# Deep and Embedded Learning Approach for Traffic Flow Prediction in Urban Informatics

Zibin Zheng, *Senior Member, IEEE*, Yatao Yang, Jiahao Liu, Hong-Ning Dai, *Senior Member, IEEE*, and Yan Zhang, *Senior Member, IEEE*

***Abstract*— Traffic flow prediction has received extensive attention recently, since it is a key step to prevent and mitigate traffic congestion in urban areas. However, most previous studies on traffic flow prediction fail to capture fine-grained traffic information (like link-level traffic) and ignore the impacts from other factors, such as route structure and weather conditions. In this paper, we propose a deep and embedding learning approach (DELA) that can help to explicitly learn from fine-grained traffic information, route structure, and weather conditions. In particular, our DELA consists of an embedding component, a convolutional neural network (CNN) component and a long short-term memory (LSTM) component. The embedding component can capture the categorical feature information and identify correlated features. Meanwhile, the CNN component can learn the 2-D traffic flow data while the LSTM component has the benefits of maintaining a long-term memory of historical data. The integration of the three models together can improve the prediction accuracy of traffic flow. We conduct extensive experiments on realistic traffic flow dataset to evaluate the performance of our DELA and make comparison with other existing models. The experimental results show that the proposed DELA outperforms the existing methods in terms of prediction accuracy.**

***Index Terms*— Urban informatics, traffic flow prediction, embedding neural networks, deep learning.**

## I. Introduction

WE ARE experiencing the urbanization shift. It is predicted in [1] that more than 60% the world's population will live in urban areas by 2050. Traffic congestion is one of the major challenges to enable urbanization. The deployment of Intelligent Transportation Systems (ITS) in urban areas brings the opportunities to prevent or reduce traffic congestion. Meanwhile, recent advances in sensing technologies and the wide distribution of various sensors of ITS result in the proliferation of massive traffic data. Big traffic data analytics can help to manage and control the urban transportation efficiently.

Traffic flow prediction serving as the key component of ITS can assist ITS to forecast and prevent traffic congestion, control and manage traffic efficiently, and plan the best traveling route. Data-driven traffic flow prediction has received extensive attention recently due to the availability of massive traffic data from various sensors deployed in ITS. Machine learning (ML) based approaches such as $k$-nearest neighbor ($k$NN) algorithm [2], Markov process based scheme [3] and Artificial Neural Network (ANN) schemes [4]–[6]. Compared with conventional ML methods, deep learning (DL) models have the advantages such as simplifying data preprocessing procedure and outperforming other ML methods in terms of accuracy. Therefore, DL schemes have received extensive attention recently in traffic flow prediction, e.g., [7]–[12].

However, most previous ML and DL approaches have the following limitations: 1) failed to capture fine-grained features of traffic data like link-level traffic data; 2) failed to consider other categorical factors such as special events, route structure and weather conditions that also affect the traffic flow even though some previous studies consider a single categorical factors (like rainfall [13]). The recent proliferation of inductive-loop sensors deployed in the highways [9], [10] and the availability of multiple categorical features (such as weather conditions, route structure and special events) [13], [14] bring the opportunities in analyzing the traffic flow at the *link level* with consideration of categorical features. Motivated by the limitations of previous studies and the availability of fine-grained traffic flow data and traffic categorical data, we propose a Deep and Embedding Learning Approach (DELA) to analyze the traffic flow in terms of travel time.

The proposed DELA consists of an embedding component, a Convolutional Neural Networks (CNN) component and a Long Short-term Memory (LSTM) component. Compared with conventional ML and DL approaches, DELA has the following merits:

1) *Learning fine-grained traffic flow features.* The CNN component can learn the traffic flow features from

Manuscript received March 25, 2018; revised August 6, 2018, December 5, 2018 and February 18, 2019; accepted March 11, 2019. This work was supported by the National Key Research and Development Program under Grant 2016YFB1000101, the National Natural Science Foundation of China under Grant 61722214, Macao Science and Technology Development Fund under Grant 0026/2018/A1, the Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (2016), the European Union's Horizon 2020 Research and Innovation Programme under the Marie Skłodowska-Curie Agreement 824019, the Sichuan Science and Technology Program under Grant 2019YFH0033, and the Program for Guangdong Introducing Innovative and Entrepreneurial Teams under Grant 2016ZT06D211. The Associate Editor for this paper was S.-H. Kong. *(Corresponding author: Yan Zhang.)*

Z. Zheng, Y. Yang, and J. Liu are with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China, and also with the National Engineering Research Center of Digital Life, Sun Yat-sen University, Guangzhou 510006, China (e-mail: zhzibin@mail.sysu.edu.cn; yangyt9@mail2.sysu.edu.cn; liujiah9@mail2.sysu.edu.cn).

H.-N. Dai is with the Faculty of Information Technology, Macau University of Science and Technology, Macau, China (e-mail: hndai@ieee.org).

Y. Zhang is with the Department of Informatics, University of Oslo, 0315 Oslo, Norway, and also with the Simula Metropolitan Center for Digital Engineering, Norway (e-mail: yanzhang@ieee.org).

Digital Object Identifier 10.1109/TITS.2019.2909904

2-D traffic flow data at the *link* level in contrast to the route-level data in conventional DL methods. Meanwhile, CNN can also overcome the limitations of common DL models like the large number of parameters and the overfitting problem.

2) *Learning long-term dependent traffic flow features.* The LSTM component has the advantages in learning the periodicity of traffic flow data due to the excellent ability of memorizing long-term dependencies.
3) *Capturing multiple categorical features.* The embedding component can essentially capture the categorical features such as special events, route structure and weather conditions and investigate the impacts of the categorical features on the travel time. In addition, it not only learn the hidden information of the impacts of special events, weather conditions and route structures, but also it can learn the relationship between different impacts.

The proposed DELA obtains the benefits from the embedding component, the CNN component and the LSTM component consequently resulting in an excellent performance in terms of prediction accuracy of the travel time. *To the best of our knowledge, this is the first hybrid approach of integrating CNN, LSTM and categorical embedding in traffic flow prediction*. The main research contributions of this paper can be summarized as follows.

- We propose a novel deep learning method (DELA) to analyze the traffic flow. We consider the impacts of traffic flow, special events, weather conditions and route structures together. Our DELA model can make full use of these information and the relationship between them.
- Our model is based on *link-level* traffic flow data unlike the conventional methods that are based on coarse route-level traffic.
- We conduct extensive experiments on realistic datasets to evaluate the performance of the proposed DELA. The experimental results show that DELA outperforms existing approaches in terms of prediction accuracy of the travel time.
- Experimental results also show that DELA is efficient in real-time traffic-flow prediction. For example, the prediction time of DELA is less than 0.5 second implying DELA can fulfill the basic requirement in real-time prediction.

The remainder of the paper is organized as follows. Section II gives an overview on related studies. We then present the problem analysis in Section III. Section IV presents the DELA model. We then give the experimental results in Section V. Finally, we conclude the paper in Section VI.

## II. Related Work

We categorize the studies on traffic flow prediction into two categories: 1) parametric approaches and 2) non-parametric approaches.

### A. Parametric Approaches

In the early stage of traffic flow prediction, parametric approaches were typically proposed. Parametric methods are usually based on the assumption on specific functions for some variables (either independent or dependent). These approaches include auto-regressive integrated moving average (ARIMA) model [15], HoltWinter [16] and their variants [17]–[19]. These linear time-series models can capture the trend and periodicity information from the flow. The advantages of these linear time-series models include simplicity and efficiency while the main disadvantage is the low accuracy due to the impacts of chaotic and fractal characteristics of traffic flows.

There are other models proposed to improve prediction accuracy of traffic flow. In [20], a method of integrating Kalman filter and ARIMA model was proposed. Experimental results also verify the effectiveness of the proposed method. The benefit of Kalman filter is to update the state variables continuously so that it can improve the prediction accuracy [21]. Moreover, Nair et al. [22] established a nonlinear time-series model to analyze a traffic data. Furthermore, the work in [23] presents a multi-layer strategy to identify and cluster the nonlinear traffic structural patterns.

### B. Non-Parametric Approaches

The proliferation of various sensors deployed in ITS results in the availability of massive traffic data. Non-parametric (or data-driven) approaches based on massive traffic data can potentially improve traffic flow prediction. In general, machine learning (ML) based methods can identify the patterns and capture the key features of traffic flows, consequently improving the prediction accuracy. In [2], a short-term traffic condition prediction model based on $k$-nearest neighbor algorithm was proposed. The study in [3] proposed a Markov process based method to predict traffic conditions between roads. Among ML-based methods, an Artificial Neural Network (ANN) can accurately recognize the traffic patterns, consequently surpassing other ML methods [24]. Typically, an ANN needs to update the weights of the hidden layer via back propagation [25]. We name such NN methods as Back-Propagation Neural Networks (BPNN). Many BPNN-based or hybrid BPNN-based models [4]–[6] were proposed in traffic flow prediction.

Compared with conventional BPNN-based methods, deep learning (DL) models have the advantages in learning complicated and hierarchical features of massive data [26], [27]. Recently, DL methods have shown their strengths in traffic flow prediction [7], [8], [10], driving behavior feature extraction [11] and city-wide crowd flows prediction [28]. One deep learning model is recurrent neural network (RNN), which has the advantage of capturing time-series characteristics during the training and predicting phase [29]–[31]. Long short-term memory (LSTM) [32], [33] improves RNN by including *memory cells* which can preserve information for a long period. As a result, LSTM can learn longer-term dependencies. Recently, both RNN and LSTM models have been investigated in travel speed prediction [34] and short-term traffic flow forecast [12]. In addition, convolutional neural networks (CNN) have also been used in transportation network speed prediction [35].

