g_1}^{g_1} \sum_{u=-g_2}^{g_2} L_{v,u} fe_{p+v,r+u} \tag{5}$$

$$L = \begin{pmatrix} L_{-g_1,-g_2} & \ldots & L_{-g_1,-g_2} \\ \vdots & L_{0,0} & \vdots \\ L_{g_1,-g_2} & \ldots & L_{g_1,g_2} \end{pmatrix} \tag{6}$$

A generally exploited smoothing filter is the discrete Gaussian filter$L_{H(\sigma)}$ as exposed in Eq. (7), where σ refers to the "standard deviation of Gaussian distribution".

$$\left(L_{H(\sigma)}\right)_{p,r} = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{p^2 + r^2}{2\sigma^2}\right) \tag{7}$$

Assume the layer s as a convolutional layer that includes $n_1^{(s)}$ feature maps with $n_2^{(s)} \times n_3^{(s)}$ dimension at its output. Eq. (8) shows the ith feature map in s layer and here $W_i^{(s)}$ points out the bias matrix and $L_{i,j}^{(s)}$ indicates filter dimension of $2g_1^{(s)} + 1 \times 2g_2^{(s)} + 1$ that links the ith feature map in s layer with j^{th} feature map in the layer$(s-1)$.

$$X_i^{(s)} = W_i^{(s)} + \sum_{j=1}^{n_1^{s-1}} L_{i,j}^{(s)} * X_j^{(s-1)} \tag{8}$$

By exploiting the discrete convolution at certain areas of input feature maps, the output feature map holds a dimension as shown in Eq. (9).

$$n_2^{(s-1)} - 2g_1^{(s)} = n_2^{(s)} \text{and} n_3^{(s-1)} - 2g_2^{(s)} = n_3^{(s)} \tag{9}$$

The convolutional layer and its function is defined in Eq. (10) and for relating it with multilayer perceptron, it can be remodeled as given in Eq. (11). All $X_i^{(s)}$ in s layer involves $n_2^{(s)}.n_3^{(s)}$ units. Consequently, Eq. (11) demonstrates the output, achieved by the computation of the unit at the position(p,r).

$$\left(X_i^{(s)}\right)_{p,r} = \left(W_i^{(s)}\right)_{p,r} + \sum_{j=1}^{n_1^{(s-1)}} \left(L_{i,j}^{(s)} * X_j^{(s-1)}\right)_{p,r} \tag{10}$$

$$\left(W_i^{(s)}\right)_{p,r} + \sum_{j=1}^{n_1^{(s-1)}} \sum_{v=-g_1^s}^{g_1^s} \sum_{u=-g_2^s}^{g_2^s} \left(L_{i,j}^{(s)}\right)_{v,u} \left(L_{i,j}^{(s)}\right)_{p,r} \left(X_j^{(s-1)}\right)_{p+v,r+u} \tag{11}$$

In the network, the trainable weights are specified by filters $L_{i,j}^{(s)}$ and bias matrices are indicated by$W_i^{(s)}$.

Assume the fully connected layers with s. If the layer $s-1$ is also the fully connected layers then s takes the input other than $n_1^{(s-1)}$ feature maps having the size $n_2^{(s-1)} \times n_3^{(s-1)}$ input and the j layer with i^{th} unit is measured as per the Eq. (12).

$$x_i^{(s)} = f(v_i^s) \text{ with } v_i^s = \sum_{j=1}^{n_1^{(s-1)}} \sum_{p=-1}^{n_2^{(s-1)}} \sum_{r=1}^{n_3^{(s-1)}} We *_{i,j,p,r}^s \left(X_j^{(s-1)}\right)_{p,r} \tag{12}$$

In which, the weight that associates the unit at the position (g,h) in the jth feature map of layer $s-1$ and the ith unit in s is elucidated by$We *_{i,j,p,r}^s$. Rather than inducing some random values for evaluation, it is worthier to make them optimal for better prediction results. In this work, the tuning is carried out by the concept of optimization. More particularly, a new algorithm is introduced for proper tuning.

4.2. Solution encoding and objective function

The implemented work aims to optimize the weights (..) of DCNN by means of a new LN-TU algorithm. The solution (weights) given to the proposed algorithm is illustrated in Figure 2, wherenu denotes the total count of weights. The objective function *(OF)* of the presented work is defined in Eq. (13), where Er indicates the error. The mathematical formula for error(Er) is shown in Eq. (14) where N is the number of samples, L^A is the ground truth table of actual outcomes and L^P is the ground truth table of predicted outcomes.

$$OF = Min(Er) \tag{13}$$

$$Er = \frac{1}{N}\sum_{i}^{N}(L^A - L^P) \tag{14}$$

4.3. Proposed LN-TU model

The presented work deploys the LN-TU model for optimizing the weights of DCNN model. Here, the existing LA approach is improved so that it could resolve the more complex optimization issues. In general, self-improvement is proven to be promising in traditional optimization algorithms (George and Rajakumar, 2013; Rajakumar, 2013a; 2013b; Rajakumar and George, 2012; Swamy et al., 2013). The LA model (Boothalingam, 2018) was inspired from the living nature of lion species. It encompasses 4 phases such as, "mating, generation of pride, **proposed territorial takeover** and territorial defense" Here, S denotes the solution vector and is addressed as$S = [s_1, s_2, ..., s_{\hat{m}}]$.

4.3.1 Pride generation

The initialization of pride takes place with a territorial lionS^{mal}, nomadic lionS^{nd}, and lionessS^{fem}. The vector component ofS^{nd}, S^{fem} and S^{mal}, that is$s^{nd}_{len}, s^{fem}_{len}$ ands^{mal}_{len} are as the random integers that lies within limits when$\hat{m} > 1$, and here $len = 1, 2....., Len$. Here, Len denotes the lion's length as shown by Eq. (15), where, $\hat{n}$ and $\hat{m}$ are variables. When$\hat{m} = 1$Eq. (16) and (17) should be gratified and $V(f_{len})$ is indicated by Eq. (18).

$$Len = \{ \begin{matrix} \hat{m}; \hat{m} > 1 \\ \hat{n}; otherwise \end{matrix} \tag{15}$$

$$V(s_{len}) \in (s^{\min}_{len}, s^{\max}_{len}) \tag{16}$$

$$\hat{n}\%2 = 0 \tag{17}$$

$$V(s_{len}) = \sum_{len=1}^{Len} s_{len} 2^{\left(\frac{Len}{2} - len\right)} \tag{18}$$

4.3.2. Fertility evaluation

If S^{fem} and S^{mal} gets saturated, then they could have reached local or global optimum, by which they could not attain the best solution. In this approach, the rate of laggardness is indicated as La_r and laggard is considered asS^{mal}, while $h(S^{mal})$ is beyondh^r and it denotes the reference fitness. The fertility of S^{fem} is denoted by sterility rateSt_r. While St_r is greater than tolerance$St^{\max}_r$, S^{fem} the update is performed according to Eq. (19). When an updated female S^{fem+} is regarded asS^{fem} the mating process can be performed. In Eq. (19) and Eq. (20), s^{fem}_d and s^{fem+}_{len} are concerned as d^{th} andlen^{th} vector components ofS^{fem+}.

$$s^{fem+}_{len} = \{ \begin{matrix} s^{fem+}_d if\ len = d \\ s^{fem}_{len} otherwise \end{matrix} \tag{19}$$

$$s^{fem+}_d = \min\left[s^{\max}_d, \max(s^{\min}_d, \nabla_d)\right] \tag{20}$$

$$\nabla_d = \left[s^{fem}_d + (0.1\breve{r}_2 - 0.05)\left(s^{mal}_d - \breve{r}_1 s^{fem}_d\right)\right] \tag{21}$$

The variable d is generated that lies among $[1,\ Len]$, ∇ indicates the female update process and $\breve{r}_1\ \breve{r}_2$ specifies the random constraints, which lies among [0, 1].

4.3.3. Mating

Mating includes the crossover & mutation process, and subsequently, gender-based clustering takes place. By carrying out mutation and crossover, the cubs are created asS^{cubs} andS^{new}, where S^{cubs} refers to the cub produced from crossover andS^{new} refers to the cubs produced from mutation. Thus, four cubs are formed during the pregnancy of a lioness and four cubs are formed by the crossover process. Further, the mutation process is carried out using these four cubs to generate four new cubs.

4.3.4. Lion operators

The territorial defense is addressed by survival fight, which produces coalition updates. S^{e-nd} is selected if Eq. (22) to Eq. (24) are met with.

$$h(S^{e-nd}) < h(S^{mal}) \tag{22}$$

$$h(S^{e-nd}) < h(S^{mal_{cub}}) \tag{23}$$

$$h(S^{e-nd}) < h(S^{fem_{cub}}) \tag{24}$$

The pride update occurs after the failure ofS^{mal}, while the nomad coalition update occurs only after the failure ofS^{nd}.

