COMPUTER-AIDED CIVIL AND INFRASTRUCTURE ENGINEERING

Computer-Aided Civil and Infrastructure Engineering **33** (2018) 999–1016

Short-Term Traffic Speed Forecasting Based on Attention Convolutional Neural Network for Arterials

Qingchao Liu*

Automotive Engineering Research Institute, Jiangsu University, Zhenjiang, P. R. China

Bochen Wang & Yuquan Zhu

School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, P. R. China

Abstract: *As an important part of the intelligent transportation system (ITS), short-term traffic prediction has become a hot research topic in the field of traffic engineering. In recent years, with the emergence of rich traffic data and the development of deep learning technologies, neural networks have been widely used in short-term traffic forecasting. Among them, the Recurrent Neural Networks (RNN), especially the Long Short-Term Memory network (LSTM) shows the excellent ability of time-series tasks. To improve the prediction accuracy of the LSTM, some research uses the spatial–temporal matrix or Convolutional Neural Network (CNN) to extract the spatial features of the data for the LSTM network to use. In this article, we propose an attention CNN to predict traffic speed. The model uses three-dimensional data matrices constructed by traffic flow, speed, and occupancy. The spatial–temporal features extraction and the attention models are all performed by the convolution unit. Experiments on traffic data at 15-minute intervals show that the proposed algorithm has considerable advantages in predicting tasks compared to other commonly used algorithms, and the proposed algorithm has an improvement effect for cases with missing data. At the same time, by visualizing the weights generated by the attention model, we can see the influence of different spatial–temporal data on the forecasting task.*

1 INTRODUCTION

With the development of the economy, the number of vehicles kept by urban residents keeps increasing, and the load of the road network is constantly on the rise. To solve the traffic problems caused by high load, an intelligent transportation system (ITS) was put forward. One of the most important and challenging parts is traffic prediction, which is also the basis for implementing other advanced capabilities. Through traffic forecasting, the ITS can predict potential road congestion and perform dynamic traffic control, traffic guidance, and other functions. This will help improve the efficiency of road traffic and promote the intelligent upgrade of road traffic networks.

The target of short-term traffic forecasting is to predict basic variables such as speed, occupancy, and other indicators of the future traffic status at the location of the sensor, usually in the range of 5–30 minutes. Traffic data have a very complex relationship in space and time. Apart from the complex time-series features, it is also influenced by a number of other factors, such as geographical location, changes in traffic hours, and incidents. These factors lead to unstable data changes, making the tasks very challenging.

Due to the importance of traffic forecasting tasks, relevant studies have been continuously carried out since the 1970s. They are basically divided into two categories: parametric models and nonparametric models. The parametric model is represented by the autoregressive integrated moving average (ARIMA) (Ahmed and Cook, 1979) and is widely used in traffic prediction (Levin and Tsao, 1980; Hamed et al., 1995). ARIMA also extends many variant models to improve the accuracy of predictive tasks, such as seasonal ARIMA (Williams and Hoel, 2003), Kohonen-ARIMA (Mascha et al., 1996), and ARIMA with the Kalman filter (Lippi et al., 2013). In addition, Jiang and Adeli (2004) proposed a statistical autocorrelation function (ACF) for the selection of the decomposition level in wavelet

*To whom correspondence should be addressed. E-mail: *lqc@ujs.edu.cn*.

DOI: 10.1111/mice.12417

multiresolution analysis of traffic flow. The Markov chain (Qi and Ishak, 2014) and Bayesian network (Ghosh et al., 2010; Wang et al., 2014) are also used for traffic forecasting tasks. However, some parametric models require stable and accurate traffic data, whereas the actual traffic data are unstable and nonlinear. Therefore, these models appear to be inadequate for traffic data that have complex nonlinear structure. In recent years, with the development of machine learning technology, a large number of nonparametric models have been used to handle traffic tasks, such as the *k*-nearest neighbor algorithm (KNN) (Habtemichael and Cetin, 2016), support vector regression (SVR) (Jeong et al., 2013), *K*-means (García-Ródenas et al., 2017), and artificial neural network (ANN) models (Chan et al., 2012; Kumar et al., 2013). Among them, the neural network model shows better performance.

In recent years, with the rapid development of deep learning technologies, related deep models have been widely used in engineering and other fields, such as structural damage detection (Lin et al., 2017; Zhang et al., 2017), health condition assessment of buildings (Rafiei and Adeli, 2017, 2018), performance estimation (Rafiei and Adeli, 2017). In addition, many studies have been devoted to the optimization of algorithms (Koziarski and Cyganek, 2017) and models (Ortega-Zamorano et al., 2017), making deep neural networks more applicable to different scenarios. Owing to the massive increase in traffic data, the deep neural network model can better fit the nonlinear features of traffic data and achieve better prediction results through the analysis of massive real-time and historical traffic data. Therefore, a variety of neural networks are widely used in traffic management and control applications. The neural network model was used to predict Link Travel Time (Dharia and Adeli, 2003), and the radial basis function neural network (RBFNN) was used in the classification of noise and cluster observation data in traffic accident detection (Adeli and Karim, 2000). A method combining of wavelet-based signal processing, statistical clustering analysis, and a neural network was used for highway incident detection (Ghosh-Dastidar and Adeli, 2003). A computational model combining discrete wavelet transforms, linear discriminant analysis, and neural networks was used for automatic traffic incident detection (Adeli and Samant, 2000). In the field of traffic prediction, Dynamic wavelet neural network was used for traffic flow forecasting (Jiang and Adeli, 2005). A self-organizing fuzzy neural network was used to predict short-term traffic flow based on time series (Boto-Giralda et al., 2010). A deep belief network (DBN) was used to predict traffic flow and capture its nonlinear characteristics well (Huang et al., 2014). The stacked autoencoder (SAEs) model was used to predict short-term traffic flow and achieve better results than parametric models such as ARIMA (Lv et al., 2015). However, the fully connected network has some difficulties in extracting spatial–temporal features and the structure of its neurons cannot fit this requirement. Recurrent Neural Networks (RNN), especially Long Short-Term Memory networks (LSTM) (Hochreiter and Schmidhuber, 1997), can successfully characterize the time correlation and show excellent ability with regard to time-series tasks. The LSTM network was used to make traffic speed forecasts, showing better performance than ARIMA and SAE (Ma et al., 2015). The Mixture Deep LSTM network was used to forecast traffic during peak hours and to further refine the postproduction forecasting model to simulate normal traffic conditions and incident patterns (Yu et al., 2017).

To further improve the prediction accuracy of applied models, some researchers use the spatial features in the traffic forecast task. The method using spatial–temporal data can preserve the estimability of the road network model while providing a complete description of the spatial–temporal interaction (Min and Wynter, 2011). The spatial–temporal random effects (STRE) model was used to make traffic forecasts. It can reduce computational complexity and can flexibly consider the pattern of traffic (Wu and Tan, 2016). A simple mapping-to-cells method was used to construct a spatial–temporal traffic diagram for a freeway network (He et al., 2017). The Support Vector Machine (SVM) model was used to predict short-term traffic speed based on spatial–temporal data (Yao et al., 2015). A neural predictor composed of time-optimized Multi-Layer Perceptron (MLP) structures can use spatial–temporal data to provide accurate short-term traffic flow prediction (Vlahogianni et al., 2007). Although LSTM networks have good data processing capabilities, simple LSTM networks do not make good use of the spatial features of data. To solve this problem, the cascaded LSTM structure was developed, as it allows the model to use spatial–temporal data to predict traffic flow (Zhao et al., 2017). Owing to the good performance of the Convolutional Neural Network (CNN) in the field of image processing (Cha et al., 2017), some researchers use CNN to capture spatial features during traffic forecasting task. The CNN-LSTM model can effectively improve the accuracy of prediction tasks (Xingjian et al., 2015). CNN-LSTM based short-term Traffic Flow Prediction method (CLTFP) uses 1D CNN on the road to capture the spatial features of volume and then combines them with the temporal features captured by LSTM (Wu and Tan, 2016). DeepST uses a deep spatial–temporal residual network for crowd flows prediction within the city grid area (Zhang et al., 2017). DLM8L (deep learning combined with the

median filter preprocessing) uses 2D CNN to extract spatial–temporal features and predict traffic speed in highway (Polson and Sokolov, 2017).

In general, many studies on traffic forecasting consider the spatial–temporal relationship of traffic data, and the relevant models tend to be complicated. However, neural network research faces several challenges: (1) the LSTM network is generally used for temporal features extraction, but an LSTM unit has the disadvantage of slow computing speed and cannot be trained in parallel; (2) the traffic data in different temporal locations and spatial locations have different effects on the prediction tasks and one cannot treat them equally; and (3) missing data are a very common situation in traffic forecasting tasks. The lack of a single attribute has a significant impact on the temporal forecasting task. To address these challenges, we introduced the Gated CNN and attention models in the prediction task. Furthermore, we also try to fuse different data to reduce the impact of missing data in the prediction task.

In this article, we propose a short-term traffic speed prediction model based on the attention CNN. Our reason for forecasting speed is that the range of traffic speed is stable and can also reflect the traffic conditions of the road, whereas the range of traffic flow depends on the capacity of different roads. The model uses general CNN to extract the spatial features of traffic data and Gated CNN to extract the temporal features of traffic data. To enhance the validity of the features, the attention model is used to weight the feature maps and the channels, respectively. Finally, the processed traffic features are used to predict the traffic speed of multiple detectors.

The remainder of the article is organized as follows. Section 2 introduces the theory of the CNN model. Section 3 shows the methods used in our system. Section 4 shows the experimental data and results. Finally, Section 5 concludes the article and looks forward to the future work.

2 CNN MODEL

The CNN is a type of feed-forward ANN in which the connectivity pattern between neurons is inspired by the organization of the animal visual cortex. Individual cortical neurons respond to stimuli in a restricted region of space known as the receptive field. The receptive fields of different neurons partially overlap such that they tile the visual field. The response of an individual neuron to stimuli within its receptive field can be approximated mathematically by a convolution operation. CNN was inspired by biological processes and was widely used for large-scale image processing (LeCun et al., 1998). The basic structure of CNN is shown in Figure 1.