Moreover, many other factors such as weather conditions and special events have influence on traffic flow. In particular, the weather conditions (such rainfalls and snows) [13], [36]–[38] can affect the urban traffic flow. For example, it is shown in [13] that the rainfall intensity and visibility can significantly affect the freeway traffic flow. Moreover, ref. [38] shows that the rainfall intensity may alter users' route choice consequently affecting the traffic flow. In addition to weather conditions, special events or accident events can also significantly affect the traffic flow. For example, it is shown in [10] that the traffic flow fluctuates differently in a special event day (like a football game) compared with an ordinary day. Moreover, the accidental event like traffic jam or an accident has significant impacts on traffic flow [14].

*Limitations of Previous Studies:* In summary, previous studies failed to address the following issues. First, most previous studies analyze the traffic flow at the *route level*. The route-level analysis conducted between the entry and the exit is relatively coarse and cannot capture the influence of other route-related factors. For example, the traffic flow of a route is also affected by traffic flow condition at each segment of the route. Second, although some previous studies consider the impacts of weather conditions and special events on traffic flow, most of them only consider a single factor like rainfall intensity or a special event and their analysis is only based on coarse traffic flow data (i.e., route-level traffic).

In this paper, we propose a new deep and embedding learning approach (DELA) to overcome the above limitations of previous studies.

## III. Problem Analysis

We investigate a real highway traffic dataset in a certain city in China from July 19, 2016 to Oct. 17, 2016 released by Knowledge Discovery and Data Mining Tools Competition (KDD CUP 2017).[1] Note that the exact city is omitted by the data publisher. In particular, this dataset contains the following types of data: i) traffic flow data, e.g., the average travel time for a particular route or a link, ii) route structure data, e.g., the number of links for a route, iii) weather condition data including atmospheric pressure, pressure at sea level, wind direction, wind speed, air temperature, relative humidity and precipitation.

### A. Preliminary Analysis

We first conducted a preliminary analysis on a subset of the whole dataset. Fig. 1 shows an example of the route structure of a route consisting of 6 routes: (1) route $A \rightarrow 1$, (2) route $A \rightarrow 2$, (3) route $A \rightarrow 3$, (4) route $B \rightarrow 1$, (5) route $B \rightarrow 2$, and (6) route $B \rightarrow 3$, where $\rightarrow$ connects a start to an end. Each route consists of multiple links. For example, route $A \rightarrow 1$ consists of four links, i.e., link 1, link 2, link 3 and link 10. In particular, we define a route denoted by $R$ as follows,

$$R := (S, E, \mathsf{linkSets}), \tag{1}$$

[1]Traffic Flow Data/KDD CUP 2017 https://tianchi.aliyun.com/datalab/dataSet.html?spm=5176.100073.0.0.63696fc1cVfp6R&dataId=60

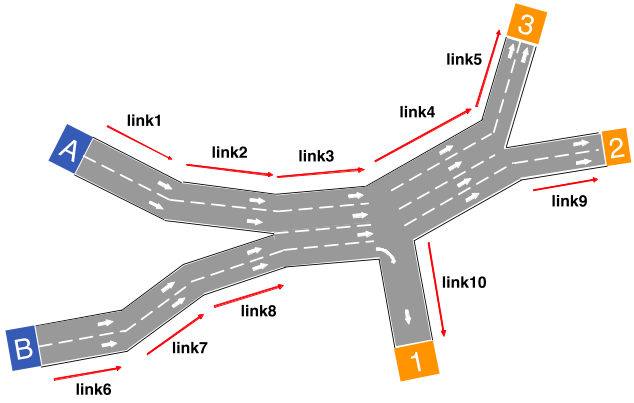
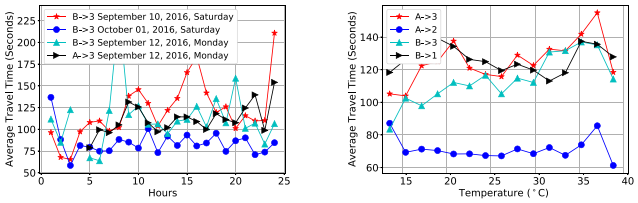



Fig. 1. An example of route structure.




(a) (b)

Fig. 2. Multiple factors affect travel time. (a) Impacts of days and routes. (b) Temperature affects travel time.

where $S$ denotes the start location, $E$ denotes the end location and $\mathsf{linkSets}$ refers to the links between the start and the end.

The route structure data has the following characteristics: i) two routes may share some common links, e.g., route $A \rightarrow 1$ shares link 3 with route $B \rightarrow 2$; ii) links of a road may be different from each other in terms of road structure, e.g., link 1 is a narrow road with 2 lanes while link 3 is a wide road with 4 lanes. The route structure should be considered in traffic flow analysis since these two features may have a significant influence on the traffic flow.

Next, we conduct a preliminary analysis on traffic flow data based on two routes $B \rightarrow 3$ and $A \rightarrow 3$ on three typical days, i.e., Sep. 10, 2016, Sep. 12, 2016 and Oct. 1, 2016. These three days represent a weekend day, a week day and a holiday,[2] respectively. Fig. 2(a) shows the average travel time in a day. We have the following observations from Fig. 2:

1) There are two peaks for average travel time in a week day. For example, there are two peaks in the date Sep. 12, 2016 for route $B \rightarrow 3$: one is from 7 am to 9 am in the morning and another one is from 6 pm to 8 pm in the afternoon.
2) The peaks in a holiday and a weekend day are different from those in a week day. For example, the morning peak hour in a weekend day (e.g., Sep. 10, 2016) is 10 am for route $B \rightarrow 3$ while that in a weekday is 8 am.
3) The peaks in a holiday are different from those in a weekend day. For example, the morning peak hour in a holiday (e.g., Oct. 1, 2016) is 2 am while that in a

[2]October 1st is the national day for China and is usually the first day of the annual 7-day national holiday)

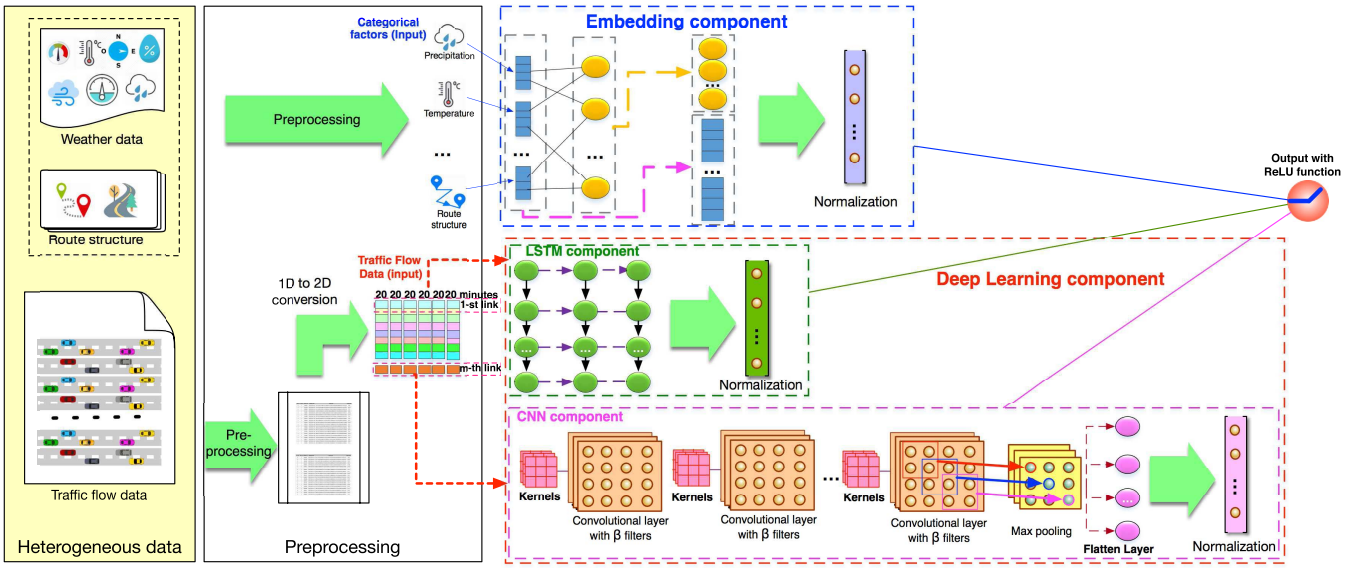



Fig. 3. Deep and Embedding Learning Approach (DELA).

weekend day (e.g., Sep. 10, 2016) is about 10 am; this may owe to the fact that the national day is usually a seven-day weekend and it is one of the busiest periods for travelling in China.

We further analyze the impact of other factors (such as weather conditions) on traffic flow. Fig. 2(b) shows the average travel time against temperature for the four routes $A \rightarrow 2$, $A \rightarrow 3$, $B \rightarrow 1$ and $B \rightarrow 3$. We find from Fig. 2(b) that the average travel time for route $A \rightarrow 2$ reaches the peak at a high temperature (i.e., 36°C) while that for route $B \rightarrow 1$ reaches at a low temperature (i.e., 18°C); it implies that the travel time is also affected by the weather condition.