Proposed Territorial takeover: It is the process of updating S^{mal} and S^{fem} based on the maximum age of cubs$A^{\max}$. Conventionally, there is no pre-defined formula for updating the S^{mal} and S^{fem} during the takeover process. In the proposed logic, the territorial takeover drives the algorithm to update S^{mal} and S^{fem} as per Eq. (25) and Eq. (26). That is, the territorial takeover update takes place based on a random variable $ran\,n$ and the size of male and female cubs.

$$S^{mal} = \left[S^{mal_{cub}} + (S^{fem_{cub}} \times ran\,n(size(S^{fem_{cub}})))\right] \tag{25}$$

$$S^{fem} = \left[S^{fem_{cub}} + (S^{mal_{cub}} \times ran\,n(size(S^{mal_{cub}})))\right] \tag{26}$$

4.3.5. Termination

The model will get terminated only when Eq. (27) or Eq. (28) is satisfiable. In Eq. (26), it denotes the count

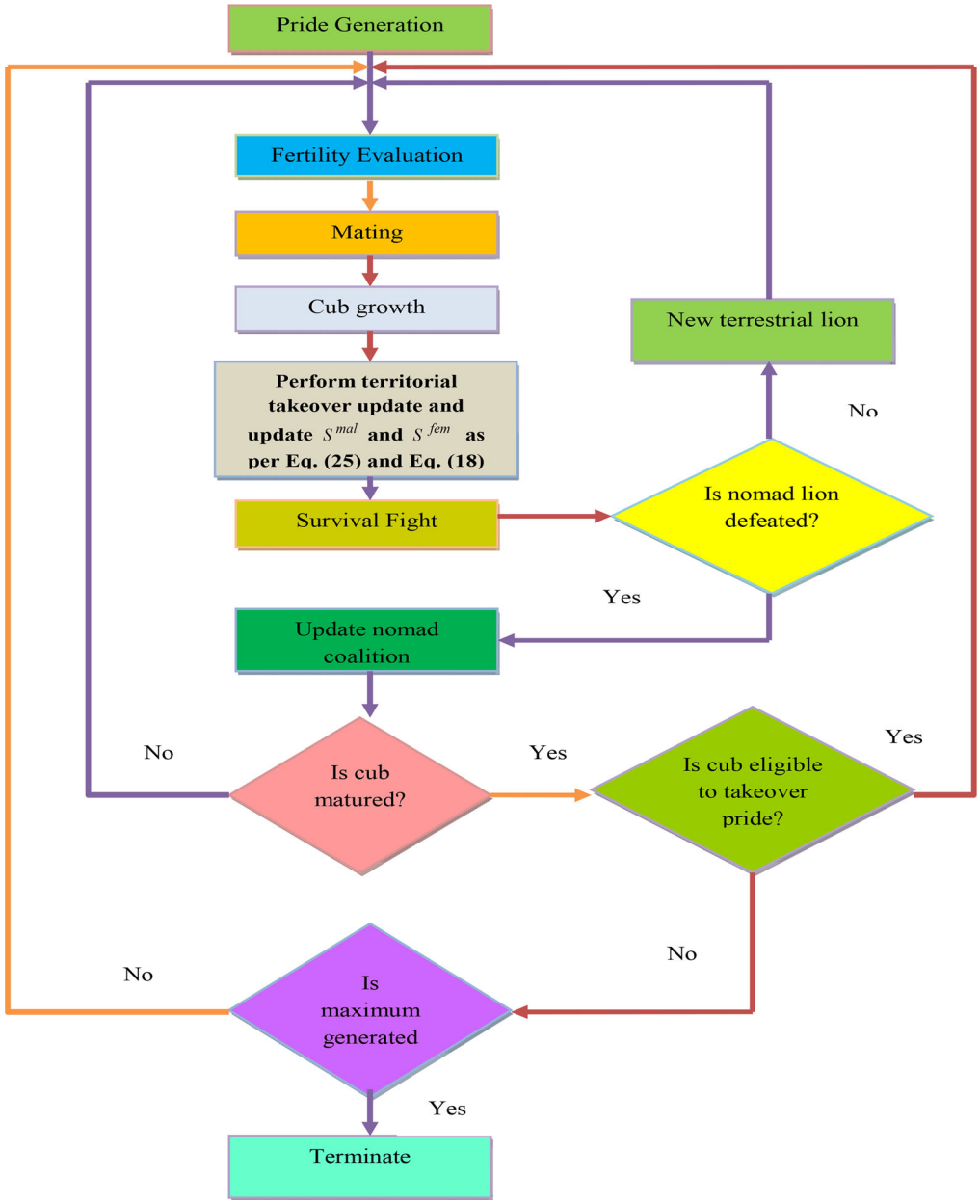

Figure 3. Flowchart of adopted LN-TU model.

of a generation that is initially set as 0 and raised by 1 when the territorial takeover is carried out.

$$it > it_{\max} \tag{27}$$

$$h(S^{mal}) - h(S^{opt}) \leq er_{th} \tag{29}$$

The maximum generations and threshold for error are denoted as $it_{\max}$ and er_{th} respectively. Algorithm 1 depicts the pseudo-code of the presented LN-TU model and its flow chart model is given by Figure 3.

Algorithm 1: Presented LN-TU Model

InitializeS^{mal}, S^{fem} and S_1^{nd}
Compute$h(S^{mal})$ $h(S^{fem})$ and $h(S_1^{nd})$
Fix$h^r = h(S^{mal})$ and $it = 0$
SaveS^{fem} and $h(S^{mal})$
Perform fertility computation as per Eq. (19)
Carry out mating
Perform gender clustering
Allocate A^{cub} as 0
Simulate the cub growth function
If defense is 0, the territorial defense is performed or follow step 4
If $A^{cub} < A^{\max}$, follow step 9.
Perform territorial takeover and update S^{mal} and S^{fem} as per Eq. (25) and Eq. (26)
Increment it by 1
If the termination is not attained, follow 4th step or finish

Table 2. Parameters of CNN.

Layers	Value
Image Input	4x1x1 images with "zerocenter" normalization
Conv 1	16 3x3x1 convolutions with stride [1 1] and padding "same"
Conv 2	16 3x3x16 convolutions with stride [1 1] and padding "same"
Conv 3	16 3x3x16 convolutions with stride [1 1] and padding "same"
Conv4	16 3x3x16 convolutions with stride [1 1] and padding "same"
Relu	ReLU
Fully Connected	1 fully connected layer
Regression Output	mean-squared-error with response "Response"
Max Epochs	50
Mini Batch Size	1000
Solver	sgdm

5. Results and discussion

5.1. Simulation procedure

The presented traffic flow prediction model using the LN-TU model was implemented in **MATLAB**, and the respective outcomes were accomplished. The dataset was downloaded from the link "https://www.kaggle.com/coplin/traffic/data". Dataset consists of 8828 data samples, out of this sample 6180 data samples are used for training and 2648 data samples are used for testing. The data set consists of data that were collected from April 4th, 2015 to January 3rd, 2016, from spring to winter, including sunny, rainy and snowy days. Traffic data were gathered from numerous separated locations on an elevated road in Hangzhou. Locations at which data are collected include the Shangcheng District, the Gongshu District, the Xihu District and the 255 Binjiang District". Moreover, the presented approach was analyzed and its superiority was proved over other conventional schemes such as DBN (Zhao et al., 2019) NN (Wu et al., 2018), CNN Tang (2019), BPNN (Sharma *et al.*, 2018), Gradient Descent (Rahimipour *et al.*, 2019), PSO (Shang et al., 2016), LA (Boothalingam, 2018) for 33 datasets. In addition, the analysis was held concerning error measures such as MAE, MSE and RMSE. Moreover, prediction analysis was also performed that revealed the betterment of the presented model. CNN parameters are given in Table 2, where 16 is the number of filters, 3×3 is the filter size, and 1 is the number of channel.

5.2. Prediction analysis: Proposed Vs. conventional models

Figures 4 and 5 shows the prediction analysis of adopted LN-TU model over conventional approaches like DBN (Zhao et al., 2019) NN (Wu et al., 2018), CNN Tang (2019), BPNN (Sharma *et al.*, 2018), Gradient Descent (Rahimipour *et al.*, 2019), PSO (Shang et al., 2016), LA (Boothalingam, 2018) with respect to vehicle count for the dataset 1 and 4. On analyzing the entire 33 datasets, the predicted output for the implemented scheme has accomplished enhanced performance, since its prediction rate is much closer to the actual value for all vehicle counts. From the graph, it is observed that the predicted output of the proposed model is almost equal to the actual value of 80. Similarly, all the predicted outputs attained by the presented LN-TU model are closer to the actual value. Thus the enhanced performance of the adopted LN-TU-based prediction model has been revealed from the simulation outcomes.