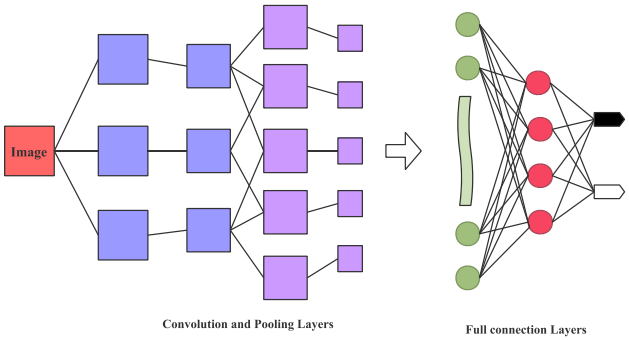

Fig. 1. Structure of CNN.

Assume a set of training samples $\{x_i,\ y_i\}_{i=1}^{n}$, for which the input data x_i are in the form of a matrix set and $x_i \in R^{H\times W\times D}$, the output data $y_i \in R$ (H, W, D represent the dimensions of the input matrix). The input data are convoluted and processed to get the original convolution features by Equation (1):

$$e_i = Conv(x_i) = \sum_{j=1}^{k} w_j{}^{*} x_i + b_j \tag{1}$$

where k is the number of convolution kernels, e_i represents the feature maps obtained by convolution of the input x_i, w_j is the weight set of jth convolution kernel, and b_j is the jth bias. After that, the feature maps e_i need to be activated to get the feature maps f_i, which are passed to the next layer, as given in Equation (2). σ is the activation function; it can enhance the expression ability of the entire network.

$$f_i = \sigma(e_i) \tag{2}$$

The loss function for the prediction task is defined as follows:

$$L = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 \tag{3}$$

Finally, the processed features of the multilayer convolutional network are sent to the full connection layer for the prediction task. The error between the predicted result and the observed result is calculated by the loss function L as shown in Equation (3), where $\hat{y}$ is the predicted data and y is the real data. The errors can be used to update the parameters of the convolutional network during training time through the backpropagation algorithm (LeCun et al., 1990). After certain epochs of training, the final model can converge toward the goal.

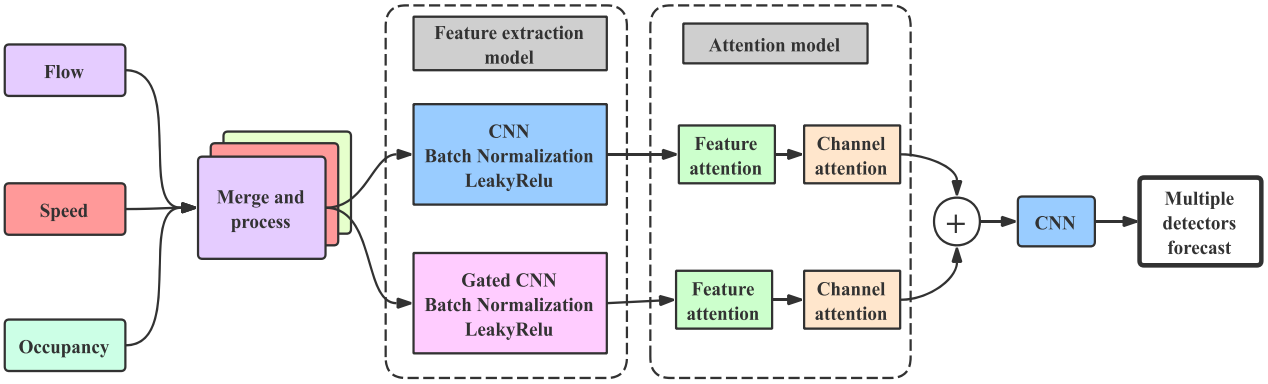

Fig. 2. Overview of the proposed approach. The input data are processed and passed through a CNN-based spatial feature extraction model and the Gated CNN-based temporal feature extraction model, respectively. Then, all the features are weighted by attention model (the feature extraction model and the attention model can be stacked several times and only one layer is shown here). Finally, we fused these features by using an elementwise add operation and then predicted the traffic speed.

3 METHODOLOGY

In a road network, detectors do not exist in isolation. The traffic speed of the upstream and downstream sections will affect the speed of the current section. In general, the data detected by each detector can roughly represent the traffic conditions in its area. Therefore, understanding spatial and temporal information can help to better identify the traffic condition of the road. Figure 2 presents the proposed road traffic speed forecast model. We used two branches to extract the spatial features and temporal features, and then used an attention model to weigh the hidden features. By stacking several convolutional layers, we can attain deep features of traffic flow. Traffic speed is predicted by the use of fusion features.

3.1 Data structure

There are interactions among these three variables of volume, speed, and density in traffic engineering theory (Roess et al., 2004; Ou et al., 2017). Figure 3 illustrates the general form of these relationships. If some data in the data set are missing, we can infer the missing data through the relevant attributes, which is beneficial to the robustness of our model.

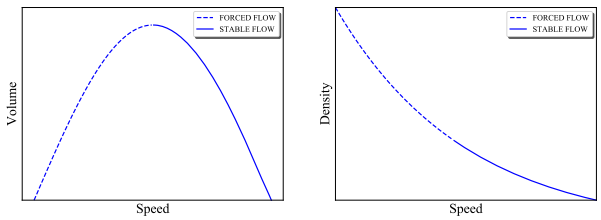

Fig. 3. Relationships among volume, speed, and density.

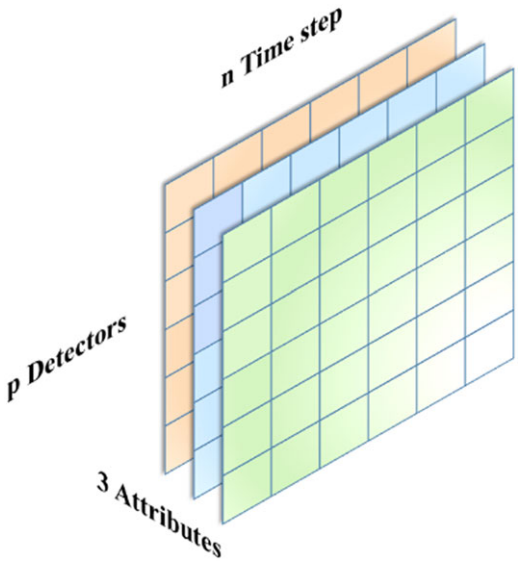

Fig. 4. Structure of input data.

We used historical traffic data to build a 3D matrix with spatial–temporal relationships as an input to CNN, as shown in Figure 4. Assuming that there are total of p detectors $d_i = \{d_1, d_2, \ldots, d_p\}$ is arranged on the road section in spatial order. The detector d_i can provide a set of temporal detection data $dr_i = \{r_{t-n}, \ldots, r_{t-1}, r_t\}$, where the detection data recorded at each time point t are $r_t^i = \{v,\ s,\ o\}$. v is that of the volume, s is the speed, and o is the occupancy. Our task is to predict the traffic speed of d_i at time t, for which we need historical data for d_i at times $\{t-n,\ \ldots,\ t-2,\ t-1\}$, that is, $dr_i = \{r_{t-n}, \ldots, r_{t-2}, r_{t-1}\}$. We combined all the data

according to the spatial and temporal distribution of the detection locations on the road to get a three-dimensional data matrix of size $(p, h, 3)$, h is the length of history data in time dimension. We divide the matrix according to the specified step n in the time direction of the matrix and get the set of matrixes $M = \{m_1, m_2, \ldots, m_q\}$. Where $q \in R$ is the number of elements in the set, the size of each element m_q is $(p, n+1, 3)$. A sample element in the set is shown by Equation (4).

$$m_q = \begin{bmatrix} dr_1 \\ dr_2 \\ . \\ . \\ dr_p \end{bmatrix} = \begin{bmatrix} r^1_{t-n} & r^1_{t-n+1} & \ldots & r^1_t \\ r^2_{t-n} & r^2_{t-n+1} & \ldots & r^2_t \\ . & . & & . \\ . & . & & . \\ r^p_{t-n} & r^p_{t-n+1} & \ldots & r^p_t \end{bmatrix} \tag{4}$$

3.2 Spatial feature extraction

In spatial feature extraction, the convolution operation in CLTFP and DeepST is only used in the spatial dimension of the current time, and the convolution operation in DLM8L mixes the spatial features of different time for speed forecast. In our model, we refer to the DLM8L model and have used data fusion for various traffic data attributes.

The input traffic data matrix m_q has the spatial correlation in the column, the temporal correlation in the row, and the correlation between the attributes in the channel. This data matrix has very similar data structures to RGB image data, so we use normal 2D convolution layers to extract its spatially and property-mixed features. We used the convolution kernel of size (M, N) for convolution operation to get the output spatial features of m_q, as shown in Equation (5):

$$f_i = \sigma\left(BN\left(Conv\left(m_q\right)\right)\right) \tag{5}$$

where f_i is the ith feature map of m_q, and $F_s = \{f_1, f_2, \ldots, f_k\}$ is the output hidden spatial feature maps of the convolutional layer, $k \in R$ is the number of the convolution kernel. BN represents the Batch Normalization function (Ioffe and Szegedy, 2015). σ represents the nonlinear activation function, and here we use the LeakyRelu function (Maas et al., 2013). This structure can be stacked to extract deep features.

3.3 Temporal feature extraction

In many traffic prediction studies based on deep learning, the RNN network is usually used to extract the temporal features, such as LSTM and Gated Recurrent Unit (GRU). However, these RNN units also have the disadvantage of being slow to operate and unable to train in parallel. To overcome these problems and cooperate with the convolutional layer of the spatial features extraction, we introduced the Gated CNN structure (Dauphin et al., 2016) instead of RNN units. The Gated CNN model is used to replace the LSTM model in natural language processing problems and shows a strong computing performance (Gehring et al., 2017).