### B. Challenges

The preliminary analysis in Section III-A shows that the traffic flow dataset also has the following characteristics:

- *Heterogeneous data types.* There are different types of data in the traffic flow dataset. For example, the data type of air temperature is different from that of the route structure).
- The travel time is *affected by multiple factors.* These factors include the route structure, traffic flow information, day information, special event and weather condition.
- The factors are often *correlated to each other.* This observation is different from previous studies that only consider one factor affecting the traffic flow.

These characteristics pose a significant challenge in traffic flow prediction. To address the above challenges, we propose a new deep and embedding learning approach (DELA).

## IV. Our Approach

As shown in Fig. 3, our analytical process mainly includes two stages: 1) data preprocess and 2) data analysis based on deep and embedding learning approach (DELA). DELA mainly consists of two major components: the embedding component to be presented in Section IV-A and the deep learning component to be presented IV-B. We then explain them in detail as follows.

### A. Embedding Component

In recent years, a number of embedding methods have been proposed. Most of them transform an ID into an embedding vector that can then be used to represent the hidden information of the ID [39], [40]. As shown in Fig. 3, the embedding component (enclosed in the blue dash box) contains three components. The input embedding layer is the first component, which learns categorical features, such as route structure information, weather conditions, and date information. However, there is no way for our DELA to learn such categorical features directly. In order to convert categorical data into a numerical form denoted by embedding vectors, we firstly exploit *one-hot encoding* to represent categorical features [41]. We then take the one-hot encoding vector as inputs of the full connection layer. We next apply back-propagation algorithm to calculate the weight of every edge in the full connection layer. The output layer consisting of the full connection layer is also an embedding layer, i.e., embedding vectors of start location IDs or link IDs. Embedding maps the data from resource space to target space with structural-preservation.

The embedding vector can represent the correlation between different categories. In particular, we derive the inner dot product of any two different embedding vectors. We then concatenate the embedding vectors with the inner dot product values. The concatenated vectors can better represent categorical features and the correlation between two different categorical features. Specifically, the output of embedding component is given by Eq. (2),

$$\hat{\mathbf{y}}_{\text{Embedding}} := f(\hat{\mathbf{x}}) := \sum_{i \in \hat{\mathbf{x}}} \sum_{j \neq i, j \in \hat{\mathbf{x}}} w_{i,j} \langle \mathbf{v}_i, \mathbf{v}_j \rangle + \sum_{i \in \hat{\mathbf{x}}} w_i \mathbf{v}_i, \tag{2}$$

where $\hat{\mathbf{x}}$ represents the set of categorical features, $i, j \in \hat{\mathbf{x}}$ are categorical features from this set, $\mathbf{v}_i$ represents the embedding feature of category $i$, $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$ is the inner dot product of two categorical embedding vectors $\mathbf{v}_i$ and $\mathbf{v}_j$, $w_{i,j}$ represents the weight of inner dot product value and $w_i$ is the weight of embedding vector. Eq. (2) essentially indicates that the embedding component not only learns features, but also captures the correlation between two different categorical features.

### B. Deep Learning component

*1) Data Preprocess:* The travel time was taken over each link every 20 minutes. If there is no vehicle passing within 20 minutes for a link, the average travel time for that link will be a missing value. In this paper, we exploit the interpolation method to recover the missing values according to the following equation,

$$f(\mathbf{x}_i^l) = \begin{cases} \dfrac{\mathbf{x}_{i-1}^l + \mathbf{x}_{i+1}^l}{2} & \mathbf{x}_i^l \in \mathsf{NaN}, \mathbf{x}_{i-1}^l, \mathbf{x}_{i+1}^l \notin \mathsf{NaN}, \\ 0 & \mathbf{x}_i^l \in \mathsf{NaN}, \mathbf{x}_{i-1}^l \text{ or } \mathbf{x}_{i+1}^l \in \mathsf{NaN}, \\ \mathbf{x}_i^l & \mathbf{x}_i^l \notin \mathsf{NaN}, \end{cases} \tag{3}$$

where $\mathbf{x}_i^l$ stands for the average travel time over the $i$-th 20 minutes within two-hour window on link $l$. If $\mathbf{x}_i^l$ is a null value, we represent it as $\mathsf{NaN}$.

As neural networks are sensitive to the diversity of input data, we need to normalize input data. Specifically, we choose the MAX-MIN scaling method to normalize the data [42].

*2) Input Data:* Our preliminary results (refer to Section III) reveal the relevance of different routes. In particular, different routes may share some common links. Motivated by this observation, we partition traffic flow data by *links* in contrast to conventional methods partitioning traffic flow data by *routes*. This *fine-grained analysis* on the traffic flow data can more accurately model the realistic traffic since the traffic is different from link to link even in the same route. Take Fig. 1 as an example again. Route $A \rightarrow 2$ consists of links 1 and 3, where link 1 only has two lanes and link 3 has four lanes. Essentially, the traffic flow in link 1 is quite different from that in link 3. The fine-grained analysis can better capture the features of traffic flow than conventional schemes. In this manner, we convert 1-D traffic flow data into 2-D data. We will show through experiments presented in Section V that this scheme can significantly improve the prediction accuracy.

Every route may contain different number of links. Each route is represented with the same number of links (i.e., the maximum number of links among all the routes) and the fields for those non-existing links are filled with zeros. For example, if the maximum number of links for all the routes is 5 and a route contains three links like [101, 105, 107]. Then after being filled with zeros, it becomes [101, 105, 107, 0, 0]. Recall that the average travel time of the $l$-th link during the $i$-th minutes is $\mathbf{x}_i^l$. In accordance with proposed non-zero padding scheme, we obtain the average travel time matrix denoted by $\mathbf{x}$ by averaging the travel time over all the links of that route and over the given time period. It is obvious that the newly constructed input data is not 1-D traffic data, but 2-D traffic data that contains both the route structure and time information.

Fig. 3 shows that Deep Learning component (enclosed in the red dash box) consists of a CNN component and an LSTM component. We then explain them in details as follows.

*3) CNN Component:* CNN overcomes the disadvantages of regular neural networks by connecting each neuron to its neighboring neurons not all the neurons [43]. CNN conventionally takes images as the input. Few studies use CNN to analyze traffic flow data. In this paper, we exploit the benefits of CNN by using 2-D traffic flow data. In our CNN component, there are multiple convolution layers as shown in Fig. 3. We use the 2-D convolution layer to extract the features from the input 2-D average travel time data. We denote the average travel time of the $p$-th link by vector $\mathbf{v}_p \in \mathbf{R}^d$, where $d$ is 6 in this paper because the travel time was taken over each link every 20 minutes within 2 hours. We then represent the concatenated average travel time of $m$ links (denoted by $\mathbf{v}_{1:m}$) as follows,

$$\mathbf{v}_{1:m} := \mathbf{v}_1 \oplus \mathbf{v}_2 \oplus ... \oplus \mathbf{v}_m, \tag{4}$$

where $\oplus$ is the concatenation operator and $m = \lceil \frac{n}{6} \rceil$. In general, let $\mathbf{v}_{p:p+k}$ refer to the concatenation of every link's average travel time $\mathbf{v}_p, \mathbf{v}_{p+1}, ..., \mathbf{v}_{p+k}$. In this manner, we finally obtain a 2-D feature map.

One convolution operation typically involves a filter $\hat{\mathbf{w}}$, which is applied to a window of size $3 \times 3$ to produce a new feature. Note that we choose the filter with $3 \times 3$ in this paper because this typical setting is shown to perform excellent in most cases [43]. Moreover, we choose $\beta$ filters, which are also adjustable in the experiments. In a similar manner, we consider a feature $\mathbf{c}_{p+1,q+1}$, which can be generated from a window of matrix $\mathbf{v}_{p:p+2,q:q+2}$ by the following equation,

$$\mathbf{c}_{p+1,q+1} := f(\hat{\mathbf{w}}(g(\mathbf{v}_{p:p+2,p:p+2})) + b_2), \tag{5}$$

where $b_2 \in R$ is a bias term and $f(\cdot)$ is a non-linear function such as the hyperbolic tangent [43]. We then apply this filter to data blocks denoted by $\mathbf{v}_{1:3,1:3}, \mathbf{v}_{1:3,2:4}, \ldots, \mathbf{v}_{m-2:m,m-2:m}$, respectively to produce a feature map as given as follows,

$$\mathbf{c} := \begin{bmatrix} \mathbf{c}_{1,1} & \cdots & \mathbf{c}_{1,6} \\ \vdots & \ddots & \vdots \\ \mathbf{c}_{m,1} & \cdots & \mathbf{c}_{m,6} \end{bmatrix}, \tag{6}$$

where $\mathbf{c}$ is the generated feature map having the same size as the raw matrix because of zero-padding.

*4) Max Pooling Layer:* The max pooling layer is typically used in CNN to reduce the number of parameters (e.g., training weights and filters) and the redundant features. Meanwhile, the max pooling layer can also be used to control the convergence of neural networks (e.g., avoid overfitting). The pooling layer typically chooses the maximum (i.e., *max*) value of the region covered by the pooling filter. In particular, we define the max pooling operation as follows,

$$\hat{\mathbf{y}}_{\text{CNN}} = \max(\mathbf{c}), \tag{7}$$

where $\mathbf{c}$ is the feature map defined in Eq. (6) serving as the input of the max pooling operation.