5.3. Error analysis

This section describes the error performance of the adopted LN-TU scheme over the traditional schemes in terms of MAE, MSE and RMSE measures. On observing the analysis outcomes, the proposed LN-TU model has attained minimum error for all measures when compared over the existing schemes. More particularly, on considering the MAE from Table 3, the adopted model for 5th dataset is 14.83%, 73.18%, 74.37%, 73.92%, 1.88%, 80.72%, 80.95%, 80.73% and 1.85% better than SVM, RNN, DBN, NN, LA, CNN, BPNN, Gradient Descent and PSO models. Similarly, on considering the 17th dataset, the implemented model is 24.68%, 80.16%, 97.72%, 80.48%, 8.61%, 80.15%, 80.72%, 80.18% and 24.75% better than SVM, RNN, DBN, NN, LA, CNN, BPNN, Gradient Descent and PSO models Moreover, the presented LN-TU approach at 32rd dataset is 35.06%, 84.18%, 83.96%, 83.92%, and 1.69% better than SVM, RNN, DBN, NN, and LA models. While focusing on the MSE measure from Table 4, the presented LN-TU scheme has accomplished minimum error for all the datasets. Specifically, for 1st dataset, the suggested LN-TU model is 10.52%, 74.77%, 96.47%, 87.59%, and 3.72% better than SVM, RNN, DBN, NN, and LA models. On analyzing the 3rd dataset, the presented model is 12.85%, 80.91%, 96.23%, 91.06%, and 3.55% better than SVM, RNN, DBN, NN and LA models. On

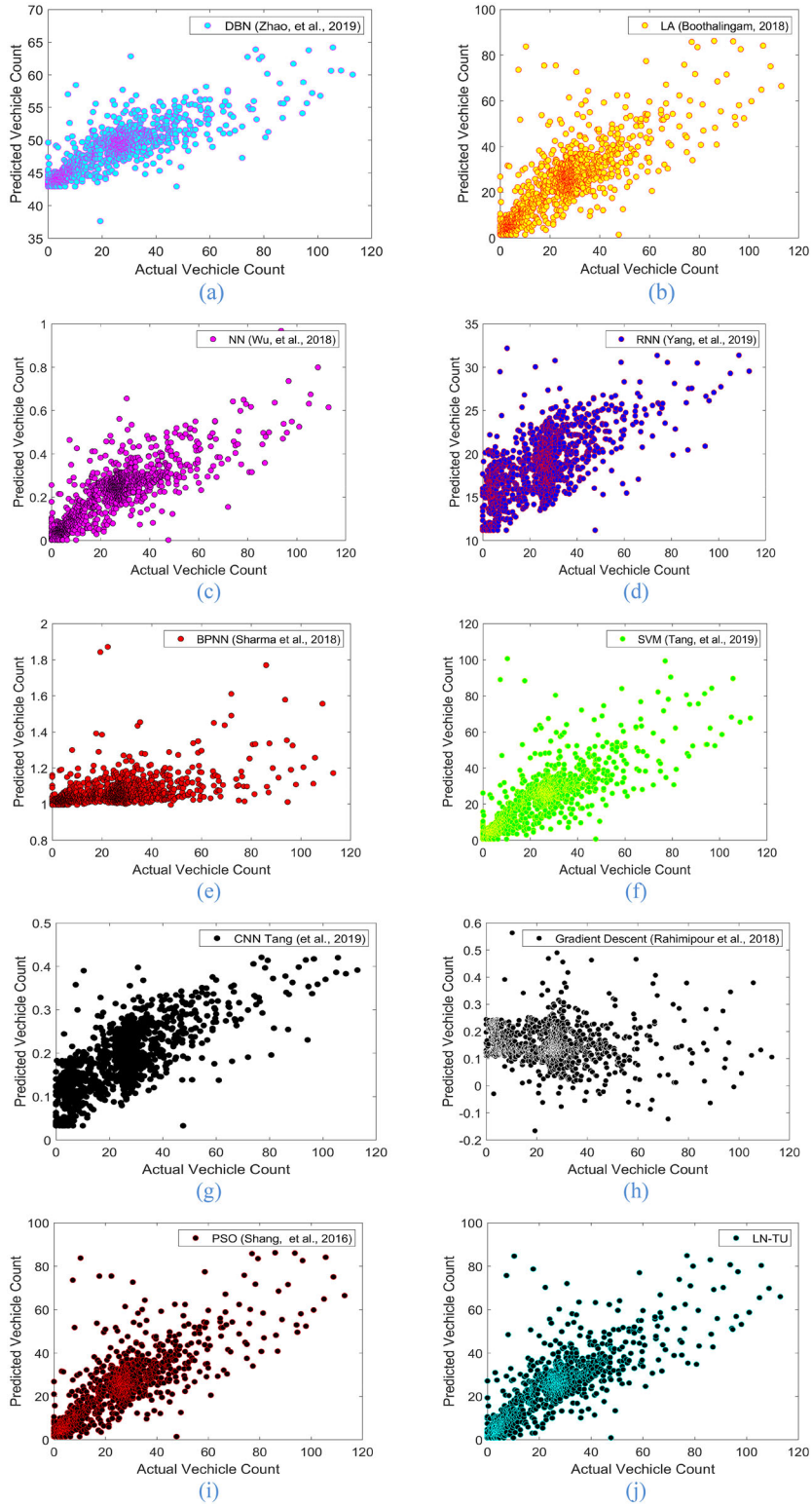

Figure 4. Prediction analysis of the proposed and conventional models for dataset 1 (a) DBN (b) LA (c) NN (d) RNN (e) BPNN (f) SVM (g) CNN (h) Gradient Descent (i) PSO (j) LN-TU.

observing the RMSE measure from Table 5, the adopted scheme at 12th dataset is 4.8%, 57.23%, 73.06%, 72.83%, and 1.02% better than SVM, RNN, DBN, NN, and LA models. In addition, the suggested scheme at 32nd dataset is 10.39%, 68.74%, 80.92%, 80.87%, and 3.16% better than SVM, RNN, DBN, NN and LA models. These attained outcomes show the enhancement of the presented LN-TU model in terms

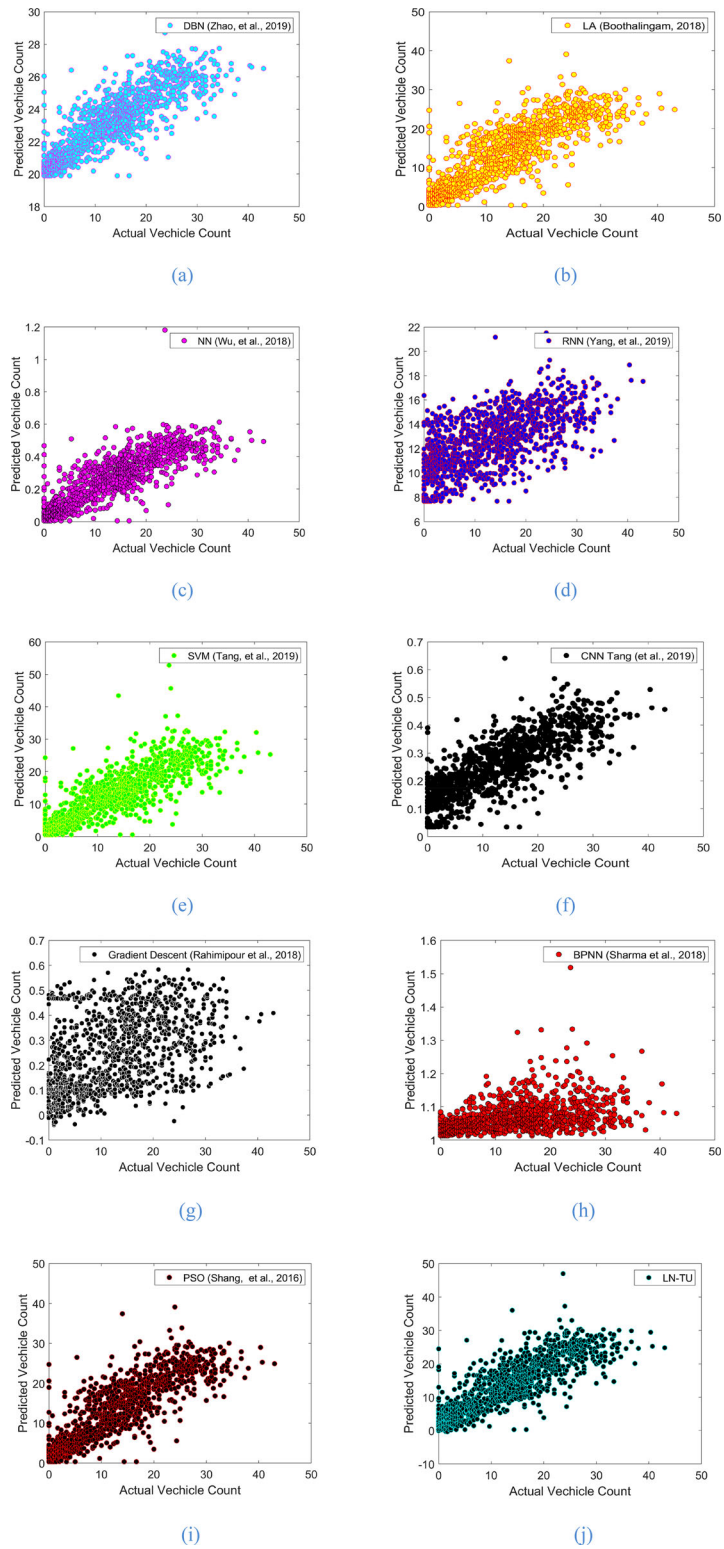

Figure 5. Prediction analysis of the proposed and conventional models for dataset 4 (a) DBN (b) LA (c) NN (d) RNN (e) SVM (f) CNN (g) Gradient Descent (h) BPNN (i) PSO (j) LN-TU.

of error measures. In our proposed method, the neuron weights are optimally selected using the Lion Algorithm with Terrestrial Update. The optimization solution is converged on the basis of minimum error. This Optimal tuning for neuron weights enhances the prediction performance when compared to other existing methods like SVM. In the proposed method, optimal CNN evaluated the features more efficiently than other

Table 3. Mean absolute error analysis: Proposed vs. Conventional models.