In the task of natural language processing, the probability of the occurrence of a certain word in the future is usually related to the preceding words, and the actual meaning of a certain sentence or its context. For traffic speed forecasting tasks, the speed is usually related to its historical speed. We can estimate future situations through evaluating the speed trends of the past. The Gated CNN model is very suitable for extracting temporal features, considering that the sequence of words in sentences is similar to that of the general time-series data. Therefore, we can extract more high-level and abstract temporal features of traffic flow data through a Gated CNN approach.

Different from normal CNN, Gated CNN is divided into two parts here, the convolution unit and the gate value convolution unit. We process the matrix m_q using two different ordinary convolutional layers to obtain the convolution activation values $A = \{a_1, a_2, \ldots, a_k\}$ and the gate values $B = \{b_1, b_2, \ldots, b_k\}$ through Equation (6). The gate values, B, pass through a nonlinear activation function σ, where the sigmoid function is used here. This is defined as Gated Linear Units (GLU). The output hidden feature maps $F_t = \{f_1, f_2, \ldots, f_k\}$ are obtained by elementwise product operation betweent A and B, as Equation (7) shows:

$$a_i = Conv\left(m_q\right), b_i = Conv\left(m_q\right) \tag{6}$$

$$f_i = A \odot \sigma\left(B\right) \tag{7}$$

Specifically, here we use a convolution kernel of size $(N, 1)$ for temporal convolution. In addition, we also need to maintain the same size for the input and output data, so the padding operation is used during convolution. After that, we use Batch Normalization and LeakyRelu functions to process F_t. The structure can then be stacked to capture long-term memory.

3.4 Feature maps attention

Variables at different temporal and spatial locations have a different impact on future predicting tasks. However, all feature variables in the same feature map are considered to have the same weight by default. Due to the common use of the attention model in natural language processing (Yin et al., 2015), we introduced it into the traffic forecast task. In this model, we used spatial–temporal features for forecasting tasks and weighted the features on the different spatial–temporal position. In

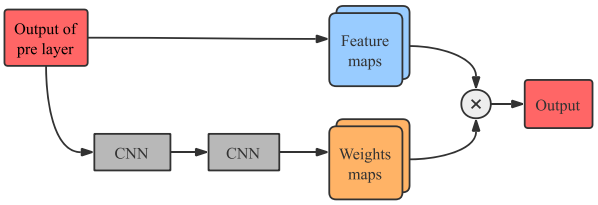

Fig. 5. Structure of feature maps attention.

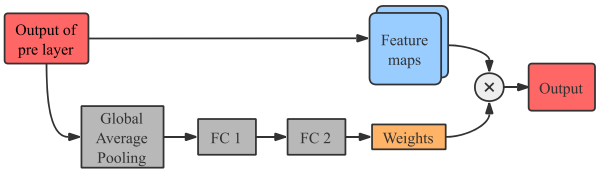

Fig. 6. Structure of channels attention.

the task of natural language processing, the weight matrix is calculated using the Euclidean distance between the input sentences, and our model uses two convolution layers to automatically learn the weight matrix. The structure of feature maps attention model is shown in Figure 5.

For the input feature maps $F = \{f_1, f_2, \ldots, f_k\}$, we can generate a set of weights matrix $V_m = \{v_1, v_2, \ldots, v_k\}$ whose size is the same as the feature maps. The weight v_i is calculated by Equation (8). The weighted feature maps F' are calculated by Equation (9).

$$v_i = \frac{\exp(f_i)}{\sum_{t=1}^{k} \exp(f_t)} \tag{8}$$

$$F' = V_m \odot F \tag{9}$$

where k is the number of channels in the output feature maps, the attention model that generates V_m consists of two layers (the first layer has $k \times s$ filters with convolution kernel size 3×3 and the second layer has k filters with convolution kernel size 1×1, $s \in R$). $\odot$ is the elementwise product.

3.5 Channels attention

The data matrix for the input model contains three different attributes, and their impact on predictive tasks is different. At the same time, the different channel outputs from the convolutional layer do not necessarily have the same effect on the model. Therefore, we weight the different channels to improve the effectiveness of the features on the prediction task. The structure of channels attention model is shown in Figure 6.

For the input feature maps $F = \{f_1, f_2, \ldots, f_k\}$, where k is the number of channels in the feature maps, we performed a global average pooling operation in F to get the mean for each channel $C = \{c_1, c_2, \ldots, c_k\}$ by using the Equation (10). Then the vector C is entered into two fully connected layers of length k, and the last layer is activated by the sigmoid function. Through the above operation, we can obtain the weight of each channel $V_c = \{v_1, v_2, \ldots, v_k\}$. The weighted feature maps F' are calculated by using Equation (11).

$$c_i = \frac{1}{n} \sum_{j=1}^{n} f_{i,j} \tag{10}$$

$$F' = V_c{}^* F \tag{11}$$

$f_{i,j}$ is the jth value in the ith feature map (n is the number of elements in the feature map, $i, j \in R$).

3.6 Forecast of the speed of multiple detectors

The input matrix data used in the task was collected from p detectors on the road. Therefore, we can achieve the forecast for all detectors in a single calculation of the model. After we attained the weighted temporal features and spatial features, the two feature maps can be elementwise added by using Equation (12) because we padded the boundary during the convolution operation. Then we convolute F_{st} through global convolution with k convolution kernel of $h \times w$ (h and w are the height and width of the front layer convolution output) and compress them into a one-dimensional feature. Finally, we used p convolution kernels size 1×1 to get the forecast result of p detectors.

$$F_{st} = F'_s \oplus F'_t \tag{12}$$

The choice of loss function on the output layer is tightly coupled with the choice of the output unit. We simply use mean square error to fit the model to predict the future speed of the different detectors. Due to the merging of features by a convolution operation, multiple regression tasks affect each other and constrain the parameters and ensure the validity of multitask regression. In the forecasting process, we used convolution instead of the traditional fully connected layer operation. This makes the entire network constitute a fully convolutional network, reducing the model parameters and increasing the speed of the model.

Table 1 shows the process of the training of the proposed algorithm. The modules we presented above are all trained together in the network. Although the network looks complicated, our model is essentially a set of weight parameters P. The distribution of P is determined after the training procedure is completed. We can abstract P into a function $f(x)$ that fits the rules we specify in the data set. When using this model, we do not need to adjust any parameters other than the input.

Table 1
The process of model training

Input:
volume v in training set
speed s in training set
occupancy o in training set
Time lag n
Output:
Model parameters P
(1) **Data process**
(2) Process data.
(3) Concatenate all $\{v,\ s,\ o\}$ as m by Equation (4)
(4) **for** all available time t **do**
(5) $m_t = \mathrm{m}\,[t-n, \ldots,\ t-2, t-1]$
(6) A training observation m_t is put to M
(7) **end for**
(8) **end data process**
(9) **Training Algorithm**
(10) Initialize a null learnt parameter set: P
(11) **repeat**
(12) Randomly extract a batch of samples M_b from M_{train}
(13) Get spatial features F_s by Equation (5)
(14) Get temporal features F_t by Equation (7)
(15) Get weighted F_s' and F_t' by Equations (9) and (11)
(16) Fusion features by Equation (12)
(17) Speed prediction using logistic regression
(18) Estimate the parameters P by loss in Equation (3)
(19) **until** convergence criterion met in M_{test}
(20) **end training**

4 EXPERIMENT

We used the experimental platform hardware configuration for the Intel Core i5 6500 3.2 GHz, GTX 650 and DDR3 6G memory. The platform software is configured for Ubuntu Linux 16.04 operating system. The experimental model was built using Python 3.5, Keras 2, and Tensorflow 1.0.0.

4.1 Data description and processing

Our experiments used traffic data collected from the Caltrans Performance Measurement System (PeMS). PeMS provides a consolidated database of traffic data collected by Caltrans placed on state highways throughout California, as well as other Caltrans and partner agency data sets. The data were collected at U.S. Route I5 from December 1, 2015 to December 31, 2015 by 12 loop detectors for a 15-minute interval, as shown in Figure 7. We utilize the data of 26 days a month as the training set (29,952 items) and 5 days as the test set (5,760 items). The aggregated average speed at 96 time points of a day, the aggregated average daily traffic speed of a week, and the average traffic speed of the locations at those 12 detectors are given in Figure 8. As can be seen from the Figure 8, different time periods have different average speed. During a whole day, the working time period has a lower speed. The distribution of average speeds during the week also varies. Figure 9 shows three scatter plots of the volume–speed for each lane in the study area. It can be seen that there is a correlation between flow and speed, and the relationship in real data is basically the same as that we mentioned in Figure 3.

In the process of collecting data, there was a lot of noise in the data set due to the detector's failure or data loss, which has a negative impact on our forecasting task. This noise is randomly distributed in the data set. Some detectors are missing the data of an attribute, and some detectors fail in a certain period of time. To solve this problem, we used KNN to fill the null attribute value and use historical data in the same time period for the invalid detector. Figure 10 shows a sample for filling a single missing attribute with KNN. Then we use the locally estimated scatterplot smoothing (LOWESS) algorithm to smooth the raw data. The processed data are smoother and more continuous than the raw data, which not only helps to train the network but also improves the network's ability to generalize unknown data. It should be noted that we only smoothed the data in the training set, and we use the original data in the test. Figure 11 shows the original traffic speed data and the smoothed data of a day (span = 0.05).