*5) LSTM Component:* Traffic flow data is highly related to the historical information. Due to the excellent ability of memorizing long-term dependencies, LSTM model has the advantages in traffic flow prediction. After following sophisticated derivation in [44], we can calculate the output values of LSTM model according to the following equation,

$$\hat{\mathbf{y}}_{\text{LSTM},t} = W_{hy} * h_t + b_h, \tag{8}$$

where $\hat{\mathbf{y}}_{\text{LSTM},t}$ is the predicted value, $W_{hy}$ is the weight of the current hidden state information and $b_h$ is the bias value.

*6) Ensembling:* Finally, the embedding component and the Deep Learning component are combined using a weighted sum as hidden features after normalization. These features are then concatenated and fed to a rectified linear unit (ReLU). We consider the weighted sum of the outputs of the embedding component, the CNN component and LSTM component together and optimize the generated parameters at the same time. We then upgrade weights via back-propagation. Essentially, CNN component can capture the periodicity of the traffic flow data and LSTM component can learn from the historical information. The integration of CNN component and LSTM component obtains the benefits of CNN and LSTM models consequently leading to the excellent performance. In particular, the prediction of the model is finally given as follows,

$$\mathbf{Y} := \mathbf{W}[\hat{\mathbf{y}}_{\text{Embedding}}, \hat{\mathbf{y}}_{\text{CNN}}, \hat{\mathbf{y}}_{\text{LSTM}}] + \mathbf{b}, \tag{9}$$

where $\mathbf{Y}$ is the travel time, $\hat{\mathbf{y}}_{\text{Embedding}}$, $\hat{\mathbf{y}}_{\text{CNN}}$ and $\hat{\mathbf{y}}_{\text{LSTM}}$ represent the features of the embedding component, the CNN component and the LSTM component, respectively, $\mathbf{W}$ is the joint weights of the above components, and $\mathbf{b}$ is the bias term. It is worth mentioning that the last layer of each component is a dense and batch-normalization layer with identical size (see Fig. 3). Finally, we concatenate the embedding, CNN and LSTM components.

### C. Computational Complexity

The computational complexity of our DELA can be estimated by summarizing up the computational costs of the embedding, CNN and LSTM components, respectively.

*1) Computational Cost of Embedding Component:* As shown in [41], [45], the computational cost of embedding component mainly depends on the dimension of input categorical vector and the embedding dimension. In particular, we denote the computational cost of the embedding component by $N_{\text{Embedding}}$, which can be calculated by

$$N_{\text{Embedding}} = \mathcal{N}_v \times \mathcal{N}_v \times \mathcal{D}_e \times \mathcal{D}_e, \tag{10}$$

where $\mathcal{N}_v$ is the number of categorical features and $\mathcal{D}_e$ is the embedding dimension. The calculation is mainly based on the concatenation of categorical feature vectors according to the embedding dimension [41].

*2) Computational Cost of CNN Component:* The standard convolution computational cost per time step in CNN is $O(N_{\text{CNN}})$, where $N_{\text{CNN}}$ is the total number of parameters in a CNN [46], [47]. In particular, $N_{\text{CNN}}$ can be calculated $N_{\text{CNN}} = \mathcal{C}_i \times \mathcal{C}_o \times \mathcal{D}_k^2 \times \mathcal{D}_f^2$, where $\mathcal{C}_i$ is the number of input channels, $\mathcal{C}_o$ is the number of output channels, $\mathcal{D}_k^2$ is the kernel size and $\mathcal{D}_f^2$ is the feature map size.

*3) Computational Cost of LSTM Component:* It is shown in [48] that the learning computational complexity per time step in LSTM is $O(N_{\text{LSTM}})$, where $N_{\text{LSTM}}$ is the total number of parameters in a standard LSTM network. In particular, $N_{\text{LSTM}}$ can be calculated by Eq. (11),

$$N_{\text{LSTM}} = \mathcal{N}_c^2 \times 4 + \mathcal{N}_i \times \mathcal{N}_c \times 4 + \mathcal{N}_c \times \mathcal{N}_o + \mathcal{N}_c \times 3, \tag{11}$$

where $\mathcal{N}_i$ is the number of input units, $\mathcal{N}_c$ is the number of memory cells, and $\mathcal{N}_o$ is the number of output units.

TABLE I
DATA INFORMATION

| Description | Value |
|---|---|
| Time window for traffic flow | July 19, 2016 - Oct 17, 2016 |
| The dimensionality of 2-D input data | (24,6) |
| Travel time | within range of [9.26, 6771.11] |
| # of routes | 6 |
| # of links | 24 |
| # of categorical features of weather condition | 7 |
| # of driving records | 1,527,136 |
| # of vehicles | 77,012 |
| # of categorical features of hour | 24, within range of [00:00, 23:59] |
| # of categorical features of day | 7, with range of [0, 6] |

## V. EXPERIMENTS

In this section, we conduct the experiments to evaluate the performance of DELA. In Section V-A, we describe basic experimental settings. Section V-B presents the performance comparison of our proposed Embedding & Deep Learning approach with other existing methods. We then investigate the efficiency and convergence of DELA in Section V-C. Section V-D presents the discussion of the results.

### A. Experimental Setup

*1) Dataset Description:* We conduct our experiments on a real highway traffic dataset. It contains nearly 3 months traffic flow information (from July. 19, 2016 to Oct. 17, 2016), and collects 1,527,136 road driving records with 77,012 vehicles. This dataset was formally released by Knowledge Discovery and Data Mining Tools Competition (KDD CUP 2017), which can be downloaded from Tian Chi Platform.[3] The metadata information about the dataset is summarized in Table I, where # means "the number of". In particular, we convert the 1-D data to the 2-D data with the dimensionality of $24 \times 6$, where the number of links is 24 and there are 6 values because the travel time was taken over each link every 20 minutes within 2 hours. It is worth noting that the travel time falls in the range of [9.26, 6771.11]. The minimum value is 9.26 seconds while the maximum value is 6771.11 seconds. In addition, the mean value is about 93.5 seconds. It implies that the distribution of travel time is extremely uneven.

[3]https://tianchi.aliyun.com/datalab/dataSet.html?spm=5176.100073.0.0.63696fc1cVfp6R&dataId=60

TABLE II
WEATHER CATEGORICAL FEATURES

| Description | Value | Range (before discretization) |
|---|---|---|
| # of categorical features of air pressure | 5 | [993.4,1018.4] |
| # of categorical features of temperature | 20 | [14.1,39.4] |
| # of categorical features of wind speed | 10 | [0,7.5] |
| # of categorical features of sea pressure | 10 | [998.2,1023.5] |
| # of categorical features of wind direction | 10 | [0,999017.0] |
| # of categorical features of humidity | 10 | [27.0,98.0] |
| # of categorical features of precipitation | 10 | [0,27.2] |

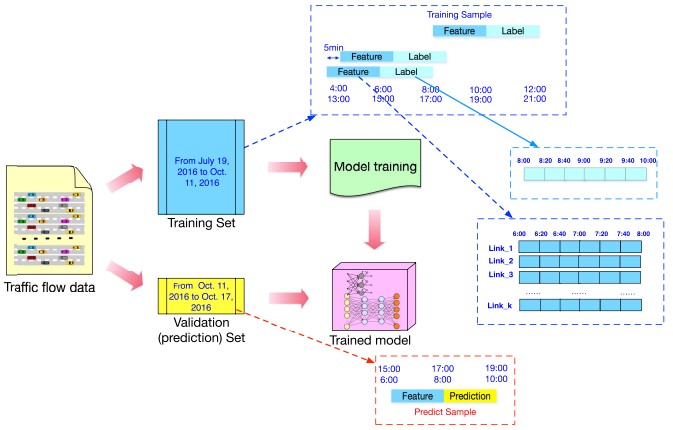



Fig. 4. Training and prediction sets.

Table II summarizes the weather indicators, which include atmospheric pressure, pressure at sea level, wind direction, wind speed, air temperature, relative humidity and precipitation. These weather indicators are collected every three hours. In order to convert categorical data into a numerical form denoted by embedding vectors (as mentioned in Section IV-A), we first discretize each of weather categorical data (i.e., indicators) and then exploit one-hot encoding to represent these categorical features.

*2) Training Settings:* In this paper, we aim to predict the travel time in the next two hours based on the traffic flow data. The training procedure can be divided into two phases: 1) the training phase and 2) the validation (prediction) phase as shown in Fig. 4. In particular, the training data set contains the traffic flow data from July 19, 2016 to Oct. 11, 2016 and the validation data set contains the data from Oct. 11, 2016 to Oct. 17, 2016. We essentially use the first two hours as the training data to predict the average travel time in the next two hours. The key step in the training phase is data sampling. Since the large number of data samples is beneficial to DELA, we attempt to obtain as many training samples as possible. However, it may take a long time to obtain a large number of data samples. In order to balance the efficiency and the accuracy of training DELA model, we choose 5 minutes as the sliding window length. For example, if a sample with the feature within the time period is [6:00,8:00) and the label within the time period is [8:00,10:00), then the next sample with the feature will be within the time period [6:05,8:05) and the next sample with the label will be within the time period [8:05,10:05).

*3) Performance Metrics:* The prediction accuracy is typically evaluated by mean absolute error (MAE), mean absolute percentage error (MAPE) and Root Mean Square Error (RMSE) [7], [8], [49]. Therefore, we also use these three metrics to evaluate the proposed approach and other baseline approaches.