Data sets	SVM (Tang et al., 2019)	RNN (Yang et al., 2019)	DBN (Zhao et al., 2019)	NN (Wu et al., 2018)	CNN (Tang, 2019)	BPNN (Sharma et al., 2018)	Gradient Descent (Rahimipour et al., 2019)	PSO (Shang et al., 2016)	LA (Boothalingam, 2018)	LN-TU
1	3.347	11.844	35.74	10.429	10.456	10.655	10.523	4.535	3.768	3.3868
2	1.387	3.458	44.09	3.356	3.365	3.5604	3.376	2.313	1.679	1.5587
3	2.254	6.653	15.11	6.052	6.0621	6.291	6.113	1.988	1.786	1.7176
4	1.873	7.240	5.628	5.524	5.5416	5.7935	5.582	2.413	1.721	1.7136
5	2.633	8.361	8.751	8.599	8.6072	8.8761	8.623	2.391	2.285	2.2427
6	3.451	13.57	31.66	12.23	12.251	12.488	12.279	3.233	2.731	2.7526
7	0.453	26.33	79.603	0.023	0.0197	0.9231	0.079	2.585	1.835	2.3665
8	1.454	25.43	76.085	8.465	8.4611	9.2193	8.535	4.251	2.647	2.8166
9	9.452	32.32	40.224	40.051	40.052	40.283	40.054	7.176	6.552	6.9426
10	10.27	37.91	91.112	45.85	45.858	45.979	45.9	9.203	6.985	6.9062
11	6.792	26.11	55.311	31.17	31.17	31.397	31.164	6.676	5.744	5.6702
12	3.233	14.03	13.516	13.401	13.403	13.597	13.427	3.438	3.253	3.2336
13	5.438	24.2	160.47	26.719	26.721	26.881	26.734	4.556	4.367	4.4311
14	5.626	26.82	175.48	26.389	26.402	26.559	26.447	5.506	4.971	4.3353
15	2.754	10.88	11.704	11.565	11.567	11.711	11.594	3.474	2.445	2.456
16	2.889	14.144	41.768	14.533	14.543	14.684	14.543	9.9307	3.2167	3.1454
17	4.05	15.376	133.93	15.63	15.632	15.761	15.633	4.2886	3.3375	3.0503
18	5.995	19.956	59.49	18.133	18.135	18.241	18.142	5.7551	5.0312	5.1936
19	5.788	30.892	58.594	29.058	29.06	29.263	29.067	5.8134	4.8451	4.9051
20	0.469	6.608	24.034	1.9588	1.9587	2.7493	1.9751	0.82435	0.55083	0.49421
21	0.733	5.336	1.8384	1.8094	1.8285	2.6039	1.8128	0.65905	0.47814	0.48356
22	13.587	44.925	163.95	36.137	36.139	36.565	36.141	14.565	12.04	12.128
23	11.383	42.386	108.14	47.433	47.438	47.68	47.454	8.4642	8.3118	7.9538
24	0.443	0.192	0.17019	0.17038	0.1731	0.91993	0.1895	0.18874	0.18731	0.18591
25	0.230	0.0750	0.866	0.0440	0.046146	0.98252	0.04572	0.067868	0.071568	0.0687
26	0.200	0.037	0.75039	0.0225	0.0235	0.994	0.024	0.035338	0.036617	0.034712
27	9.7364	30.629	56.079	35.157	0.17999	1.0481	0.17961	36.132	8.6604	34.941
28	8.1591	36.389	280.02	30.218	0.10863	1.0358	0.12307	32.161	6.1836	31.974
29	6.1905	19.871	192.94	17.637	35.245	35.417	35.263	20.032	4.4829	19.271
30	3.673	23.907	77.022	5.3168	30.264	30.412	30.25	23.69	2.4117	24.833
31	22.276	84.888	79.496	79.36	17.582	17.805	17.652	65.471	14.774	65.44
32	15.459	64.736	63.843	63.699	5.2141	5.54	5.2312	63.328	10.418	59.362
33	2.7765	16.508	33.206	16.214	79.371	79.523	79.37	64.474	2.7283	64.561

Table 4. Mean square error analysis: Proposed vs. Conventional models.

Data sets	SVM (Tang et al., 2019)	RNN (Yang et al., 2019)	DBN (Zhao et al., 2019)	NN (Wu et al., 2018)	CNN (Tang 2019)	BPNN (Sharma et al., 2018)	Gradient Descent (Rahimipour et al., 2019)	PSO (Shang et al., 2016)	LA (Boothalingam, 2018)	LN-TU
1	55.327	196.22	1402.6	398.84	400.18	383.02	401.92	64.378	51.422	49.509
2	4.9282	16.64	1960.7	36.268	36.351	30.869	36.559	11.034	7.2474	5.9246
3	11.681	53.32	269.84	113.82	114.09	105.18	113.94	12.043	10.555	10.18
4	11.545	61.375	110.8	106.88	107.24	99.711	107.45	12.644	10.618	10.73
5	17.526	80.462	237.74	229.56	229.71	219.31	229.6	16.668	16.216	16.437
6	28.531	203.53	1175.6	438.82	439.47	421.42	439.76	27.396	23.328	24.297
7	0.20521	693.46	6336.6	0.0005	0.0003	0.9932	0.0063	6.6855	3.3707	5.6003
8	19.819	669.03	6129.3	661.27	661.33	649.04	661.56	31.564	23.538	24.697
9	213.13	1249.8	4797.8	4757.4	4758.4	4707.9	4758.7	204.66	188.13	188.51
10	226.97	1984.4	10173	6198.2	6201.4	6128.6	6212.5	318.81	191.48	192.14
11	159.83	912.74	3777.5	3062.7	3063.6	3021.2	3061.7	186.74	144.3	149.07
12	47.674	236.13	595.24	585.42	586.29	566.66	584.9	47.649	44.094	43.203
13	85.927	673.97	27199	2146.8	2147.3	2107.6	2146.3	80.859	76.772	76.079
14	88.972	825.9	32185	2074.5	2076.7	2035.8	2076.6	105.18	81.131	83.156
15	22.797	141.8	406.83	397.56	398.21	381.14	398.79	38.196	21.099	20.985
16	45.043	284.51	1942.3	686.54	687.74	665.68	687.26	164.93	40.534	41.224
17	49.2	353.66	18575	873.62	874.57	849.09	873.76	63.314	45.385	43.36
18	143.96	700.73	3839.8	1419.2	1420	1391.7	1421.8	133.9	125.43	126.95
19	104.47	1038.8	4313.4	2405.6	2406.4	2360.6	2408	109.72	96.503	95.595
20	3.3113	47.459	597.81	44.537	44.621	42.604	44.736	4.2636	3.5821	3.598
21	3.7462	31.543	39.095	37.945	38.095	36.229	38.082	4.4632	3.7193	3.7127
22	1375.7	5652.1	27880	10661	10665	10620	10656	1372.1	1197.6	1244.2
23	340.42	2327.4	14068	7212.7	7215.6	7142.4	7214.9	332.07	337.09	313.36
24	0.25789	0.19171	0.21761	0.2011	0.2082	0.90573	0.22365	0.19214	0.19277	0.19025
25	0.0552	0.0135	0.7614	0.0123	0.0124	0.9780	0.0132	0.01257	0.012907	0.012762
26	0.0418	0.0067	0.566	0.0064	0.0065	0.993	0.0065	0.0065	0.0067	0.0066
27	329.84	1264.8	3635.3	3822.8	0.0856	1.106	0.0847	3594.2	314.86	306.02
28	134.08	1578	80241	2730.3	0.0312	1.0772	0.0428	2826	141.74	130.88
29	62.908	476.8	37909	987.96	3840.9	3787.4	3839.1	1178.4	67.69	70.468
30	20.447	635.93	5982.5	116.57	2739.2	2687.5	2737.8	1681.4	25.378	44.575
31	1191.2	8166.6	22846	22768	984.02	964.31	986.29	16028	1125.9	1120.8
32	571.74	4698.8	12605	12549	113.23	107.26	113.21	9975.1	489.55	459.07
33	31.267	304.29	1355.7	793	22788	22669	22789	16449	27.741	30.162

Table 5. Root mean square error analysis: Proposed vs. Conventional models.