To prevent inconsistencies in the data distribution and gradient explosions, we need to process the input data by scaling the attribute to a specified maximum and minimum value (usually 1–0) by min-max normalization. For each attribute in the data set, we use Equation (13) to obtain the scaled values, where X is a set of attribute values of the data we want to process, and X_{scale} is a scaled set of values.

$$X_{\text{scale}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{13}$$

4.2 Evaluation criteria

Three indicators are commonly used to evaluate the performance of traffic prediction models. They are the mean absolute error (MAE), the root mean square error, also known as standard error (RMSE) and the mean absolute percentage error (MAPE), defined as Equations (14)–(16):

$$MAE\,(y, \hat{y}) = \frac{1}{n_{\text{sample}}} \sum_{i=0}^{n_{\text{sample}}-1} |y_i - \hat{y}_i| \tag{14}$$

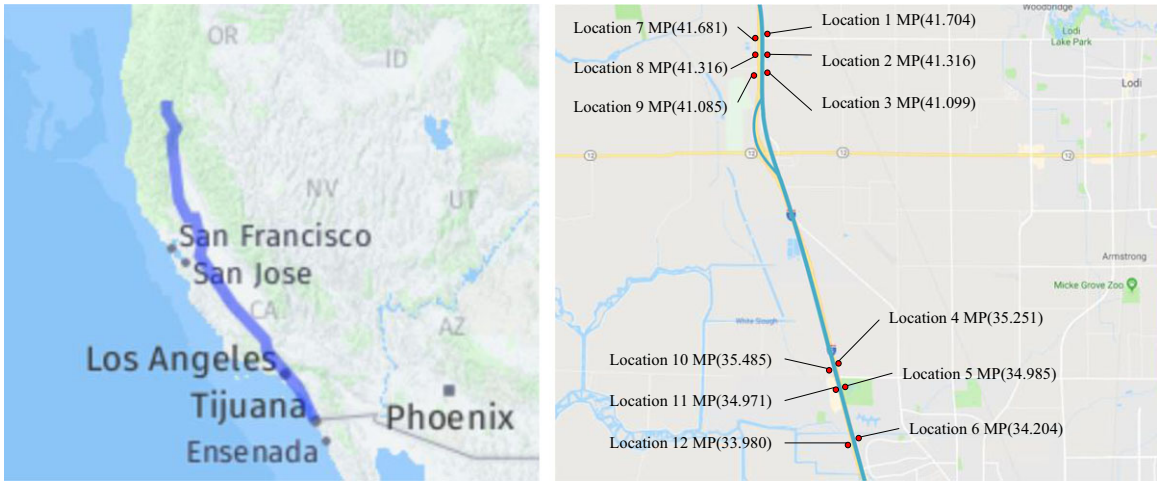

Fig. 7. Spatial locations of the research targets.

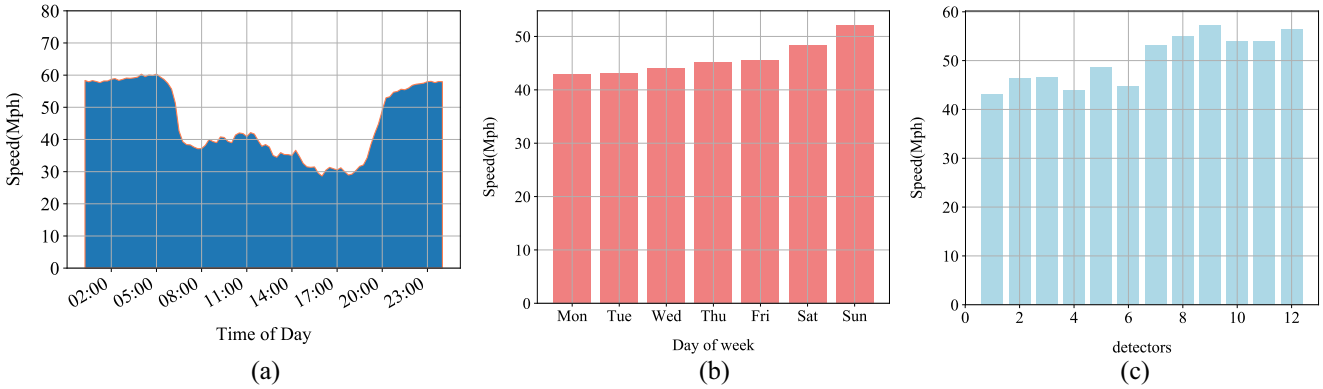

Fig. 8. The average speed: (a) 96 time points of a day, (b) different days in a week, (c) the locations of those 12 detectors.

$$RMSE(y, \hat{y}) = \sqrt{\frac{1}{n_{\text{sample}}} \sum_{i=0}^{n_{\text{sample}}-1} (y_i - \hat{y}_i)^2} \quad (15)$$

$$MAPE(y, \hat{y}) = \frac{100}{n_{\text{sample}}} \sum_{i=0}^{n_{\text{sample}}-1} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (16)$$

where $\hat{y}$ is the predicted value, y is the observed value, and n_{sample} is the size of the test set. MAE is the evaluation indicator for regression task. Compared to MAE, RMSE is more sensitive to outliers and can amplify larger prediction bias values. It is usually used to compare the stability of different prediction models. MAPE provides the forecast error in terms of the percentage difference between the observed and predicted traffic speed, as a measure of prediction accuracy of a forecasting method in statistics. These measures of performance provide a deep understanding of the nature of the prediction errors.

4.3 Parameter determination

Choosing the best lag time is very important to minimize the prediction error of the model. The lag time affects the performance of the model in forecasting traffic speed, as it is a major variable for learning the similar speed pattern. In general, a shorter lag time is suitable for short-term traffic forecasting, whereas a relatively long time is suitable for long-term traffic forecasting. We chose a time lag between 15 minutes and 2 hours. Figure 12 shows the effect of lag period on the prediction accuracy, with each lag period in 15 minutes. As shown in Figure 12, when the time lag is 3, our model can get the best performance in MAE, RMSE, and MAPE. Therefore, a 45-minute lag is considered the most appropriate in our experiment.

In addition to the best lag time, we also needed to select the appropriate parameter for the model. There are two types of parameters in the deep neural network model: one is the elementary parameter, such

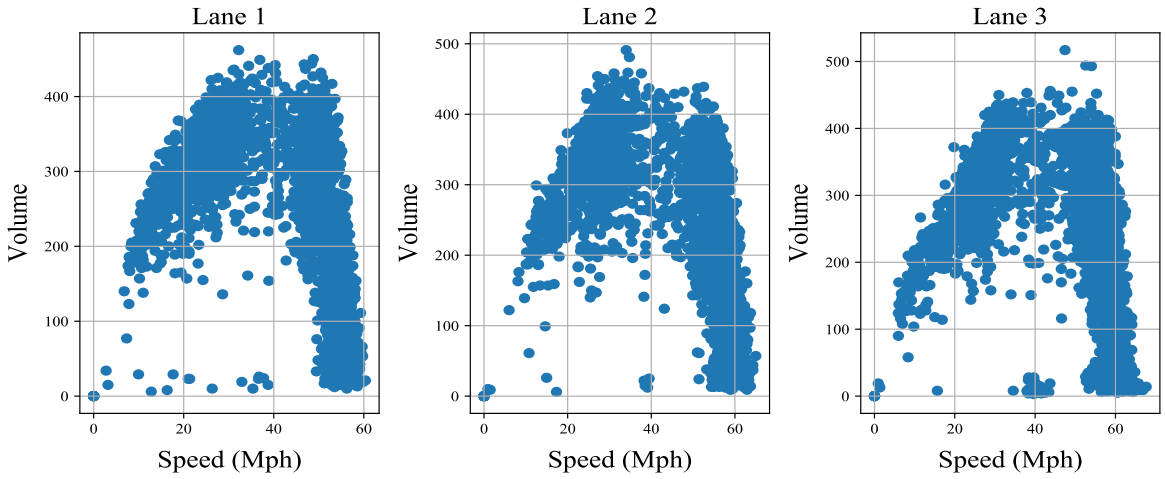

Fig. 9. Volume and speed of each lane.

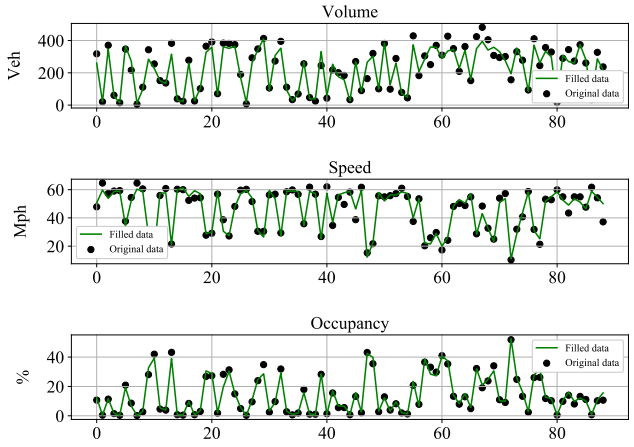

Fig. 10. A sample of filling single missing attribute with KNN.

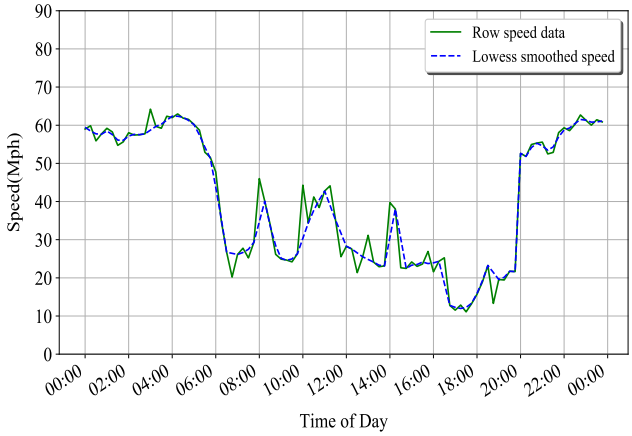

Fig. 11. A sample of raw and LOWESS smoothed traffic speed data.

as the weight and bias of convolutional layer or fully connected layer. The other is hyper parameter, such as learning rate during network training and coefficient of L2 regularization term in loss function.

In practical applications, deep neural networks wanting to get good performance is very dependent on the selection of a good set of parameters. In our case, we used two feature extraction modules and two attention

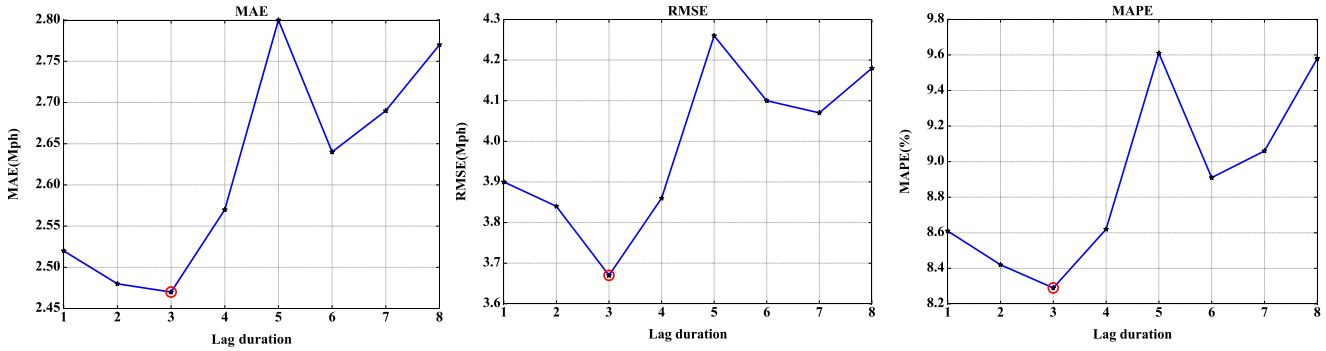

Fig. 12. Impact of lag duration on forecast error.