MAE is the absolute value of the difference between the predicted value and the actual value. In particular, we define MAE as follows,

$$\text{MAE} := \frac{\sum_{i=1}^{N} |y_i - \hat{y}_i|}{N}, \tag{12}$$

where $y_i$ denotes the observed real-world travel time value of the $i$-th sample, $\hat{y}_i$ is the predicted travel time value, $N$ is the number of all predicted values. It is relatively easy to calculate MAE while MAE can also result in the bias measurement when there are some observed values significantly different from other values (these values are also named as *outliers*). It is shown in [50] that a single large error can sometimes dominate the calculation of MAE.

MAPE represents the prediction error in a percentage and is usually used in the case of identifying the significant error when MAE is quite small [51]. We define MAPE as follows,

$$\text{MAPE} := \frac{1}{n}\sum_{i=1}^{N} |\frac{y_i - \hat{y}_i}{y_i}|. \tag{13}$$

RMSE is defined as follows,

$$\text{RMSE} := \sqrt{\frac{\sum_{i=1}^{N} |y_i - \hat{y}_i|^2}{N}}. \tag{14}$$

The lower values of MAE, MAPE and RMSE imply higher accuracy of models.

## B. Performance Evaluation

We conduct extensive experiments to evaluate our DELA by comparing with other representative baseline approaches. In particular, we categorize the experiments into three groups: 1) the comparison between *1-D road* traffic flow input and *2-D link* traffic flow input; 2) the comparison between the 2-D link traffic flow with categorical factors and the 2-D link traffic flow dataset without categorical factors; 3) the performance evaluation on our DELA with consideration of the impacts of parameters.

*1) 1-D vs 2-D:* Most previous studies only analyze the traffic flow at the route level which is fairly coarse and cannot capture the impacts of other conditions like route structure. In our DELA, we conduct a fine-grained analysis on the traffic flow at link level. In particular, we convert the 1-D route traffic data to the 2-D link traffic data. We choose the following baseline approaches and compare the performance of them with 1-D route traffic data and 2-D link traffic data, respectively. It is worth mentioning that both ARIMA and BPNN approaches can only accept 1-D route traffic data.

- **ARIMA** (1-D/2-D): This method is a common model in the field of time series prediction. which is widely used in the early research period.

TABLE III
PERFORMANCE COMPARISON OF DELA WITH BASELINE METHODS

| | FTP = 60 minutes | | | FTP = 120 minutes | | |
|---|---|---|---|---|---|---|
| **Methods** | **MAPE** | **MAE** | **RMSE** | **MAPE** | **MAE** | **RMSE** |
| ARIMA (1-D) | 0.3186 | 39.19 | 55.17 | 0.4474 | 52.59 | 66.87 |
| ARIMA (2-D) | 0.2977 | 35.78 | 52.03 | 0.4216 | 48.95 | 63.72 |
| BPNN (1-D) | 0.1895 | 26.27 | 45.98 | 0.1904 | 26.34 | 46.01 |
| BPNN (2-D) | 0.1866 | 25.91 | 45.80 | 0.1878 | 25.60 | 45.76 |
| RNN (1-D) | 0.1873 | 25.44 | 45.53 | 0.1852 | 25.31 | 45.47 |
| RNN (2-D) | **0.1828** | **25.31** | **45.87** | **0.1824** | **25.06** | **45.22** |
| LSTM (1-D) | 0.1815 | 26.17 | 45.88 | 0.1847 | 25.13 | 45.09 |
| LSTM (2-D) | **0.1801** | **25.23** | **45.01** | **0.1802** | **25.00** | **44.89** |
| CNN (2-D) | **0.1808** | **25.77** | **45.61** | **0.1826** | **25.18** | **45.32** |
| CNN (2-D) & LSTM (2-D) | **0.1795** | **25.39** | **44.97** | **0.1797** | **25.03** | **44.82** |
| Embedding & LSTM (2-D) | 0.1781 | 25.63 | 44.76 | 0.1784 | 24.56 | 44.73 |
| Embedding & CNN (2-D) | 0.1785 | 25.42 | 44.87 | 0.1769 | 24.71 | 44.98 |
| Embedding & CNN & LSTM (2-D) | **0.1721** | **24.37** | **43.19** | **0.1707** | **23.31** | **43.11** |

- **BPNN** (1-D/2-D): Many prior studies on traffic flow forecasting have been based on BPNN.
- **RNN** (1-D/2-D): This method can capture the time series characteristics.
- **LSTM** (1-D/2-D): It is an improved version of RNN. It has the advantages of capturing the long temporal feature of input time series. Recently, LSTM has been applied in traffic flow forecasting.
- **CNN (2-D)**: In this paper, we use this method to learn the 2-D link traffic flow. Since CNN cannot process 1-D route traffic data, we convert the 1-D route traffic data into the 2-D link traffic flow data, which can be processed by CNN. The CNN can be work and deal with the relationship of different links through convolution operations.
- **CNN (2-D) & LSTM (2-D)**: This method can also be regarded as a special case of our DELA with the removal of Embedding component. We conduct experiments mainly to evaluate the effect of embedding component.

TABLE IV
EXPERIMENT PARAMETERS FOR RESULTS OF TABLE III AND OTHER GROUPS OF EXPERIMENTS IF NO SPECIFICATION

| **Method** | **Parameters** |
|---|---|
| ARIMA | (d,p,q) = (0,1,1) |
| BPNN | a fully-connect layer<br>neurous numbers: 128 |
| RNN | layer number: 3<br>neurous numbers: 128 |
| LSTM | layer number: 3<br>neurons numbers: 128 |
| CNN | layer number: 3<br>filter size: (3x3)<br>filter numbers: 64 |
| Embedding | dimensionality: 18 |

Table III shows the experimental results when forecasting future time period (FTP) is equal to 60 minutes and 120 minutes, respectively. In particular, Table IV summarizes the experimental parameters used in experiments (we fix them in other groups of experiments if no further specification).

It is shown in Table III that RNN, LSTM and CNN models outperform ARIMA and BPNN (see bold-font values). The improvement of RNN, LSTM and CNN models mainly owes to the *enhanced learning capability from deep neural networks*. Moreover, when comparing RNN (1-D), LSTM (1-D) with RNN (2-D), LSTM (2-D), respectively, we find that both RNN (2-D) and LSTM (2-D) perform better than those with 1-D inputs. This is because the *generalization* of RNN and LSTM models can be enhanced with 2-D input data. In addition, we also find that LSTM is slightly better than RNN. It is possible that LSTM effectively preserves *long-term memory* and will show the advantage in learning travel time data.

Furthermore, in order to make a fair comparison with conventional methods like ARIMA and BPNN, we also convert 1-D traffic flow data to 2-D traffic flow data (at link-level) and put them into ARIMA and BPNN consequently obtaining the 2-D data results for ARIMA and BPNN [see ARIMA (2-D) and BPNN (2-D) in Table III] after averaging prediction metrics at all the links along the route. It is shown in Table III that ARIMA (2-D) and BPNN (2-D) outperform ARIMA (1D) and BPNN (1-D). This is mainly due to the improvement brought by the fine-grained analysis on link-level traffic data.

Last by not least, we also conduct experiments with a slightly-modified version of our DELA with the removal of embedding component. We name such model as LSTM (2-D)

& CNN (2-D) model. It is shown in Table III that LSTM (2-D) & CNN (2-D) model outperforms other existing models; this may owe to the improved learning capability from both LSTM and CNN models on 2-D link-level data.

*2) With Categorical Factors vs Without Categorical Factors:* Most previous studies failed to consider the impacts of categorical factors on the traffic flow. In our DELA, we design an embedding component that takes categorical factors into account. In the 2nd group of experiments, we evaluate the performance of the following models.

- **Embedding & LSTM (2-D)**: This method can be regarded as a special case of our DELA with the removal of CNN component. We conduct experiments mainly to evaluate the effect of embedding & LSTM (2-D) in contrast to LSTM (2-D) model only.
- **Embedding & CNN (2-D)**: This method can also be regarded as a special case of our DELA with the removal of LSTM component. We conduct experiments mainly to evaluate the effect of embedding & CNN (2-D) in contrast to CNN (2-D) model only.

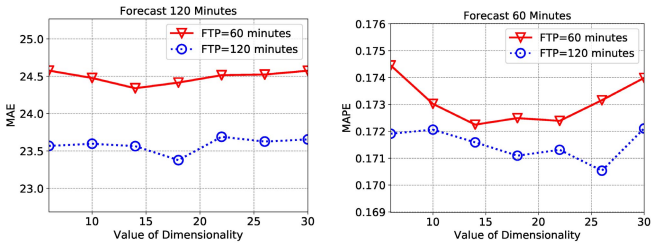
It is shown Table III that both Embedding & LSTM and Embedding & CNN models perform better than LSTM only and CNN only. For example, when FTP is 60 minutes, compared with CNN (2-D) model, the proposed DELA can improve the prediction accuracy by reducing MAPE nearly 4.8% (obtained by $(0.1808-0.1721)/0.1808$), reducing MAE by nearly 5.4% and reducing RMSE by nearly 5.3%. Compared with LSTM (2-D) model, DELA reduces MAPE by 4.4%, MAE by 3.4%, and RMSE by 4.0%.