Data sets	SVM (Tang et al., 2019)	RNN (Yang et al., 2019)	DBN (Zhao et al., 2019)	NN (Wu et al., 2018)	CNN (Tang 2019)	BPNN (Sharma et al., 2018)	Gradient Descent (Rahimipour et al., 2019)	PSO (Shang et al., 2016)	LA (Boothalingam, 2018)	LN-TU
1	7.4382	14.008	37.452	19.971	20.005	19.571	20.048	8.0236	7.1709	7.0363
2	2.2199	4.0792	44.28	6.0223	6.0292	5.556	6.0464	3.3217	2.6921	2.4341
3	3.4177	7.302	16.427	10.669	10.681	10.256	10.674	3.4703	3.2489	3.1906
4	3.3978	7.8342	10.526	10.338	10.356	9.9856	10.366	3.5559	3.2585	3.2756
5	4.1864	8.9701	15.419	15.151	15.156	14.809	15.153	4.0826	4.027	4.0542
6	5.3414	14.266	34.287	20.948	20.964	20.529	20.97	5.2341	4.8299	4.9292
7	0.45301	26.334	79.603	0.0230	0.0197	0.9943	0.0797	2.5856	1.836	2.3665
8	4.4518	25.866	78.29	25.715	25.716	25.476	25.721	5.6182	4.8516	4.9696
9	14.599	35.352	69.266	68.974	68.981	68.614	68.983	14.306	13.716	13.73
10	15.066	44.547	100.86	78.728	78.749	78.286	78.82	17.855	13.837	13.861
11	12.642	30.212	61.462	55.341	55.35	54.966	55.332	13.665	12.013	12.209
12	6.9046	15.367	24.398	24.195	24.213	23.805	24.185	6.9028	6.6404	6.5729
13	9.2697	25.961	164.92	46.334	46.338	45.909	46.329	8.9921	8.7619	8.7223
14	9.4325	28.738	179.4	45.547	45.57	45.12	45.569	10.256	9.0073	9.119
15	4.7747	11.908	20.17	19.939	19.955	19.523	19.97	6.1803	4.5934	4.5809
16	6.7114	16.867	44.071	26.202	26.225	25.801	26.216	12.843	6.3666	6.4206
17	7.0143	18.806	136.29	29.557	29.573	29.139	29.559	7.957	6.7369	6.5848
18	11.998	26.471	61.966	37.672	37.683	37.305	37.706	11.572	11.2	11.267
19	10.221	32.231	65.677	49.047	49.055	48.586	49.071	10.475	9.8236	9.777
20	1.8197	6.8891	24.45	6.6736	6.6799	6.5272	6.6885	2.0648	1.8927	1.896
21	1.9355	5.6163	6.2526	6.16	6.1721	6.0191	6.171	2.1126	1.9285	1.9268
22	37.09	75.181	166.97	103.25	103.27	103.05	103.23	37.042	34.606	35.27
23	18.451	48.243	118.61	84.928	84.945	84.513	84.94	18.223	18.36	17.702
24	0.50782	0.43784	0.46648	0.44844	0.45629	0.9517	0.47291	0.43833	0.43906	0.4361
25	0.2350	0.11659	0.8725	0.1112	0.11166	0.9884	0.1150	0.11212	0.11361	0.11297
26	0.2046	0.0820	0.7529	0.080225	0.0806	0.996	0.0811	0.0810	0.08189	0.0813
27	18.162	35.564	60.294	61.829	0.29273	1.0517	0.29117	59.952	17.744	17.493
28	11.579	39.724	283.27	52.252	0.17675	1.0379	0.20706	53.16	11.905	11.44
29	7.9314	21.836	194.7	31.432	61.975	61.542	61.96	34.328	8.2274	8.3945
30	4.5218	25.218	77.346	10.797	52.337	51.841	52.324	41.005	5.0377	6.6764
31	34.514	90.369	151.15	150.89	31.369	31.053	31.405	126.6	33.554	33.478
32	23.911	68.548	112.27	112.02	10.641	10.356	10.64	99.875	22.126	21.426
33	5.5917	17.444	36.82	28.16	150.96	150.56	150.96	128.25	5.267	5.492

Table 6. Mean absolute error analysis: Proposed vs. Conventional models.

Data sets	SVM (Tang et al., 2019)	RNN (Yang et al., 2019)	DBN (Zhao et al., 2019)	NN (Wu et al., 2018)	CNN (Tang, 2019)	BPNN (Sharma et al., 2018)	Gradient Descent (Rahimipour et al., 2019)	PSO (Shang et al., 2016)	LA (Boothalingam, 2018)	Proposed method without LN-TU	Proposed method with LN-TU
1	22.40	23.581	48.148	0.5089	0.13759	0.84714	0.13718	0.11059	0.12264	0.9431	0.10835
2	5.024	5.2678	47.893	0.8253	0.059213	0.97639	0.16795	0.033863	0.03471	0.03129	0.034793
3	11.89	12.372	0.2396	0.4377	0.16178	0.85884	0.28203	0.12029	0.12059	0.1077	0.1181
4	14.48	15.109	23.291	0.6052	0.19459	0.7945	0.25279	0.16394	0.15731	0.1982	0.16214
5	21.57	22.18	25.673	0.8567	0.25398	0.69961	0.33489	0.19193	0.20116	0.20764	0.2006
6	28.61	29.175	47.743	0.1812	0.17782	0.80565	0.15652	0.1246	0.12236	0.1291	0.12624
7	66.53	68.896	95.478	0.3470	0.22593	0.74927	0.21389	0.17161	0.1682	0.1935	0.18213
8	79.21	81.269	217.63	0.1617	0.21122	0.72756	0.32245	0.16584	0.15451	0.15381	0.16848
9	85.41	87.671	106.39	0.597	0.25998	0.69572	0.090414	0.18871	0.20411	0.2196	0.1946
10	46.60	45.131	0.1414	0.1698	0.14425	0.9136	0.16244	0.13062	0.1392	0.1495	0.12723
11	38.97	38.849	82.998	0.3176	0.16319	0.86441	0.14196	0.19215	0.16389	0.14751	0.14632
12	25.82	27.098	47.53	0.2180	0.16519	0.84576	0.22353	0.14846	0.15062	0.14975	0.1449
13	55.564	57.145	84.233	0.2623	0.16558	0.83094	0.22731	0.1688	0.16445	0.16589	0.16492
14	55.573	57.244	89.789	0.1574	0.16991	0.83997	0.17249	0.15627	0.14836	0.14892	0.14321
15	21.091	21.901	36.27	1.154	0.19191	0.84516	0.17151	0.18674	0.16465	0.19528	0.18396
16	25.31	26.871	57.962	0.2297	0.13237	0.8881	0.20214	0.12341	0.12614	0.12896	0.12416
17	28.386	29.178	63.447	0.1725	0.13693	0.91762	0.1571	0.12858	0.12323	0.13841	0.11089
18	33.768	35.389	72.786	0.7468	0.14377	0.89775	0.25518	0.14565	0.13306	0.14395	0.13998
19	62.317	63.324	95.22	0.1322	0.19552	0.84439	0.26475	0.13487	0.12847	0.13892	0.12962
20	17.199	17.899	29.637	0.2412	0.17641	0.8674	0.37061	0.15529	0.14502	0.15921	0.1635
21	16.079	16.515	27.979	0.2102	0.18102	0.88703	0.20135	0.15576	0.15758	0.16913	0.15971
22	76.842	73.985	193.24	0.358	0.16956	0.88368	0.1042	0.182	0.12379	0.17943	0.16862
23	86.961	86.854	162.42	0.3256	0.20379	0.85017	0.34811	0.13	0.18094	0.16391	0.16761
24	0.41971	0.0552	1.6363	0.0264	0.01725	1.0051	0.03865	0.022834	0.022998	0.02284	0.023035
25	0.21622	0.0576	0.8713	0.212	0.042253	0.99751	0.086859	0.037349	0.037421	0.03879	0.037664
26	0.18943	0.0433	0.7467	0.598	0.030014	1.0006	0.071208	0.02812	0.028719	0.02945	0.028196
27	67.961	68.818	189.31	0.240	0.2287	0.7057	0.15314	0.22026	0.2082	0.26156	0.21672
28	134.16	136.78	149.9	0.310	0.25596	0.6521	0.29359	0.20859	0.206	0.20643	0.2036
29	66.295	68.949	95.692	0.13	0.26506	0.73551	0.19086	0.20727	0.19186	0.19536	0.19729
30	41.251	41.389	0.1853	1.030	0.16507	0.85516	0.10423	0.16104	0.16649	0.17663	0.18928
31	282.69	288.03	570.84	0.7922	0.32238	0.55391	0.32946	0.21494	0.22256	0.17318	0.16221
32	140.51	140.8	0.32645	0.6433	0.21841	0.76554	0.22659	0.18058	0.17699	0.18901	0.18441
33	36.806	38.181	53.462	0.2827	0.21968	0.78804	0.16783	0.16263	0.16497	0.16543	0.15705

Table 7. Mean square error analysis: Proposed vs. Conventional models.