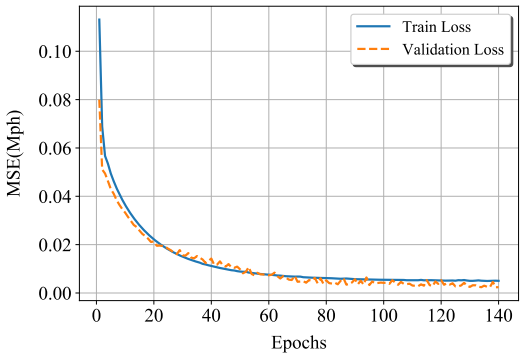

Fig. 13. Loss and the number of training epochs.

modules. The number of filters used by the feature extraction module was (16, 32) and the number of filters used by feature maps attention is (32, 16, 64, 32), which means we set s to 2 in feature maps attention. The kernel size of spatial CNN is 2×2 and 3×1 for temporal CNN. The model was trained using the Adam optimizer with a learning rate of 0.001, L2 is 5e−4, batch size is 256, and max epoch is 600. The training process was terminated when the loss did not decline and validation accuracy did not change. Training loss and validation loss during training epochs are shown in Figure 13.

4.4 Experiment analysis

To evaluate the performance of the proposed method, the proposed models are compared with the algorithms of ARIMA, LSVR (Linear Support Vector Regression), KNN, SAEs, LSTM, and DLM8L, respectively. Since in many studies, the deep learning model performs better than the traditional model (i.e., ARIMA), we focus on comparing deep learning-based models. The (p, d, q) parameters of the ARIMA algorithm use (4, 0, 4). The LSVR kernel uses the linear function, penalty parameter is 1.0, and tolerance for stopping criterion is 1e−3. The KNN algorithm uses five neighbors and the weight function is uniform. The above three algorithms use one-dimensional temporal data to train and run the prediction test at every test detector point. For deep neural network-based algorithms, SAEs adopt a three-layer structure with (400, 400, 400) units in the hidden layer. The LSTM uses a single-layer structure with 64 units in the hidden layer. The DLM8L uses three hidden layers with (32, 64, 128). The above three methods use spatial–temporal data for training, and all the test points are tested at one time.

Table 2 shows the performance of our proposed method and the method of comparison in predicting traffic speed over the entire data set. The tests for the next 15 minutes and 30 minutes were evaluated separately. The error was calculated using the average error for all detectors.

When the 15-minute interval is used for prediction, the proposed method can achieve an average MAPE of 7.35% and its MAE and RMSE also perform better than the alternate method. When the 30-minute interval is used for prediction, the performance is still better than that of the comparative method, even if the error increases. According to the prediction results, the proposed method is effective for the prediction of short-term traffic speed in total data set.

Figure 14 shows the comparison of the prediction errors produced by different methods. Due to the complexity and noise contained in the real data, even the data preprocessing cannot completely remove its influence, so the model will produce anomalies when making predictions. It can be seen from the figure that among the three error indexes, the proposed method has the lowest outlier value compared with other methods. This shows that the proposed method can more effectively reduce the impact of outliers. In addition, the proposed method also demonstrates better performance than other methods in terms of the maximum, minimum, and the median of errors, and the error distribution of the proposed method is more concentrated in the distribution of errors. It can be seen in the figure that the proposed method has a smaller distance between Q1 and Q3. This shows that the proposed method produces less error and displays more stable prediction ability than other methods.

A common way to test the difference between two models' prediction results is to compute a paired t-test. It is also used in traffic forecast tasks to compare different methods (Habtemichael and Cetin, 2016). The significance of the changes in the mean forecast errors between the proposed method and base method is examined using a one-tailed paired t-test. The null hypothesis for the paired t-test is that there is no difference in the forecast error between the proposed method and base method, whereas the alternative hypothesis is that there is significant difference. The results indicate that the reductions in forecast errors are found to be very statistically significant, and thus the null hypothesis is rejected. Table 3 shows the statistical significance of forecast errors.

In addition to the forecast errors, the forecasting accuracy of spatial and temporal distributions are also important indices. Therefore, we perform predictions at multiple locations and time slots. The average correlation (AC) is defined to measure the performance of

Table 2
Evaluation of the prediction model

	15 min			*30 min*		
Method	*MAE (Mph)*	*RMSE (Mph)*	*MAPE (%)*	*MAE (Mph)*	*RMSE (Mph)*	*MAPE (%)*
ARIMA	3.78	5.27	14.04%	4.50	6.67	16.37%
LSVR	2.67	4.25	9.34%	3.78	6.21	12.57%
KNN	3.22	4.84	11.81%	4.56	7.04	16.45%
SAEs	3.05	4.39	9.59%	4.05	6.16	12.45%
LSTM	**2.58**	4.04	8.74%	3.66	5.76	11.72%
DLM8L	2.63	4.11	8.49%	3.60	5.91	11.81%
Ours	2.61	**3.96**	**7.35%**	**3.04**	**4.92**	**10.65%**

Note: Boldface values indicate the best performance.

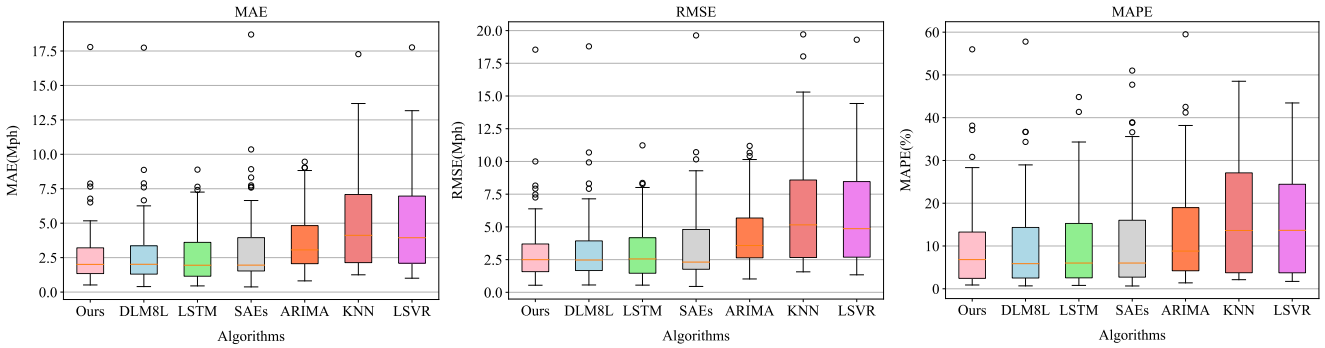

Fig. 14. Comparison of forecast errors for each method.

spatial and temporal distribution forecasting as shown in Equation (17):

$$AC\,(x,\ y) = \frac{1}{n}\sum_{i=1}^{n} Corrcoef\,(x_i, y_i) \tag{17}$$

where y_i is the actual traffic speed vectors, x_i is the predicted traffic speed vectors, and n are the number of predicted vectors on time dimension and space dimension, respectively. *Corrcoef* is the Pearson correlation coefficient.

Table 4 gives the average correlation of all of these methods on time and space dimensions. ACT represents the similarity between the predicted value and the observed value at the same time. ACS represents the similarity between the predicted value and the observed value in the same place. The results show that the proposed method has higher accuracy in both temporal and spatial dimensions.

We also use all samples to make a prediction over 24 hours. Figure 15 shows the performance of the traffic forecast using the Neural Network (NN) model in terms of MAE, MAPE, and RMSE. Over the course of an entire day, the forecast error of MAPE during off-peak hours is relatively lower than that of peak hours. The MAE and RMSE are low during late night and early morning hours, due to the decrease in observed traffic. As can be seen from the figure, the proposed method performs better at different time points compared with SAEs and LSTM, especially during peak hours. Although there are some anomalies that increase the prediction error, the MAPE of proposed method is still less than 15%. According to its RMSE, the prediction performance is stable throughout the day.

In addition to accuracy and stability, efficiency is another key factor that must be considered when evaluating a predictive model is efficient. Table 5 shows, in seconds, the training times and test times of comparable methods. Training time refers to the time required to train using the training set, and test time is the time required to run using the test set. All algorithms use the same software and hardware.