This implies that using the additional categorical features can greatly improve the prediction accuracy since the embedding component can effectively learn these categorical features. Moreover, we also observe that Embedding & LSTM slightly outperforms Embedding & CNN in short time forecasting task, e.g., when FTP is 60 minutes. On the contrary, Embedding & CNN model performs slightly better than Embedding & LSTM model in long time forecasting task, e.g., when FTP is 120 minutes. The experimental results also show that *our DELA outperform other existing models due to the integration of the embedding component, the LSTM component and the CNN component* as shown in the last row in Table III.

*3) Parameter Study:* We next evaluate the impacts of parameters of our DELA. In particular, we first consider impacts of parameters $\alpha$, $\beta$ and $\gamma$ independently. We then jointly investigate the optimal values of them.

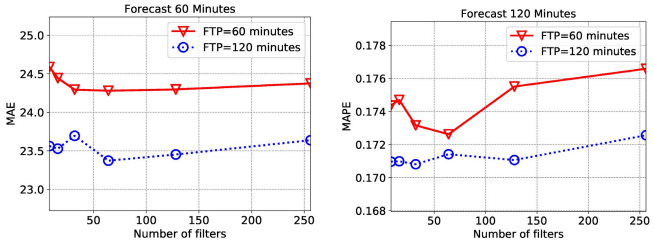
*a) Impact of $\alpha$:* $\alpha$ is a parameter controlling the dimensionality of embedding vector in the embedding component. To investigate the impact of $\alpha$ on the prediction results, we vary the values of $\alpha$ from 6 to 30 with the step value of 2. At the same time, we fix $\beta = 64$ and $\gamma = 128$. We conduct two groups of experiments with FTP equal to 60 minutes and 120 minutes, respectively.

Fig. 5 shows the experimental results. When FTP is 60 minutes, we can find that both MAE and MAPE decrease at first when the number of dimensionality increases. when $\alpha$ is greater than 14. MAE and MAPE increase when the number of dimensionality increases. It also shows that the optimal value of dimensionality is 14 at FTP $= 60$ minutes. When FTP is 120 minutes, the optimal values of dimensionality are 16 and 26 for MAE and MAPE, respectively. Moreover, when FTP becomes longer, the optimal value of dimensionality also increases. It may owe to the fact that higher-dimensional information is needed to express long-term information.




(a) (b)

Fig. 5. Impact of $\alpha$. (a) MAE. (b) MAPE.



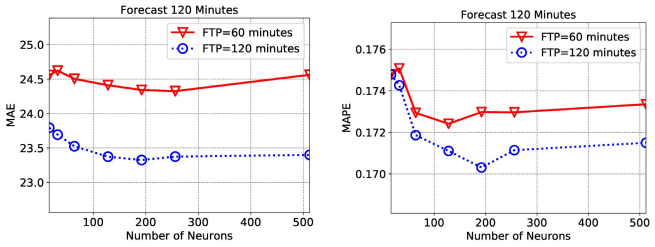

(a) (b)

Fig. 6. Impact of $\beta$. (a) MAE. (b) MAPE.

*b) Impact of $\beta$:* $\beta$ is a parameter controlling the number of filters in the CNN component. To investigate the impact of $\beta$ on the prediction results, we vary the values of $\beta$ according to the following set of values $(8, 16, 32, 64, 128, 256)$. At the same time, we fix $\alpha = 15$ and $\gamma = 128$. We also conduct two groups of experiments with FTP with 60 minutes and 120 minutes, respectively.

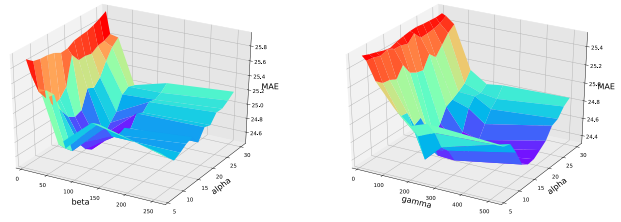
Fig. 6 shows the experimental results. When FTP is 60 minutes, we can also find both MAE and MAPE decrease at first when the number of filters increases. When $\beta$ is greater than 64, both MAE and MAPE increase when the number of filters increases. It shows that the optimal value of $\beta$ is 64 at FTP $= 60$ minutes. When FTP is 120 minutes, the optimal values of $\beta$ are 64 and 32 for MAE and MAPE, respectively. It implies that DELA obtains an excellent performance with few filters in the case. It may owe to the fact that the high travel time value has a great impact on the performance of CNN model.

*c) Impact of $\gamma$:* $\gamma$ is a parameter controlling the number of neurons in LSTM component. To investigate the impact of $\gamma$ on the prediction results, we vary the values of $\gamma$ according to the set of values $(16, 32, 64, 128, 192, 256, 512)$. At the same time, we fix $\alpha = 15$ and $\beta = 64$. We also conduct two groups of experiments with FTP equal to 60 minutes and 120 minutes, respectively.




(a) (b)

Fig. 7. Impact of $\gamma$. (a) MAE. (b) MAPE.




(a) (b)

Fig. 8. Impact of $\alpha$, $\beta$ and $\gamma$. (a) MAE. (b) MAE.

Fig. 7 shows the experimental results. When FTP is 120 minutes, we can also find both MAE and MAPE decrease at first when the number of neurons increases. When $\gamma$ is greater than 192, both MAE and MAPE increase when the number of neurons increases. It also shows that the optimal value of $\gamma$ is 192 at FTP $=$ 120 minutes. In addition, both MAE and MAPE of the model change little after 192. When the FTP is 60 minutes, the optimal values of $\gamma$ are 256 and 128 for MAE and MAPE, respectively.

*d) Joint investigation of optimal values of $\alpha$, $\beta$ and $\gamma$:* Fig. 7(a) shows that MAE reaches the minimum when $\gamma$ is 128 (no progress made after $\gamma$ exceeds 128). Therefore, we first fix $\gamma$ at 128 and conduct another group of experiments. We plot the results of MAE against $\alpha$ and $\beta$ in Fig. 8 (a). It is shown in Fig. 8(a) that the minimum value of MAE is obtained when $\alpha$ is around 15 and $\beta$ is around 64. Similarly, we observe from Fig. 6(a) that MAE is the minimum when $\beta$ is 64 (no progress made after $\beta$ exceeds 64). Thus, we fix $\beta$ at 64 and conduct experiments again. We plot the results of MAE against $\gamma$ and $\alpha$ in Fig. 8(b). It is shown in Fig. 8(b) that the minimum value of MAE is obtained when $\gamma$ is around 128 and $\alpha$ is around 15. Finally, we obtain the optimal values of $\alpha$, $\beta$ and $\gamma$ at $(15, 64, 128)$ after jointly considering both Fig. 8(a) and Fig. 8(b) together.

### C. Efficiency and Convergence of DELA

The efficiency of travel time prediction plays an important role in ITS. We next evaluate the efficiency of our DELA in terms of training time and prediction accuracy.

We first compare the training time of the proposed DELA with other baseline approaches like ARIMA, BPNN, RNN, LSTM, CNN with both 1-D and 2-D traffic flow data. It is worth mentioning that ARIMA (1-D/2-D) model ran in a PC with single thread CPU (no GPU enabling) and other approaches ran in a PC with GPU enabling (i.e., one GeForce GTX 1080 Ti graphic card installed).

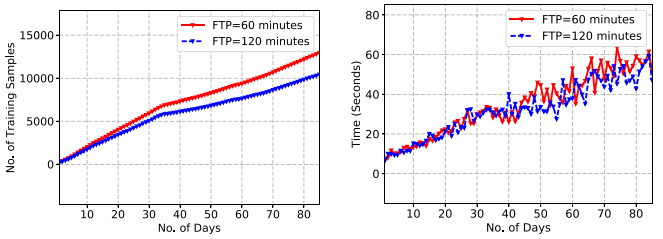
TABLE V
TRAINING TIME COMPARISON WITH EXISTING MODELS (FTP $=$ 120 MINUTES)

| Model | Training time (seconds) |
|---|---|
| ARIMA(1-D) | 1.18 |
| ARIMA(2-D) | 5.75 |
| BPNN(1-D) | 10.78 |
| BPNN(2-D) | 48.57 |
| RNN(1-D) | 34.81 |
| RNN(2-D) | 40.85 |
| LSTM(1-D) | 36.17 |
| LSTM(2-D) | 45.31 |
| CNN(2-D) | 14.19 |
| Embedding & LSTM (2-D) | 53.12 |
| Embedding & CNN (2-D) | 16.68 |
| Embedding &CNN & LSTM (2-D) | 60.18 |

Table V shows the results. It is shown in Table V that it takes the minimum training time for ARIMA while it takes the longest time for the proposed DELA; this is mainly due to the complexity of the model. However, it takes about only one minute to train the DELA model at the current experimental data scale (i.e., 3-month traffic flow data, 6 roads, and 24 links) while DELA achieves the superior performance compared with other existing methods (as shown in Section V-B). As shown in Section IV-C, the computational time heavily depends on the input size, output size and the model parameters. Therefore, the computational time will increase with the increment of data scale. In a large scale ITS, it may take a long training time for DELA though this disadvantage can be overcome by joint training at cloud servers, edge servers and end devices as shown in recent work [52]. Moreover, the prediction time of our DELA is pretty small (to be shown later) and it can fulfill the basic requirement of real-time prediction in ITS since the prediction time plays a more important role than the training time in realistic application of ITS.