Data sets	SVM (Tang et al., 2019)	RNN (Yang et al., 2019)	DBN (Zhao et al., 2019)	NN (Wu et al., 2018)	CNN (Tang, 2019)	BPNN (Sharma et al., 2018)	Gradient Descent (Rahimipour et al., 2019)	PSO (Shang et al., 2016)	LA (Boothalingam, 2018)	Proposed method without LN-TU	Proposed method with LN-TU
1	738.48	557.25	2331.4	0.277	0.03243	0.7353	0.02740	0.0183	0.022371	0.019754	0.017542
2	68.27	27.83	2296.2	0.7322	0.00659	0.95554	0.04384	0.00320	0.0033426	0.00453	0.0033937
3	185.16	153.37	0.07877	0.21324	0.0382	0.75314	0.11639	0.0197	0.019857	0.02454	0.019055
4	299.67	229.06	547.13	0.41434	0.0594	0.66196	0.0948	0.03655	0.033725	0.034857	0.03554
5	621.43	494.65	667.46	0.8077	0.0941	0.53892	0.17154	0.0482	0.052558	0.05768	0.053153
6	1043.6	856.58	2291.7	0.04244	0.04017	0.66589	0.0402	0.0198	0.019199	0.026886	0.020294
7	6174.2	4766.8	9211	0.15986	0.0768	0.59991	0.063501	0.0364	0.035623	0.05896	0.041458
8	7997.4	6647.3	47363	0.044249	0.0587	0.5621	0.1661	0.035758	0.031199	0.03946	0.036749
9	10108	7706.2	11485	0.59611	0.0920	0.53302	0.0130	0.0444	0.051449	0.048903	0.047018
10	4873.7	2115.3	0.0517	0.0501	0.0406	0.85482	0.0495	0.023577	0.026747	0.028944	0.022579
11	3456.9	1577	7007.8	0.13072	0.0507	0.78485	0.0537	0.045258	0.034039	0.045757	0.027581
12	1015.3	740.24	2278.1	0.075088	0.0394	0.74022	0.077772	0.028402	0.029041	0.0284954	0.027051
13	4436.8	3297	7175.1	0.10088	0.0384	0.72156	0.0708	0.035361	0.033609	0.03785	0.033844
14	4367.8	3282.2	8139.6	0.037825	0.0426	0.73043	0.050366	0.032161	0.028801	0.029446	0.027022
15	722.43	483.51	1331	1.3929	0.0545	0.74481	0.042437	0.043112	0.033894	0.05335	0.041842
16	1045.5	729.98	3384.3	0.078912	0.0291	0.8076	0.066124	0.022984	0.023747	0.02469	0.023241
17	1399.8	855.27	4060.9	0.046038	0.03456	0.86165	0.042222	0.0301	0.027917	0.02613	0.022878
18	2283.3	1281.4	5361.6	0.62934	0.0430	0.83352	0.0985	0.03846	0.031518	0.03557	0.035679
19	4891.7	4017.2	9125.1	0.039259	0.0501	0.73363	0.11513	0.023662	0.021197	0.02334	0.021588
20	435.52	321.68	885.91	0.071365	0.0467	0.77708	0.20165	0.03227	0.028341	0.074694	0.035839
21	386.6	276.27	789.74	0.065284	0.0501	0.8142	0.060238	0.032837	0.033555	0.03536	0.034398
22	18191	5775.5	38051	0.15747	0.0712	0.82748	0.0289	0.056669	0.0313	0.03869	0.050272
23	13541	7771.7	26761	0.14411	0.0633	0.75752	0.14221	0.02161	0.038031	0.0384	0.032925
24	0.1774	0.00344	2.6787	0.0018552	0.00153	1.0126	0.00399	0.0013475	0.0013286	0.0016954	0.0013283
25	0.0490	0.00480	0.7631	0.04845	0.00634	0.99994	0.010094	0.00409	0.0040744	0.004095	0.004089
26	0.0377	0.0040	0.5613	0.36311	0.00462	1.005	0.00711	0.0038097	0.0038213	0.003894	0.0038057
27	7093.4	4819.3	35837	0.076558	0.0677	0.56283	0.038559	0.058125	0.051851	0.05869	0.056132
28	24812	18885	22881	0.11987	0.089061	0.48761	0.11811	0.053262	0.051858	0.058934	0.050685
29	6758.1	4820.6	9300.6	0.028548	0.10225	0.59085	0.060377	0.051952	0.044656	0.04953	0.047163
30	4003.7	1757.9	0.09342	1.135	0.0534	0.77311	0.025498	0.033451	0.03596	0.03489	0.04495
31	1.09E + 05	83934	3.26E + 05	0.79534	0.12948	0.39387	0.157	0.055628	0.059765	0.05642	0.033427
32	27669	19945	0.15425	0.43871	0.0678	0.62193	0.078087	0.03923	0.038146	0.05313	0.040875
33	1805.7	1462	2883.9	0.11477	0.0682	0.65	0.051161	0.03352	0.034606	0.03652	0.031316

prediction models. Thus, the proposed CNN with LA-TU outperforms the other prediction models for all the datasets.

5.4. *Validation* results

This section describes the validation results of the adopted LN-TU scheme over the traditional schemes in terms of MAE, MSE and RMSE measures. On observing the analysis outcomes, the proposed LN-TU model has attained minimum error for all measures when compared over the existing schemes. More particularly, on considering the MAE from Table 6, the LU-TU model for 10th dataset is 99.5% better than SVM and RNN, 5.25%, 25.5%, 5.7%, 86.07%, 25.76% better than DBN, NN, CNN, BPNN, Gradient Descent and 76.9% better than PSO, LA and LN-TU (without optimal) models. While focusing the MSE measure from Table 7, the presented LN-TU scheme has accomplished minimum error for all the datasets. The adopted scheme at 24th dataset is 99.25%, 1.19%, 99.95%, 28.4%, 13.1%, 99.86%, 66.70%, 1.42% and 1.12% better than SVM, RNN, DBN, NN, CNN, BPNN, Gradient descent, PSO and LA models. On observing the RMSE measure from Table 8, the adopted scheme at 23rd dataset is 99.25%, 99.19%, 99.95%, 16.4%, 5.6%, 12.56%, 34.60%, 1.56% and 1.13% better than SVM, RNN, DBN, NN, CNN, BPNN, Gradient descent, PSO and LA models. The results presented in Tables 6–8 are obtained for the trained samples. The conventional methods underperform even for the trained samples when compared to Proposed Optimized DCNN. In the proposed model the weights are tuned optimally by employing the LA-TU algorithm. Table 9 represents the validation results of the weight optimization process. The table describes the actual flow value, predicted flow value using Optimal DCNN and predicted flow value using DCNN without optimization. The proposed model (Optimized DCNN) predicted the flow value more accurately when compared to the conventional DCNN. This is because of the optimal tuning of weights. Thus the attained outcomes show the enhancement of the presented LN-TU model in terms of error measures.

5.5. Coefficient of determination of the R^2

R-Squared (R^2 or the coefficient of determination) is a statistical measure in a regression model that determines the proportion of variance in the dependent variable that can be explained by the independent

Table 8. Root mean square error analysis: Proposed vs. Conventional models.