For the ARIMA algorithm, training takes a considerable amount of time due to the need to use historical data for training at each prediction point. LSVR has an excellent training and testing time. KNN is a "lazy-learning algorithm," so it takes almost no training time; the time spent in testing is also acceptable. Compared with the above methods, neural network-based methods consume more time in training, but there is not much difference in test time. We can also transfer the

Table 3
Statistical significance ($\alpha = 0.05$)

Metrics	*Comparison*	*t-Statistics*	*p-Value*	*Difference statistically significant*
MAE	Ours vs. ARIMA	−7.5445614763385835	2.710399655878317e−11	Yes
	Ours vs. LSVR	−7.639904635067548	1.714573322081373e−11	Yes
	Ours vs. KNN	−8.175901058009797	1.2805879347355704e−12	Yes
	Ours vs. SAEs	−2.920847316547501	0.004360623784599088	Yes
	Ours vs. LSTM	−2.5018579129064042	0.009205254144255483	Yes
	Ours vs. DLM8L	−1.8432394088308039	0.04683910846580523	Yes
RMSE	Ours vs. ARIMA	−7.881467850644976	5.3461219810072556e−12	Yes
	Ours vs. LSVR	−7.98287479771291	3.2712212438718124e−12	Yes
	Ours vs. KNN	−8.41847309465414	3.922729825404111e−13	Yes
	Ours vs. SAEs	−2.7122707463300033	0.007933053987762664	Yes
	Ours vs. LSTM	−2.764840130567599	0.0065290809772492	Yes
	Ours vs. DLM8L	−1.964840130567599	0.035290809772492	Yes
MAPE	Ours vs. ARIMA	−6.148233171777975	1.8365925512574116e−08	Yes
	Ours vs. LSVR	−5.9479214252010095	4.503745998678581e−08	Yes
	Ours vs. KNN	−6.463951266850565	4.367118113705813e−09	Yes
	Ours vs. SAEs	−3.058044284066558	0.002894127093241357	Yes
	Ours vs. LSTM	−2.8722331370253094	0.005026832899192536	Yes
	Ours vs. DLM8L	−2.2291438906786483	0.028161220655261478	Yes

Table 4
Average correlation

Methods	*ACT*	*ACS*
ARIMA	0.8395	0.9758
LSVR	0.8184	0.9507
KNN	0.8055	0.9495
SAEs	0.9197	0.9811
LSTM	0.9147	0.9830
DLM8L	0.9172	0.9846
Ours	0.9239	0.9863

Table 5
Comparison of train and test time

Method	*Train time (s)*	*Test time (s)*
ARIMA	1,293.436	1.972
LSVR	26.197	0.251
KNN	0.039	0.193
SAEs	138.429	0.131
LSTM	276.921	0.258
DLM8L	207.181	0.219
Ours	196.281	0.343

time-consuming training process offline without affecting the real-time performance of the algorithm. It can be seen from the table that although our proposed method includes multiple convolutional layers, the training speed is still faster than a single layer LSTM network. This shows that the prediction model based on CNN has better computational efficiency than the model based on RNN.

4.5 Traffic state estimation

It is also important to further determine the traffic conditions by predicted traffic speed; we must therefore assess whether the predicted speed accurately reflects future traffic trends. Figure 16 shows the comparison of the respective forecasts of normal days and holidays. The road speed during the holidays was significantly lower than the normal days, which indicates that the traffic conditions during the holidays are more congested. As can be seen from the figure, the proposed method produces good results on both normal days and holidays; it can be said that the proposed method provides reliable and accurate forecasts of speed in the study area. The predicted traffic speed has a similar traffic pattern to the observed traffic speed, and the prediction results display consistent similarity to the original data.

To evaluate the comprehensive performance of the proposed model, we divide the speed into different groups by Levels of Service (LOS) as shown in Table 6. This is also a common method in some forecasting tasks (Sun et al., 2010; Habtemichael and Cetin, 2016). We utilize the accuracy, the one-level

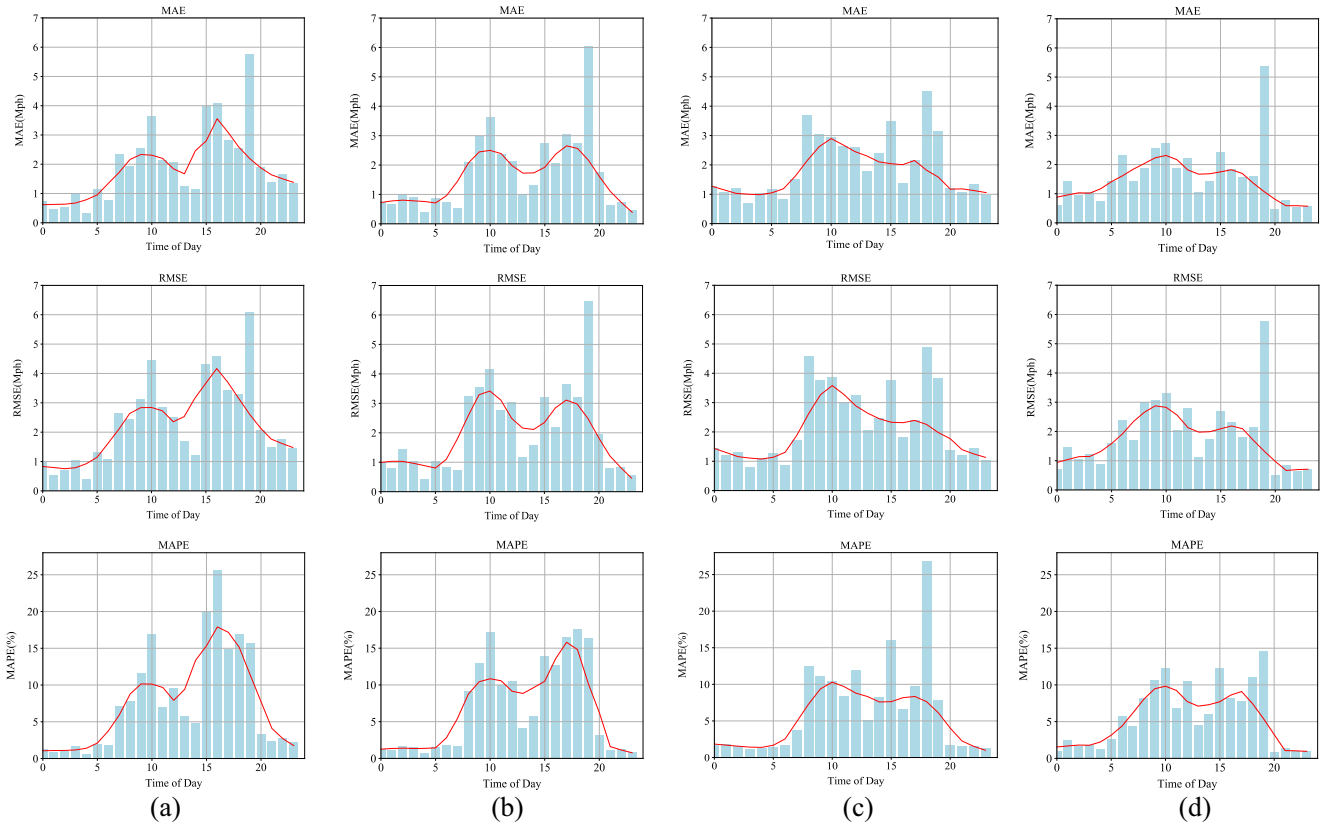

Fig. 15. Forecast errors for NN method in 24 hours. (a) SAEs; (b) LSTM; (c) DLM8L; (d) Ours.

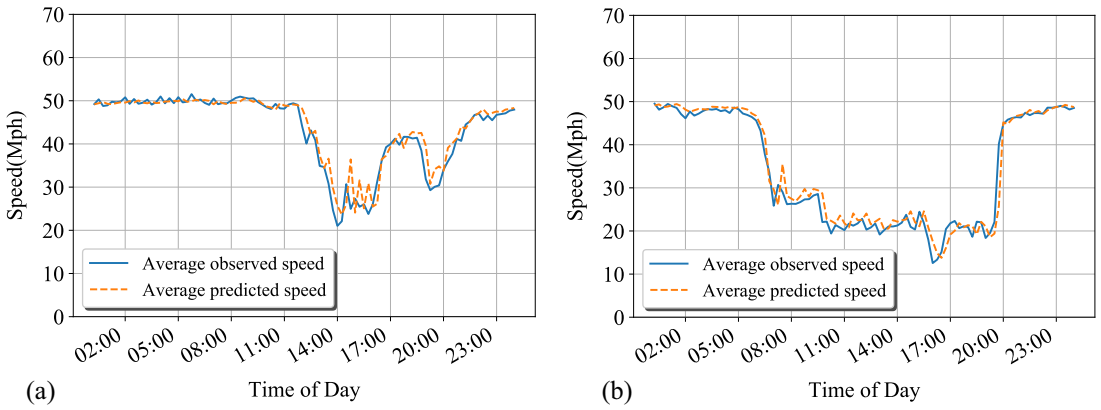

Fig. 16. Comparison of the forecasts with (a) normal day and (b) holiday.

deviation (OLD) and the high-level deviation (HLD) as the evaluation criteria in the traffic state estimation as Equations (18)–(20). The accuracy is represented as a ratio of the number of correct predictions to the number of total predictions. The OLD is the proportion of the number of the predicted group which is one level deviated from the truth to the total number, and the residual proportion is the HLD. As LOS reflects the human feeling to the traffic conditions, and different people have different feelings to the same status, the OLD is considered in the permissible error range. The results show that we can still obtain an acceptable result after converting the speed into traffic state estimation, as shown in Table 7.

$$Accuracy = \frac{n_{\text{right}}}{n_{\text{sample}}} \tag{18}$$

Table 6
Definition of the speed groups

Speed groups	*Speed (km/h)*	*LOS*
Group1	>48	A
Group2	40–48	B
Group3	32–40	C
Group4	24–32	D
Group5	0–24	E, F

Table 7
Prediction accuracy on different days

Days	*Accuracy (%)*	*OLD (%)*	*HLD (%)*
Normal	81.94%	15.97%	2.08%
Holiday	84.37%	14.06%	1.56%
Average	83.16%	15.02%	1.82%

$$OLD = \frac{n_{\text{one-deviation}}}{n_{\text{sample}}} \tag{19}$$

$$HLD = \frac{n_{\text{sample}} - n_{\text{right}} - n_{\text{one-deviation}}}{n_{\text{sample}}} \tag{20}$$

4.6 Impact of missing data

Data loss is a common problem in traffic data sets, due to detector failure, data transfer, and storage issues. Therefore, the robustness of the model to account for missing data is very important, especially for real-time traffic applications. Otherwise, these models may provide erroneous prediction result.

Although it is possible to estimate missing values, it is difficult to ensure that the inputs are accurate because the composition of traffic variables is affected by time and space changes and the relationship between them is not linear. For the neural network model, the network needs to analyze the interrelationship between input data. When time lag is small, models are more sensitive to missing data. The proposed short-term traffic forecasting method has the advantage of being able to deal with the problem of missing data better than the SAEs, LSTM, and DLM8L models. By randomly deleting some values of the test set, we can test the robustness of the proposed forecasting method in the absence of data. We consider using different proportions of missing data and comparing the prediction results. The missing rates are 0–15%. The impact of missing data on forecast errors of NN model is shown in Figure 17. We randomly delete various data, then apply all models to the same data and make comparisons. The x-axis represents the missing rate in the whole data set, whereas the y-axis represents the performance of the models for this missing rate. It can be seen that as the rate of missing rate increases, the forecasting errors of the model gradually increase.