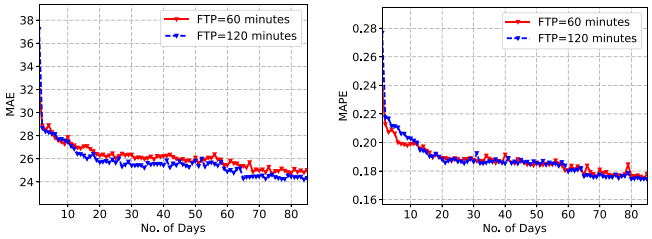
The second group of experiments was conducted to investigate the impact of the number of days on the training set size and the training time. We choose $\mathcal{N}$ days among 85 days between July 19, 2016 and Oct 17, 2016 to train the model. In particular, Fig. 9(a) shows that the number of training samples increases with the increased number of days, implying the increased computational cost. Fig. 9(b) shows the training time (seconds) versus the number of days. It is shown in Fig. 9(b) that the training time increases with the increased number of days though the maximum training time is about 60 seconds (achieved on $\mathcal{N} = 85$). It implies that the training time consumption of our DELA is relatively small.

We then conduct another group of experiments to investigate the impact of the number of days on the accuracy of our DELA. Fig. 10 shows MAE and MAPE versus the number of days in the training set. We observe from Fig. 10 that the



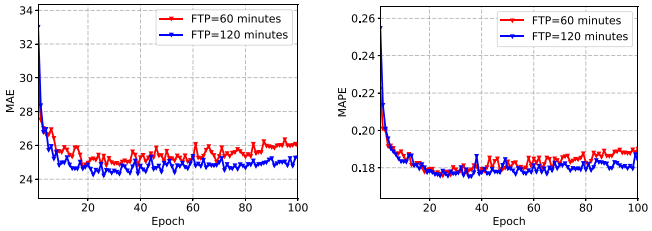

(a) (b)

Fig. 9. Impact of number of days with step value 1. (a) No. of training samples vs. No. of days. (b) Training time vs. No. of days.




(a) (b)

Fig. 10. Impact of number of days on prediction accuracy with step value 1. (a) MAE vs. No. of days. (b) MAPE vs. No. of days.




(a) (b)

Fig. 11. Convergence analysis of DELA. (a) MAE vs. epoch. (b) MAPE vs. epoch.

increased number of days results in the significant reduction on MAE and MAPE (i.e., the improved prediction accuracy). In particular, when the number of days is about 65, the minimum values of MAE and MAPE are achieved while this trend tends to be even and smooth after 65. This finding shows that an optimal prediction accuracy can be achieved when $\mathcal{N} = 65$ is chosen (nearly two months), implying that we may not need the total number of 85 days to train the model.

We next investigate the convergence of our DELA. In particular, we choose the number of epochs as the parameter to control the training round. An epoch is defined by one forward pass and one backward pass of all training samples. Fig. 11 shows the results. Note that the dimensionality of embedding vector is 18, the filter number in a convolutional layer is 64 and the number of cells of LSTM is 128. We observe from Fig. 11 that both MAE and MAPE decrease with the increased number of epochs while the optimal values of MAE and MAPE achieve when the number of epochs is about 20.

Real-time prediction is a critical issue in ITS. We also evaluate the real-time performance of our DELA in terms

TABLE VI
PREDICTION TIME OF DELA WHEN FTP = 120 MINUTES

| Method | Prediction Time (seconds) |
|---|---|
| Embedding & CNN | 0.09198 |
| Embedding & LSTM | 0.45963 |
| Embedding & CNN & LSTM | 0.49038 |

TABLE VII
NO. OF PARAMETERS WHEN FTP = 120 MINUTES

| Component | No. of parameters |
|---|---|
| Embedding | 2,484 |
| CNN | 44,486 |
| LSTM | 336,176 |

of prediction time. In particular, we conduct another group of experiments on the prediction time of the learned models of Embedding & CNN, Embedding & LSTM, Embedding & CNN & LSTM when FTP = 120 minutes. Note that the prediction time is essentially the time spent on a single run on one input. Table VI shows that Embedding & CNN has the shortest prediction time while Embedding & CNN & LSTM has the longest prediction time. It implies that LSTM consumes a large portion of prediction time. This is mainly because the large number of parameters of LSTM component results in the high computational cost agreeing with our analysis in Section IV-C.

Table VII shows that LSTM component has the largest number of parameters. Our result also shows that our DELA can fulfill the basic requirement on real-time prediction in practical ITS (e.g., the threshold of 0.5 second is necessary to avoid traffic conflicts in [53]). It is worth mentioning that our experiment is based on FTP = 120. In fact, the prediction time can be even smaller when we choose a smaller FTP value.

### *D. Discussion*

The extensive experimental results show that our DELA outperforms other existing models in terms of prediction accuracy (i.e., lowest values of MAE and MAPE). Meanwhile, a well-trained DELA model can predict the travel time within 0.5 second (i.e., nearly real-time prediction). Moreover, DELA may perform well in larger scale road networks due to the following reasons: 1) CNN component of DELA is good at learning the *fine-grained link-level* data. The input 2-D traffic flow data is beneficial to CNN component because CNN is good at extracting features in the local region while there exist the correlation between two adjacent rows (i.e., two adjacent links) and the correlation between two adjacent columns (i.e., the traffic flow of two adjacent time slots along one link) in the input 2-D traffic data. Meanwhile, there are 0's in the other non-existing links which has no effect on CNN. Therefore, the merit of CNN can help DELA to learn traffic flow data even in larger scale road networks (as there is always a correlation between two adjacent links along a road

or between two time slots along a link). 2) LSTM component of DELA can effectively extract *time-series features* from the input traffic data. Recent advances in [44] show that LSTM is effective and scalable to learn time-series data such as traffic flow data. Therefore, the feature of LSTM can also make DELA be scalable to large scale traffic flow dataset.

However, there are several limitations in the proposed DELA:

- *Poor explanation capability* of deep learning models. Our DELA has shared one of the common limitations with other deep learning models - the poor model interpretability as shown in [10] though the embedding component in our DELA can partially improve the explanation capability of deep learning models. Therefore, we will improve the explanation capability of our DELA by adopting other machine learning models in the future.
- *Limited learning capability* of embedding component. The embedding component in our DELA only learns weather-related categorical features such as air pressure, temperature and wind speed from one city in one season (as the given dataset only discloses such information). However, different cities may have different weather characteristics in different seasons. Thus, the embedding component in our DELA is poor in generalization of weather features due to the lack of characteristics of weather information from different cities in different seasons. In the future, we will improve DELA by learning weather information from other available datasets related to the weather in the current city.

## VI. Conclusion

In this paper, we propose a Deep and Embedding Learning Approach (DELA) to predict the traffic flow. In particular, our DELA consists of an embedding component, a CNN component and an LSTM component. The embedding component can capture the categorical feature information, such as the structure of routes and weather information. Moreover, the CNN component can learn the 2-D link traffic flow information while LSTM has the advantages of maintaining a long-term memory. We conduct extensive experiments on real traffic dataset to evaluate the performance of DELA. The experiment results show that our proposed DELA outperforms existing methods in terms of prediction accuracy.

Regarding future work, we will improve the explanation power of deep learning models via the adoption of other machine learning models. Meanwhile, we will enhance our DELA by learning from other traffic flow datasets. Moreover, we will extend DELA to jointly investigate other traffic flow metrics, such as traffic volume and traffic speed together.

## References


[1] Y. Tan, H. Xu, L. Jiao, J. J. Ochoa, and L. Shen, "A study of best practices in promoting sustainable urbanization in China," *J. Environ. Manage.*, vol. 193, pp. 8–18, May 2017.

[2] B. Yu, X. Song, F. Guan, Z. Yang, and B. Yao, "$k$-nearest neighbor model for multiple-time-step prediction of short-term traffic condition," *J. Transp. Eng.*, vol. 142, no. 6, 2016, Art. no. 04016018.

[3] E. Ko, J. Ahn, and E. Kim, "3D Markov process for traffic flow prediction in real-time," *Sensors*, vol. 16, no. 2, p. 147, 2016.

[4] F. Jin and S. Sun, "Neural network multitask learning for traffic flow forecasting," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Jun. 2008, pp. 1897–1901.

[5] K. Y. Chan, T. S. Dillon, J. Singh, and E. Chang, "Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and Levenberg–Marquardt algorithm," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 644–654, Jun. 2012.

[6] D. Chen, "Research on traffic flow prediction in the big data environment based on the improved RBF neural network," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 2000–2008, Aug. 2017.

[7] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 5, pp. 2191–2201, Oct. 2014.

[8] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015.

[9] Y. Duan, Y. Lv, Y.-L. Liu, and F. Wang, "An efficient realization of deep learning for traffic data imputation," *Transp. Res. C, Emerg. Technol.*, vol. 72, pp. 168–181, Nov. 2016.

[10] N. G. Polson and V. O. Sokolov, "Deep learning for short-term traffic flow prediction," *Transp. Res. C, Emerg. Technol.*, vol. 79, pp. 1–17, Jun. 2017.

[11] H. Liu, T. Taniguchi, Y. Tanaka, K. Takenaka, and T. Bando, "Visualization of driving behavior based on hidden feature extraction by using deep learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 9, pp. 2477–2489, Sep. 2017.

[12] Z. Zhao *et al.*, "LSTM network: A deep learning approach for short-term traffic forecast," *IET Intell. Transp. Syst.*, vol. 11, no. 2, pp. 68–75, 2017.