Data sets	SVM (Tang, et al., 2019)	RNN (Yang et al., 2019)	DBN (Zhao et al., 2019)	NN (Wu, et al., 2018)	CNN (Tang, 2019)	BPNN (Sharma et al., 2018)	Gradient Descent (Rahimipour et al., 2019)	PSO (Shang et al., 2016)	LA (Bootha lingam, 2018)	Proposed method without LN-TU	Proposed Method with LN-TU
1	27.175	23.606	48.285	0.52664	0.18009	0.8575	0.16554	0.1354	0.14957	0.13761	0.13244
2	8.2626	5.2755	47.918	0.85573	0.081233	0.97752	0.2094	0.05658	0.057815	0.053291	0.058256
3	13.607	12.384	0.28066	0.46178	0.19564	0.86784	0.34116	0.1404	0.14092	0.14629	0.13804
4	17.311	15.135	23.391	0.64369	0.24374	0.81361	0.30804	0.19111	0.18364	0.18547	0.18852
5	24.929	22.241	25.835	0.89875	0.30676	0.73411	0.41418	0.22066	0.22926	0.23801	0.23055
6	32.305	29.267	47.872	0.20602	0.20042	0.81602	0.20069	0.14077	0.13856	0.15391	0.14246
7	78.576	69.042	95.974	0.39982	0.27727	0.77454	0.25199	0.19195	0.18874	0.23651	0.20361
8	89.428	81.531	217.63	0.21035	0.2423	0.74973	0.40755	0.1891	0.17663	0.18539	0.1917
9	100.54	87.785	107.17	0.77208	0.30338	0.73008	0.11444	0.21153	0.22682	0.22981	0.21684
10	69.812	45.992	0.22748	0.22405	0.20162	0.92457	0.22249	0.15355	0.16354	0.173481	0.15026
11	58.795	39.712	83.713	0.36156	0.22521	0.88592	0.23188	0.21274	0.1845	0.174531	0.16607
12	31.864	27.207	47.73	0.27402	0.1984	0.86036	0.27888	0.16853	0.17041	0.17649	0.16447
13	66.609	57.419	84.706	0.31762	0.1961	0.84944	0.26609	0.18804	0.18333	0.193472	0.18397
14	66.09	57.291	90.22	0.19449	0.20653	0.85465	0.22442	0.17933	0.16971	0.17813	0.16438
15	26.878	21.989	36.483	1.1802	0.2335	0.86303	0.206	0.20763	0.1841	0.195328	0.20455
16	32.334	27.018	58.175	0.28091	0.17083	0.89867	0.25715	0.1516	0.1541	0.159032	0.15245
17	37.414	29.245	63.725	0.21456	0.1859	0.92825	0.20548	0.17359	0.16708	0.18732	0.15125
18	47.784	35.797	73.223	0.79331	0.20754	0.91297	0.31395	0.19611	0.17753	0.198421	0.18889
19	69.941	63.382	95.525	0.19814	0.22403	0.85652	0.33931	0.15382	0.14559	0.154871	0.14693
20	20.869	17.935	29.764	0.26714	0.21617	0.88152	0.44906	0.17964	0.16835	0.17515	0.18931
21	19.662	16.621	28.102	0.25551	0.22385	0.90233	0.24543	0.18121	0.18318	0.19745	0.18547
22	134.87	75.997	195.07	0.39683	0.26701	0.90966	0.17	0.23805	0.17692	0.26591	0.22421
23	116.36	88.157	163.59	0.37962	0.25168	0.87035	0.37711	0.147	0.19502	0.20156	0.18145
24	0.42129	0.058663	1.6367	0.043072	0.039166	1.0063	0.063177	0.036298	0.036449	0.03183	0.036445
25	0.22144	0.069285	0.87358	0.22011	0.079678	0.99997	0.10047	0.064025	0.063831	0.06321	0.063945
26	0.19423	0.063977	0.74921	0.60259	0.067975	1.0025	0.084343	0.06169	0.061817	0.061732	0.061723
27	84.223	69.421	189.31	0.27669	0.2602	0.75022	0.19637	0.24109	0.22771	0.22853	0.23692
28	157.52	137.42	151.26	0.34623	0.29843	0.69829	0.34367	0.23079	0.22772	0.227532	0.22513
29	82.208	69.431	96.439	0.16896	0.31977	0.76866	0.24572	0.22793	0.21132	0.21943	0.21717
30	63.275	41.927	0.30565	1.0654	0.23129	0.87927	0.15968	0.1829	0.18963	0.22452	0.21201
31	329.58	289.71	570.84	0.89182	0.35984	0.62759	0.39738	0.23586	0.24447	0.19273	0.18283
32	166.34	141.23	0.39275	0.66235	0.2604	0.78863	0.27944	0.19806	0.19531	0.21256	0.20218
33	42.493	38.236	53.702	0.33878	0.26126	0.80623	0.22619	0.18308	0.18603	0.18867	0.17696

Table 9. Analysis of the Actual vs Predicted values.

Actual value	Optimized DCNN	Without Optimization
37.667	25.658	16.721
35.667	36.183	28.194
32	36.312	27.939
32.667	33.218	25.796
31.667	30.336	15.016
35.333	28.2	13.99
29.667	31.834	25.242
33.333	26.747	17.979
32.667	33.149	24.725
30.667	33.541	23.077
36.333	29.682	23.198

Table 10. Coefficient of determination of the R^2 for dataset 1.

	Estimate	SE	τ value	P value
Intercept	0.50413	0.14945	3.3732	0.00074767
x1	4.0739	1.6484	2.4714	0.013485
x2	82.687	0.79197	104.41	0
x3	4.2417	0.31142	13.62	1.21E-41

Linear regression model: y ∼ 1 + x1 + x2 + x3

variable. In other words, r-squared shows how well the data fit the regression model (the goodness of fit). R-squared can take any values between 0 to 1. Tables 10 and 11 depicts the coefficient determination of the R^2 for dataset 1 and dataset 4. Table 12 represents the regression Statistics of dataset 1 and dataset 4.

Table 11. Coefficient of determination of the R^2 for dataset 4.

	Estimate	SE	τ value	p value
Intercept	0.9643	0.18767	5.1382	2.86E-07
x1	10.019	1.661	6.0316	1.72E-09
x2	91.9	0.83532	110.02	0
x3	3.0907	0.40235	7.6816	1.82E-14

Linear regression model: y ∼ 1 + x1 + x2 + x3

Table 12. Regression statistics.

Parameters	Dataset **1**	Dataset **4**
Number of observations	6180	6180
Error degrees of freedom	6176	6176
Root Mean Squared Error	7.19	8.79
R-squared	0.776	0.822
Adjusted R-Squared	0.775	0.821
F-statistic vs. constant model	7.11e + 03, p-value = 0	9.48e + 03, p-value = 0

6. Conclusion

This work has established an improved traffic flow prediction model using optimal DCNN, where the weights were chosen optimally. For the optimal selection of weights, a novel LN-TU model was established, which was an improved version of LA model. Finally, the performance of the presented model was evaluated over the conventional methods in terms of

prediction analysis and error analysis. More particularly, on considering the MAE, the adopted model for 5th dataset was 14.83%, 73.18%, 74.37%, 73.92%, and 1.88% better than SVM, RNN, DBN, NN, and LA models. Similarly, on considering the 17th dataset, the implemented model was 24.68%, 80.16%, 97.72%, 80.48%, and 8.61% better than SVM, RNN, DBN, NN and LA models. Moreover, the presented LN-TU approach at 32rd dataset was 35.06%, 84.18%, 83.96%, 83.92% and 1.69% better than SVM, RNN, DBN, NN and LA models. On observing the RMSE measure from Table 5, the adopted scheme at 12th dataset was 4.8%, 57.23%, 73.06%, 72.83% and 1.02% better than SVM, RNN, DBN, NN and LA models. Thus, the enhancement of the presented scheme has been validated effectively.

Disclosure statement

No potential conflict of interest was reported by the author(s).

References

Aramini, B., & Fan, J. (2019). Technique for myasthenia gravis: subxiphoid approach. *Thoracic Surgery Clinics*, *29*(2), 195–202. https://doi.org/10.1016/j.thorsurg.2018.12.010

Benocci, R., Molteni, A., Cambiaghi, M., Angelini, F., Roman, H. E., & Zambon, G. (2019). Reliability of Dynamap traffic noise prediction. *Applied Acoustics*, *156*, 142–150. https://doi.org/10.1016/j.apacoust.2019.07.004

Boothalingam, R. (2018). Optimization using lion algorithm: a biological inspiration from lion's social behavior. *Evolutionary Intelligence*, *11*(1–2), 31–52. https://doi.org/10.1007/s12065-018-0168-y

Candini, O., Grisendi, G., Foppiani, E. M., Brogli, M., Aramini, B., Masciale, V., Spano, C., Petrachi, T., Veronesi, E., Conte, P., Mari, G., & Dominici, M. (2019). A novel 3D in vitro platform for pre-clinical investigations in drug testing, gene therapy, and immuno-oncology. *Scientific Reports*, *9*(1), 7154– 7112. https://doi.org/10.1038/s41598-019-43613-9

Chen, W., An, J., Li, R., Fu, L., Xie, G., Bhuiyan, M. Z. A., & Li, K. (2018). A novel fuzzy deep-learning approach to traffic flow prediction with uncertain spatial–temporal data features. *Future Generation Computer Systems*, *89*, 78–88. https://doi.org/10.1016/j.future.2018.06.021

Cheng, A., Jiang, X., Li, Y., Zhang, C., & Zhu, H. (2017). Multiple sources and multiple measures based traffic flow prediction using the chaos theory and support vector regression method. *Physica A: Statistical Mechanics and Its Applications*, *466*, 422–434. https://doi.org/10.1016/j.physa.2016.09.041

Cun, Y. L., Kavukvuoglu, K., & Farabet, C. (2010). Convolutional networks and applications in vision. In *Circuits and Systems*, *International Symposium* (pp. 253–256).

Deng, S., Jia, S., & Chen, J. (2019). Exploring spatial–temporal relations via deep convolutional neural networks for traffic flow prediction with incomplete data. *Applied Soft Computing*, *78*, 712–721. https://doi.org/10.1016/j.asoc.2018.09.040

Do, L. N. N., Vu, H. L., Vo, B. Q., Liu, Z., & Phung, D. (2019). An effective spatial-temporal attention based neural network for traffic flow prediction. *Transportation Research Part C: Emerging Technologies*, *108*, 12–28. https://doi.org/10.1016/j.trc.2019.09.008

Emami, A., Sarvi, M., & Bagloee, S. A. (2020). Short-term traffic flow prediction based on faded memory Kalman Filter fusing data from connected vehicles and Bluetooth sensors. *Simulation Modelling Practice and Theory*, *102*, 102025.