Among them, SAEs produced the largest forecasting errors. Although the rate of error began to decline when the missing rate reached a certain percentage, it was still the fastest growing of the three methods. The MAE and RMSE of LSTM model have a slow growth rate, but the MAPE of LSTM grew rapidly, even exceeding SAEs even at a deletion rate of 15%. This situation is caused by the structure of LSTM; during the forecasting process, the next point in temporal sequence will be affected by the current time point. However, the artificial filling of 0 can undermine the relationship between the temporal sequence data, thereby negatively affecting the performance of the model. To verify the impact of multiattribute fusion on the prediction of missing data, we tested the proposed model with only the speed attribute, then with all attribute data. It can be seen that in the case of using only speed as input, the performance of the model shows a significant decline when the missing rate exceeds 5%. When using multiattribute fusion, the model has excellent performance; not only is the error value low, but also the error growth rate. This is because: (1) we use multiattribute data in forecasting tasks, so even if the most important value speed is missing, other attributes can express the speed value of the invalid point. (2) During the feature extraction process, the convolution operation can be used to fill in values of 0 with nearby values. (3) By adaptively weighting the features, the influence of the invalid feature on the forecasting task can be further reduced.

4.7 Visualization of spatial–temporal feature maps and attention model

The reason for utilizing an attention-based spatial–temporal DL architecture is that spatial–temporal correlations exist among the traffic variables. In the traffic forecast task, we considered that the spatial–temporal relationship of the data may have a different impact on the prediction results. Therefore, the feature maps attention module is used to weight the temporal features and spatial features, respectively. We have tested the proposed method without using the attention model in the case of fixed model hyperparameters. The results show that the attention model can help improve the prediction accuracy, as shown in Table 8.

To study the influence of spatial–temporal features on forecasting tasks, we have extracted some feature maps of the feature-extraction model and attention weights during the testing process. The weights generated during the test are accumulated and averaged

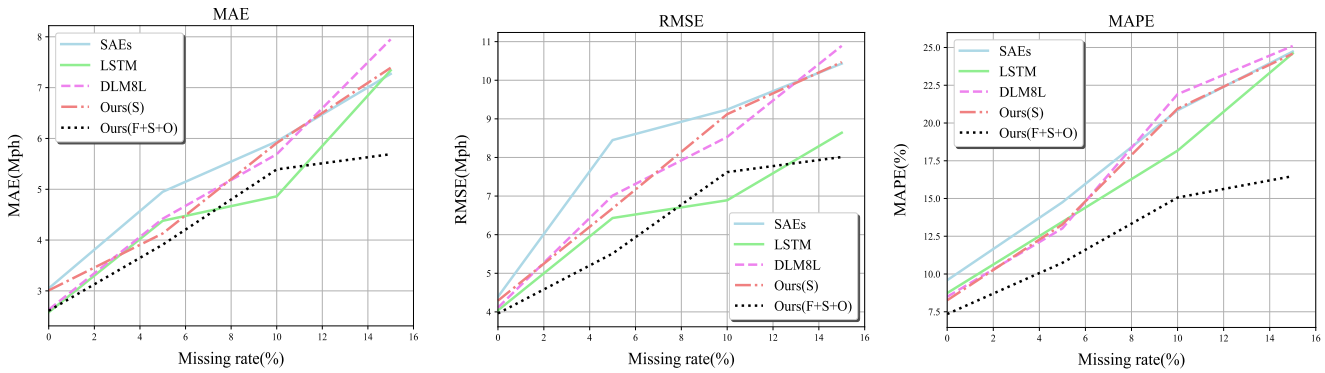

Fig. 17. Impact of missing data on forecast errors of NN model.

Table 8
Evaluation of the attention model

Metrics	*With attention*	*Without attention*
MAE (Mph)	**2.61**	2.73
RMSE (Mph)	**3.96**	4.02
MAPE (%)	**7.35%**	7.78%

Note: Boldface values indicate the best performance.

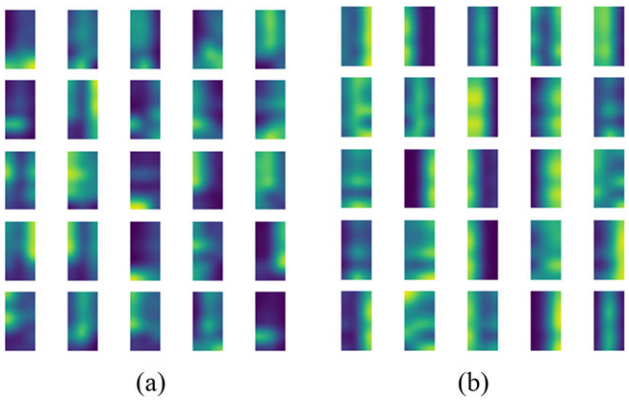

Fig. 18. Some feature maps. (a) Spatial feature maps; (b) Temporal feature maps.

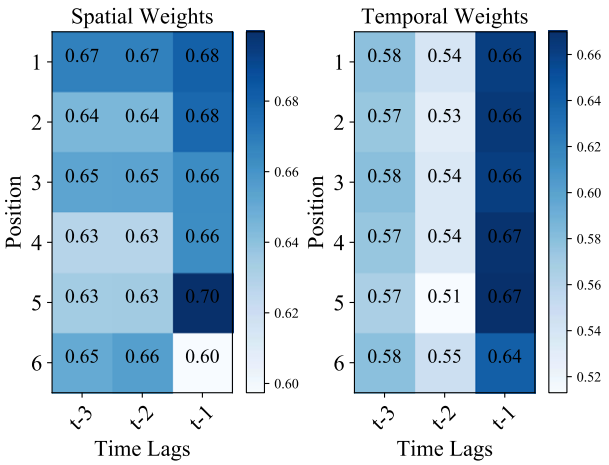

Fig. 19. Average spatial–temporal weighting matrix.

because the deep neural network has a large number of parameters and it is difficult to interpret the corresponding relationships within the network. Figure 18 shows some feature maps of spatial feature extraction and temporal feature extraction. Figure 19 shows the average spatial–temporal weighting matrix.

In the visualized feature maps, the yellow and green regions represent higher response values, and the blue regions represent lower response values. As can be seen from the figure, in the spatial feature maps, higher activation values tend to appear in different spatial locations. In the temporal feature maps, higher activation values tend to occur at different time lags. This can explain the tendency of different feature extraction models.

The spatial weight matrix is used to weight the extracted spatial features. Each detector on the road may be affected by upstream and downstream detectors, so different feature maps should correspond to different weights, but this cannot be reflected by the average matrix. However, we can see from the figure that downstream detectors tend to be assigned more weight than upstream detectors. This is because the speed of the detected point will be affected by the speed of the front. When the speed of the downstream vehicle decreases, the speed of the upstream vehicle will also be affected, namely, reduced (the direction of the vehicle is from position 6 to position 1). For the temporal weight matrix, we can see from the figure that for the predicted time point t, the historical data have an impact on the prediction task in the order of $\{t-1,\ t-3,\ t-2\}$. It is an interesting result that the data in $t-3$ are more effective than the data in $t-2$. The relationship between these spatial–temporal variables is rather complex. The

proposed method can automatically weight the different features in the model, and at the same time, obtain accurate traffic speed forecasting results.

5 CONCLUSIONS

In traffic management and control applications, it is important to predict the traffic flow in a timely and accurate manner. Short-term traffic forecasting tasks can provide more proactive traffic management strategies and provide travelers with reliable travel schedules. In this article, we propose an attention-based CNN to predict traffic speed. The model extracts the spatial features and the temporal features by using the ordinary convolution units and the gated convolution units and weights the features by using the attention module. Finally, the traffic speed of multiple detector locations is predicted by logistic regression. In the experiment, the data collected from the loop detector on U.S. Route I5 were used to evaluate the performance of the proposed method. Overall, despite the fact that the actual traffic data are complex and noise-intensive, our model achieves good results in short-term traffic speed forecasting tasks at different time intervals.

The major contributions of this study are as follows. (1) Our model used mixed traffic flow data, which not only improve prediction accuracy, but also reduce the accuracy degradation caused by missing data. (2) By introducing Gated CNN, the model is able to extract temporal features faster and more efficiently. (3) Attention model can effectively improve the accuracy and robustness of the model. At the same time, through the visualization of weights, it can also explain some influence of different traffic flow data on the prediction task.

In future work, in addition to further improving the accuracy of the model, the study locations will also be expanded from a single road to the entire road network. The existing model tests a single section of a single road. Although the comparison can prove the validity of the model, we believe that more impact factors need to be considered. Therefore, trying to use the entire road network as a model input is an available option. Owing to the development of deep learning technology, this effect can be achieved through a deep graph convolution network, which is an important direction for our future research.

ACKNOWLEDGMENTS

This work was supported by the National Key Research and Development Program of China (2017YFB0102603); the National Natural Science Foundation of China under Grants (U1564201, U1764264, 51775247, 61601203); China Postdoctoral Science Foundation (2017M611729); the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (17KJB580003); Key Project for the Development of Strategic Emerging Industries of Jiangsu Province (2016-1094, 2015-1084); Key Research and Development Program of Jiangsu Province (BE2015162, BE2016149); Key Laboratory for New Technology Application of Road Conveyance of Jiangsu Province (BM20082061503); Nanjing Science and Technology Development Program (201805008); Jiangsu University Scientific Research Foundation for Senior Professionals (16JDG046).

REFERENCES

Adeli, H. & Karim, A. (2000), Fuzzy-wavelet RBFNN model for freeway incident detection, *Journal of Transportation Engineering*, **126**(6), 464–71.

Adeli, H. & Samant, A. (2000), An adaptive conjugate gradient neural network—wavelet model for traffic incident detection, *Computer-Aided Civil and Infrastructure Engineering*, **15**(4), 251–60.