[13] Y.-Q. Wang and J. Luo, "Study of rainfall impacts on freeway traffic flow characteristics," *Transp. Res. Procedia*, vol. 25, pp. 1533–1543, Jan. 2017.

[14] L. Lin, J. Li, F. Chen, J. Ye, and J. Huai, "Road traffic speed prediction: A probabilistic model fusing multi-source data," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 7, pp. 1310–1323, Jul. 2018.

[15] M. van der Voort, M. Dougherty, and S. Watson, "Combining Kohonen maps with ARIMA time series models to forecast traffic flow," *Transp. Res. C, Emerg. Technol.*, vol. 4, no. 5, pp. 307–318, 1996.

[16] H. Grubb and A. Mason, "Long lead-time forecasting of UK air passengers by Holt–Winters methods with damped trend," *Int. J. Forecasting*, vol. 17, no. 1, pp. 71–82, 2001.

[17] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, Nov. 2003.

[18] A. Abadi, T. Rajabioun, and P. A. Ioannou, "Traffic flow prediction for road transportation networks with limited traffic data," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 653–662, Apr. 2015.

[19] T. M. Dantas, F. L. C. Oliveira, and H. M. V. Repolho, "Air transportation demand forecast through Bagging Holt Winters methods," *J. Air Transp. Manage.*, vol. 59, pp. 116–123, Mar. 2017.

[20] A. Stathopoulos and G. M. Karlaftis, "A multivariate state space approach for urban traffic flow modeling and prediction," *Transp. Res. C, Emerg. Technol.*, vol. 11, no. 2, pp. 121–135, 2003.

[21] I. Okutani and Y. J. Stephanedes, "Dynamic prediction of traffic volume through Kalman filtering theory," *Transp. Res. B, Methodol.*, vol. 18, no. 1, pp. 1–11, 1984.

[22] A. S. Nair, J.-C. Liu, L. Rilett, and S. Gupta, "Non-linear analysis of traffic flow," in *Proc. IEEE Intell. Transp. Syst.*, Aug. 2001, pp. 681–685.

[23] E. T. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Temporal evolution of short-term urban traffic flow: A nonlinear dynamics approach," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 23, no. 7, pp. 536–548, 2008.

[24] B. L. Smith and M. J. Demetsky, "Short-term traffic flow prediction models-a comparison of neural network and nonparametric regression approaches," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, vol. 2, Oct. 1994, pp. 1706–1709.

[25] M. Buscema, "Back propagation neural networks," *Substance Use Misuse*, vol. 33, no. 2, pp. 233–270, Jan. 1998.

[26] A.-R. Mohamed, T. N. Sainath, G. Dahl, B. Ramabhadran, G. E. Hinton, and M. A. Picheny, "Deep belief networks using discriminative features for phone recognition," in *Proc. IEEE ICASSP*, May 2011, pp. 5060–5063.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[28] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. AAAI*, 2017, pp. 1655–1661.

[29] J. W. C. van Lint, S. Hoogendoorn, and H. van Zuylen, "Freeway travel time prediction with state-space neural networks: Modeling state-space dynamics with recurrent neural networks," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1811, no. 1, pp. 30–39, 2002.

[30] Y. Gao and M. J. Er, "NARMAX time series model prediction: Feedforward and recurrent fuzzy neural network approaches," *Fuzzy Sets Syst.*, vol. 150, no. 2, pp. 331–350, Mar. 2005.

[31] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. 11th Annu. Conf. Int. Speech Commun. Assoc.*, vol. 2, 2010, pp. 1045–1048.

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

[33] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," in *Proc. 9th Int. Conf. Artif. Neural Netw.*, 1999, pp. 850–855.

[34] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transp. Res. C, Emerg. Technol.*, vol. 54, pp. 187–197, May 2015.

[35] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[36] J. S. Oh, Y. U. Shim, and Y. H. Cho, "Effect of weather conditions to traffic flow on freeway," *KSCE J. Civil Eng.*, vol. 6, no. 4, pp. 413–420, 2002.

[37] D. Akin, V. P. Sisiopiku, and A. Skabardonis, "Impacts of weather on traffic flow characteristics of urban freeways in Istanbul," *Procedia-Social Behav. Sci.*, vol. 16, pp. 89–99, Jan. 2011.

[38] W. H. K. Lam, M. L. Tam, X. Cao, and X. Li, "Modeling the effects of rainfall intensity on traffic speed, flow, and density relationships for urban roads," *J. Transp. Eng.*, vol. 139, no. 7, pp. 758–770, 2013.

[39] Y. Wu, F. Xie, L. Chen, C. Chen, and Z. Zheng, "An embedding based factorization machine approach for Web service qos prediction," in *Proc. Int. Conf. Service-Oriented Comput.*. Cham, Switzerland: Springer, 2017, pp. 272–286.

[40] S. Xie, Z. Zheng, L. Chen, and C. Chen, "Learning semantic representations for unsupervised domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5419–5428.

[41] C. Guo and F. Berkhahn. (2016). "Entity embeddings of categorical variables." [Online]. Available: https://arxiv.org/abs/1604.06737

[42] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 448–456.

[43] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–13.

[44] K. Greff, R. K. Srivastava, J. Koutnìk, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[45] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[46] A. G. Howard *et al.* (2017). "MobileNets: Efficient convolutional neural networks for mobile vision applications." [Online]. Available: https://arxiv.org/abs/1704.04861

[47] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec. 2017.

[48] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc. 15th Annu. Conf. Int. Speech Commun. Assoc.*, 2014, pp. 1–5.

[49] A. Koesdwiady, R. Soua, and F. Karray, "Improving traffic flow prediction with weather information in connected cars: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 9508–9517, Dec. 2016.

[50] C. Chen, J. Twycross, and J. M. Garibaldi, "A new accuracy measure based on bounded relative error for time series forecasting," *PLoS ONE*, vol. 12, no. 3, 2017, Art. no. e0174202.

[51] R. K. Klimberg, G. P. Sillup, K. J. Boyle, and V. Tavva, "Forecasting performance measures—What are their practical meaning?" in *Advances in Business and Management Forecasting*. Bingley, U.K.: Emerald, 2010, pp. 137–147.

[52] S. Teerapittayanon, B. McDanel, and H. T. Kung, "Distributed deep neural networks over the cloud, the edge and end devices," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2017, pp. 328–339.

[53] C. Katrakazas, M. Quddus, and W. H. Chen, "A simulation study of predicting real-time conflict-prone traffic conditions," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 10, pp. 3196–3207, Oct. 2018.


**Zibin Zheng** received the Ph.D. degree from The Chinese University of Hong Kong in 2011. He is currently a Full Professor with Sun Yat-sen University, Guangzhou, China. His research interests include service computing, software engineering, and blockchain. He was a recipient of the IBM Ph.D. Fellowship Award. He received the ACM SIGSOFT Distinguished Paper Award at the ICSE 2010 and the Best Student Paper Award at the ICWS 2010.

**Yatao Yang** received the B.S. degree from Zhengzhou University, Zhengzhou, China, in 2014, and the M.S. degree from Sun Yat-Sen University, Guangzhou, China, in 2017, where he is currently pursuing the Ph.D. degree with the School of Data and Computer Science. His research interests include machine learning, data mining, and service computing.

**Jiahao Liu** received the B.S. degree from Sun Yat-sen University, Guangzhou, China, in 2015, where he is currently pursuing the M.S. degree with the School of Data and Computer Science. His research interests include machine learning and data mining.

**Hong-Ning Dai** received the Ph.D. degree in computer science and engineering from the Department of Computer Science and Engineering, Chinese University of Hong Kong. He is currently an Associate Professor with the Faculty of Information Technology, Macau University of Science and Technology. He also holds visiting positions at the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, and the School of Electrical Engineering and Telecommunications, the University of New South Wales. His research interests include cyber-physical systems, wireless networks, and blockchain. He is an Editorial Board Member of the *International Journal of Industrial Engineering Computations*, the *International Journal of Wireless and Mobile Communication for Industrial Systems*, and the *International Journal of Data and Network Science*. He has served as a Guest Editor for the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS.

**Yan Zhang** received the Ph.D. degree with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore. He is currently a Full Professor with the Department of Informatics, University of Oslo, Norway. His current research interests include next-generation wireless networks leading to 5G, green, and secure cyber-physical systems (e.g., smart grid, healthcare, and transport). He is also a Senior Member of the IEEE ComSoc, the IEEE CS, the IEEE PES, and the IEEE VT Society. He is a fellow of the IET. He serves as a TPC member for numerous international conferences, including the IEEE INFOCOM, the IEEE ICC, the IEEE GLOBECOM, and the IEEE WCNC. He received the 2018 Highly Cited Researcher Award (top 1% by citations) according to Clarivate Analytics. He served in chair positions in a number of conferences, including the IEEE GLOBECOM 2017, the IEEE VTC-Spring 2017, the IEEE PIMRC 2016, the IEEE CloudCom 2016, the IEEE ICCC 2016, the IEEE CCNC 2016, the IEEE SmartGridComm 2015, and the IEEE CloudCom 2015. He is an Associate Technical Editor of the *IEEE Communications Magazine*, an Editor of the IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING, the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, and the IEEE INTERNET OF THINGS JOURNAL, and an Associate Editor of the IEEE ACCESS. He is the IEEE Vehicular Technology Society (VTS) Distinguished Lecturer.