George, A., & Rajakumar, B. R. (2013). APOGA: An adaptive population pool size based genetic algorithm. *AASRI procedia - 2013 AASRI conference on intelligent systems and control (ISC 2013)*, 4, pages: 288–296. https://doi.org/10.1016/j.aasri.2013.10.043

Guo, J., Huang, W., & Williams, B. M. (2014). Adaptive Kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification. *Transportation Research Part C: Emerging Technologies*, *43*(1), 50–64. https://doi.org/10.1016/j.trc.2014.02.006

Hou, Q., Leng, J., Ma, G., Liu, W., & Cheng, Y. (2019). An adaptive hybrid model for short-term urban traffic flow prediction. *Physica A: Statistical Mechanics and Its Applications*, *527*, 121065. https://www.tradingtechnologies.com/xtrader-help/x-study/technical-indicator-definitions/list-of-technical-indicators/. https://doi.org/10.1016/j.physa.2019.121065

Kong, F., Li, J., Jiang, B., & Song, H. (2019). Short-term traffic flow prediction in smart multimedia system for Internet of Vehicles based on deep belief network. *Future Generation Computer Systems*, *93*, 460–472. https://doi.org/10.1016/j.future.2018.10.052

Liang, X., Wang, G., Min, M. R., Qi, Y., & Han, Z. (2019). A deep spatio-temporal fuzzy neural network for passenger demand prediction. In *Proceedings of the 2019 SIAM international conference on data mining* (pp. 100–108), May. Society for Industrial and Applied Mathematics.

Li, F., Liao, S. S., & Cai, M. (2016). A new probability statistical model for traffic noise prediction on free flow roads and control flow roads. *Transportation Research Part D: Transport and Environment*, *49*, 313–322. https://doi.org/10.1016/j.trd.2016.10.019

Lin, Y., Zhang, J.-W., & Liu, H. (2019). Deep learning based short-term air traffic flow prediction considering temporal–spatial correlation. *Aerospace Science and Technology*, *93*Article, 105113. https://doi.org/10.1016/j.ast.2019.04.021

Li, L., Su, X., Wang, Y., Lin, Y., Li, Z., & Li, Y. (2015). Robust causal dependence mining in big data network and its application to traffic flow predictions. *Transportation Research Part C: Emerging Technologies*, *58*, 292–307. https://doi.org/10.1016/j.trc.2015.03.003

Li, W., Wang, J., Fan, R., Zhang, Y., Guo, Q., Siddique, C., & Ban, X. (J.). (2020). Short-term traffic state prediction from latent structures: Accuracy vs. *Transportation Research Part C: Emerging Technologies*, *111*, 72–90. https://doi.org/10.1016/j.trc.2019.12.007

Miglani, A., & Kumar, N. (2019). Deep learning models for traffic flow prediction in autonomous vehicles: A review, solutions, and challenges. *Vehicular Communications*, *20*Article, 100184. https://doi.org/10.1016/j.vehcom.2019.100184

Peng, H., Wang, H., Du, B., Bhuiyan, M. Z. A., Ma, H., Liu, J., Wang, L., Yang, Z., Du, L., Wang, S., & Yu, P. S. (2020). Spatial Temporal Incidence Dynamic Graph Neural Networks for Traffic Flow Forecasting. *Information Sciences*, *521*, 277–290. In press, journal pre-proof Available online 20. https://doi.org/10.1016/j.ins.2020.01.043

Peng, Y., & Xiang, W. (2020). Short-term traffic volume prediction using GA-BP based on wavelet denoising and phase space reconstruction. *Physica A: Statistical Mechanics and Its Applications*, *549*, 123913. In press, corrected proofAvailable online 30 December Article 123913. https://doi.org/10.1016/j.physa.2019.123913

Rahimipour, S., Moeinfar, R., & Hashemi, S. M. (2019). Traffic prediction using a self-adjusted evolutionary neural network. *Journal of Modern Transportation*, *27*(4), 306–316. https://doi.org/10.1007/s40534-018-0179-5

Rajakumar, B. R. (2013a). Static and adaptive mutation techniques for genetic algorithm: a systematic comparative analysis. *International Journal of Computational Science and Engineering*, *8*(2), 180–193. https://doi.org/10.1504/IJCSE.2013.053087

Rajakumar, B. R. (2013b). Impact of static and adaptive mutation techniques on the performance of Genetic Algorithm. *International Journal of Hybrid Intelligent Systems*, *10*(1), 11–22. https://doi.org/10.3233/HIS-120161

Rajakumar, B. R., & George, A. (2012). *A new adaptive mutation technique for genetic algorithm* [Paper presentation]. Coimbatore, India. In proceedings of IEEE International Conference on Computational Intelligence and Computing Research (ICCIC) (pp. 1–7, 18–20).

Rewadkar, D., & Doye, D. (2018). Traffic-aware routing protocol in VANET using adaptive autoregressive crow search algorithm. *Journal of Networking and Communication Systems*, *1*(1), 36–42.

Rewadkar, D., & Doye, D. (2019). Traffic-aware routing in urban VANET using PSO model. *Journal of Networking and Communication Systems*, *2*(2), 29–36.

Ryu, U., Wang, J., Kim, T., Kwak, S., & Juhyok, U. (2018). Construction of traffic state vector using mutual information for short-term traffic flow prediction. *Transportation Research Part C: Emerging Technologies*, *96*, 55–71. https://doi.org/10.1016/j.trc.2018.09.015

Shang, Q., Lin, C., Yang, Z., Bing, Q., & Zhou, X. (2016). Short-term traffic flow prediction model using particle swarm optimization–based combined kernel function-least squares support vector machine combined with chaos theory. *Advances in Mechanical Engineering*, *8*(8), 168781401666465. 4. https://doi.org/10.1177/1687814016664654

Sharma, B., Kumar, S., Tiwari, P., Yadav, P., & Nezhurina, M. I. (2018). ANN based short-term traffic flow forecasting in undivided two lane highway. *Journal of Big Data*, *5*(1), 48. https://doi.org/10.1186/s40537-018-0157-0

Swamy, S. M., Rajakumar, B. R., & Valarmathi, I. R. (2013). *Design of hybrid wind and photovoltaic power system using opposition-based genetic algorithm with Cauchy mutation* [Paper presentation]. IET Chennai Fourth International Conference on Sustainable Energy and Intelligent Systems (SEISCON 2013), Chennai, India, December. https://doi.org/10.1049/ic.0361[10.1049/ic.2013.0361]

Taghipour, A., & Frayret, J. M. (2010). Negotiation-based coordination in supply chain: model and discussion. In 2010 *IEEE international conference on systems, man and cybernetics* (pp. 1643–1649). IEEE.

Tang, Y. (2019). Traffic flow prediction using convolutional neural network accelerated by spark distributed cluster.

Tang, J., Chen, X., Hu, Z., Zong, F., Han, C., & Li, L. (2019). Traffic flow prediction based on combination of support vector machine and data denoising schemes. *Physica A: Statistical Mechanics and Its Applications*, *534*, 120642. https://doi.org/10.1016/j.physa.2019.03.007

Tian, Y., Zhang, K., Li, J., Lin, X., & Yang, B. (2018). LSTM-based traffic flow prediction with missing data. *Neurocomputing*, *318*, 297–305. https://doi.org/10.1016/j.neucom.2018.08.067

Vosooghidizaji, M., Taghipour, A., & Canel-Depitre, B. (2020). Supply chain coordination under information asymmetry: A review. *International Journal of Production Research*, *58*(6), 1805–1834. https://doi.org/10.1080/00207543.2019.1685702

Wang, W., Zhang, H., Li, T., Guo, J., Huang, W., Wei, Y., & Cao, J. (2020). An interpretable model for short term traffic flow prediction. *Mathematics and Computers in Simulation*, *171*, 264–278. https://doi.org/10.1016/j.matcom.2019.12.013

Wu, Y., Tan, H., Qin, L., Ran, B., & Jiang, Z. (2018). A hybrid deep learning based traffic flow prediction method and its understanding. *Transportation Research Part C: Emerging Technologies*, *90*, 166–180. https://doi.org/10.1016/j.trc.2018.03.001

Yang, H.-J., & Hu, X. (2016). Wavelet neural network with improved genetic algorithm for traffic flow time series prediction. *Optik*, *127*(19), 8103–8110. https://doi.org/10.1016/j.ijleo.2016.06.017

Yang, B., Sun, S., Li, J., Lin, X., & Tian, Y. (2019). Traffic flow prediction using LSTM with feature enhancement. *Neurocomputing*, *332*, 320–327. https://doi.org/10.1016/j.neucom.2018.12.016

Zhao, L., Zhou, Y., Lu, H., & Fujita, H. (2019). Parallel computing method of deep belief networks and its application to traffic flow prediction. *Knowledge-Based Systems*, *163*, 972–987. https://doi.org/10.1016/j.knosys.2018.10.025