Ahmed, M. S. & Cook, A. R. (1979), Analysis of freeway traffic time-series data by using Box–Jenkins techniques, *Transportation Research Record*, **722**, 1–9.

Boto-Giralda, D., Díaz-Pernas, F. J., González-Ortega, D., Díez-Higuera, J. F., Antón-Rodríguez, M., Martínez-Zarzuela, M. & Torre-Díez, I. (2010), Wavelet-based denoising for traffic volume time series forecasting with self-organizing neural networks, *Computer-Aided Civil and Infrastructure Engineering*, **25**(7), 530–45.

Cha, Y. J., Choi, W. & Büyüköztürk, O. (2017), Deep learning-based crack damage detection using convolutional neural networks, *Computer-Aided Civil and Infrastructure Engineering*, **32**(5), 361–78.

Chan, K. Y., Dillon, T. S., Singh, J. & Chang, E. (2012), Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and Levenberg–Marquardt algorithm, *IEEE Transactions on Intelligent Transportation Systems*, **13**(2), 644–54.

Dauphin, Y. N., Fan, A., Auli, M. & Grangier, D. (2016), Language modeling with gated convolutional networks, arXiv preprint arXiv:1612.08083.

Dharia, A. & Adeli, H. (2003), Neural network model for rapid forecasting of freeway link travel time, *Engineering Applications of Artificial Intelligence*, **16**(7–8), 607–13.

García-Ródenas, R., López-García, M. L. & Sánchez-Rico, M. T. (2017), An approach to dynamical classification of daily traffic patterns, *Computer-Aided Civil and Infrastructure Engineering*, **32**(3), 191–212.

Gehring, J., Auli, M., Grangier, D., Yarats, D. & Dauphin, Y. N. (2017), Convolutional sequence to sequence learning, arXiv preprint arXiv:1705.03122.

Ghosh, B., Basu, B. & O'Mahony, M. (2010), Random process model for urban traffic flow using a wavelet-Bayesian hierarchical technique, *Computer-Aided Civil and Infrastructure Engineering*, **25**(8), 613–24.

Ghosh-Dastidar, S. & Adeli, H. (2003), Wavelet-clustering-neural network model for freeway incident detection, *Computer-Aided Civil and Infrastructure Engineering*, **18**(5), 325–38.

Habtemichael, F. G. & Cetin, M. (2016), Short-term traffic flow rate forecasting based on identifying similar traffic patterns, *Transportation Research Part C: Emerging Technologies*, **66**, 61–78.

Hamed, M. M., Al-Masaeid, H. R. & Said, Z. M. B. (1995), Short-term prediction of traffic volume in urban arterials, *Journal of Transportation Engineering*, **121**(3), 249–54.

He, Z., Zheng, L., Chen, P. & Guan, W. (2017), Mapping to cells: a simple method to extract traffic dynamics from probe vehicle data, *Computer-Aided Civil and Infrastructure Engineering*, **32**(3), 252–67.

Hochreiter, S. & Schmidhuber, J. (1997), Long short-term memory, *Neural Computation*, **9**(8), 1735–80.

Huang, W., Song, G., Hong, H. & Xie, K. (2014), Deep architecture for traffic flow prediction: deep belief networks with multitask learning, *IEEE Transactions on Intelligent Transportation Systems*, **15**(5), 2191–201.

Ioffe, S. & Szegedy, C. (2015), Batch normalization: accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167.

Jeong, Y.-S., Byon, Y.-J., Castro-Neto, M. M. & Easa, S. M. (2013), Supervised weighting-online learning algorithm for short-term traffic flow prediction, *IEEE Transactions on Intelligent Transportation Systems*, **14**(4), 1700–07.

Jiang, X. & Adeli, H. (2004), Wavelet packet-autocorrelation function method for traffic flow pattern analysis, *Computer-Aided Civil and Infrastructure Engineering*, **19**(5), 324–37.

Jiang, X. & Adeli, H. (2005), Dynamic wavelet neural network model for traffic flow forecasting, *Journal of Transportation Engineering*, **131**(10), 771–79.

Koziarski, M. & Cyganek, B. (2017), Image recognition with deep neural networks in presence of noise—dealing with and taking advantage of distortions, *Integrated Computer-Aided Engineering*, **24**(4), 337–49.

Kumar, K., Parida, M. & Katiyar, V. (2013), Short term traffic flow prediction for a non urban highway using artificial neural network, *Procedia-Social and Behavioral Sciences*, **104**, 755–64.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E. & Jackel, L. D. (1990), Handwritten digit recognition with a back-propagation network, in *Advances in Neural Information Processing Systems*, Morgan Kaufman Publishers, San Francisco, CA, pp. 396–404.

LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998), Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, **86**(11), 2278–324.

Levin, M. & Tsao, Y.-D. (1980), On forecasting freeway occupancies and volumes (abridgment), *Transportation Research Record*, **773**, 47–49.

Lin, Y. Z., Nie, Z. H. & Ma, H. W. (2017), Structural damage detection with automatic feature-extraction through deep learning, *Computer-Aided Civil and Infrastructure Engineering*, **32**(12), 1025–46.

Lippi, M., Bertini, M. & Frasconi, P. (2013), Short-term traffic flow forecasting: an experimental comparison of time-series analysis and supervised learning, *IEEE Transactions on Intelligent Transportation Systems*, **14**(2), 871–82.

Lv, Y., Duan, Y., Kang, W., Li, Z. & Wang, F.-Y. (2015), Traffic flow prediction with big data: a deep learning approach, *IEEE Transactions on Intelligent Transportation Systems*, **16**(2), 865–73.

Ma, X., Tao, Z., Wang, Y., Yu, H. & Wang, Y. (2015), Long short-term memory neural network for traffic speed prediction using remote microwave sensor data, *Transportation Research Part C: Emerging Technologies*, **54**, 187–97.

Maas, A. L., Hannun, A. Y. & Ng, A. Y. (2013), Rectifier nonlinearities improve neural network acoustic models, *Proceedings the 30th International Conference on Machine Learning*, **28**, 1–6.

Mascha, V. D. V., Dougherty, M. & Watson, S. (1996), Combining Kohonen maps with ARIMA time series models to forecast traffic flow, *Transportation Research Part C: Emerging Technologies*, **4**(5), 307–18.

Min, W. & Wynter, L. (2011), Real-time road traffic prediction with spatio-temporal correlations, *Transportation Research Part C: Emerging Technologies*, **19**(4), 606–16.

Ortega-Zamorano, F., Jerez, J. M., Gómez, I. & Franco, L. (2017), Layer multiplexing FPGA implementation for deep back-propagation learning, *Integrated Computer-Aided Engineering*, **24**(2), 171–85.

Ou, J., Xia, J., Wu, Y.-J. & Rao, W. (2017), Short-term traffic flow forecasting for urban roads using data-driven feature selection strategy and bias-corrected random forests, *Transportation Research Record: Journal of the Transportation Research Board*, **2645**, 157–67.

Polson, N. G. & Sokolov, V. O. (2017), Deep learning for short-term traffic flow prediction, *Transportation Research Part C: Emerging Technologies*, **79**, 1–17.

Qi, Y. & Ishak, S. (2014), A hidden Markov model for short term prediction of traffic conditions on freeways, *Transportation Research Part C: Emerging Technologies*, **43**, 95–111.

Rafiei, M. H. & Adeli, H. (2017), A novel machine learning-based algorithm to detect damage in high-rise building structures, *Structural Design of Tall and Special Buildings*, **26**(18), 1–11.

Rafiei, M. H. & Adeli, H. (2018), A novel unsupervised deep learning model for global and local health condition assessment of structures, *Engineering Structures*, **156**, 598–607.

Roess, R. P., Prassas, E. S. & McShane, W. R. (2004), *Traffic Engineering*, Pearson/Prentice Hall, Upper Saddle River, NJ.

Sun, X., Jia, L., Dong, H., Qin, Y. & Guo, M. (2010), Urban expressway traffic state forecasting based on multimode maximum entropy model, *Science China Technological Sciences*, **53**(10), 2808–16.

Vlahogianni, E. I., Karlaftis, M. G. & Golias, J. C. (2007), Spatio-temporal short-term urban traffic volume forecasting using genetically optimized modular networks, *Computer-Aided Civil and Infrastructure Engineering*, **22**(5), 317–25.

Wang, J., Deng, W. & Guo, Y. (2014), New Bayesian combination method for short-term traffic flow forecasting, *Transportation Research Part C: Emerging Technologies*, **43**, 79–94.

Williams, B. M. & Hoel, L. A. (2003), Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: theoretical basis and empirical results, *Journal of Transportation Engineering*, **129**(6), 664–72.

Wu, Y. & Tan, H. (2016), Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework, arXiv preprint arXiv:1612.01022.

Xingjian, S., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K. & Woo, W.-C. (2015), Convolutional LSTM network: a machine learning approach for precipitation nowcasting, in *Proceedings of Advances in Neural Information Processing Systems*, vol. 9199, 802–10.

Yao, B., Chen, C., Cao, Q., Jin, L., Zhang, M., Zhu, H. & Yu, B. (2017), Short-term traffic speed prediction for an urban corridor, *Computer-Aided Civil and Infrastructure Engineering*, **32**(2), 154–69.

Yin, W., Schütze, H., Xiang, B. & Zhou, B. (2015), ABCNN: attention-based convolutional neural network for modeling sentence pairs, arXiv preprint arXiv:1512.05193.

Yu, R., Li, Y., Shahabi, C., Demiryurek, U. & Liu, Y. (2017), Deep learning: a generic approach for extreme condition traffic forecasting, in *Proceedings of the 2017 SIAM International Conference on Data Mining*, 777–85.

Zhang, J., Zheng, Y. & Qi, D. (2017), Deep spatio-temporal residual networks for citywide crowd flows prediction, in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 1655–61.

Zhao, Z., Chen, W., Wu, X., Chen, P. C. & Liu, J. (2017), LSTM network: a deep learning approach for short-term traffic forecast, *IET Intelligent Transport Systems*, **11**(2), 68–75.