TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 06/11 pp52–64
DOI: 10.26599/TST.2018.9010033
Volume 24, Number 1, February 2019

Geospatial Data to Images: A Deep-Learning Framework for Traffic Forecasting

Weiwei Jiang* and Lin Zhang

Abstract: Traffic forecasting has been an active research field in recent decades, and with the development of deep-learning technologies, researchers are trying to utilize deep learning to achieve tremendous improvements in traffic forecasting, as it has been seen in other research areas, such as speech recognition and image classification. In this study, we summarize recent works in which deep-learning methods were applied for geospatial data-based traffic forecasting problems. Based on the insights from previous works, we further propose a deep-learning framework, which transforms geospatial data to images, and then utilizes the state-of-the-art deep-learning methodologies such as Convolutional Neural Network (CNN) and residual networks. To demonstrate the simplicity and effectiveness of our framework, we present a formulation of the New York taxi pick-up/drop-off forecasting problem, and show that our framework significantly outperforms traditional methods, including Historical Average (HA) and AutoRegressive Integrated Moving Average (ARIMA).

Key words: geospatial data; deep learning; convolutional neural network; residual network; traffic forecasting

1 Introduction

In recent years, smart city has been a new urban development vision, in which information and communication technologies are integrated with all kinds of components in a city, including the transportation systems. The success of the intelligent transportation system and smart city heavily relies on accurate traffic information as well as short-term traffic forecasting. By forecasting the traffic volume, travel speed, and occupancy of the next five to thirty minutes, traffic can be scheduled and planned in advance, which can alleviate traffic congestion and prevent traffic accidents.

The prerequisite of traffic forecasting is the acquisition of accurate traffic information. Because of the development of sensing technologies and the popularity of smart devices, e.g., smart phones and tablets, we can collect and store large amounts of geospatial data, which include massive Global Positioning System (GPS) traces, smart card data, cell phone records, and users' locations and trajectories on social media, such as Facebook and Foursquare check-ins. On the premise of privacy protection, more datasets are being shared for research purposes, e.g., Geolife GPS trajectory dataset[1], Caltrans performance measurement system database[2], and New York trip record data[3].

The large volume of these open datasets necessitates new processing methodologies. In recent years, deep learning has been an outstanding research field, because of the advent of new deep-learning models, increased chip processing abilities (e.g., graphics processing units), and widely available training datasets. Deep-learning methods have been used to

• Weiwei Jiang is with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China. E-mail: jww13@mails.tsinghua.edu.cn.
• Lin Zhang is with the Department of Electronic Engineering and Tsinghua-Berkeley Shenzhen Institute, Tsinghua University, Shenzhen 518055, China. E-mail: linzhang@tsinghua.edu.cn.
* To whom correspondence should be addressed.
Manuscript received: 2017-07-05; accepted: 2017-09-28

solve various problems, such as object recognition[4], handwriting recognition[5], speech recognition[6], image classification[7], and machine translation[8], and have resulted in a significant improvement over traditional methods.

Because large amounts of data can be processed with deep learning, it has been applied to various geospatial traffic forecasting problems, which include bus ridership forecasting[9], crowd flow forecasting[10,11], extreme condition traffic forecasting[12], human trajectories forecasting[13,14], supply-demand forecasting for online car-hailing service[15], taxi destination forecasting[16,17], traffic accident forecasting[18], traffic accident severity forecasting[19], traffic congestion forecasting[20], traffic flow forecasting[21–28], traffic speed forecasting[29–33], and travel time forecasting[34,35]. In this study, we summarize the previous works of deep-learning applications to geospatial traffic forecasting problems by investigating the geospatial datasets and deep-learning models used in these works as well as the traditional methods with which these models were compared.

Based on insights from the previous works, we further propose a deep-learning framework that transforms geospatial data to images, and then utilizes state-of-the-art deep-learning methodologies such as Convolutional Neural Network (CNN)[36] and residual network[37]. In our framework, geospatial data and external factors, such as weather condition, wind speed, and holiday information, are taken as inputs. Then we conduct data transformation and image preprocessing for the geospatial data and feature extraction for the external factors. The processed data are fed into the deep-learning models, which we then use to generate desired predictions.

To demonstrate the simplicity and effectiveness of our framework, we present a formulation of the New York taxi pick-up/drop-off forecasting problem and compare our framework with traditional models such as Historical Average (HA) and AutoRegressive Integrated Moving Average (ARIMA). We also investigate the influences of different parameters, e.g., the length of lookback window and the number of Residual Network Units (RNUs). The experimental results show that compared to the traditional methods, our framework can achieve a remarkable improvement in traffic forecasting.

In summary, our contributions in this study are as follows:

- We review works in which deep-learning models have been used to solve traffic forecasting problems.
- We propose a deep-learning framework which transforms geospatial data to images before utilizing the state-of-the-art deep-learning methodologies and can be applied in different kinds of traffic forecasting problems.
- We take the New York taxi pick-up/drop-off forecasting problem as a case study and demonstrate the simplicity and effectiveness of our framework over traditional methods.

The rest of this paper is organized as follows: Section 2 presents a short introduction to deep learning basics and a summary of previous works in which deep learning methodologies were applied to traffic forecasting problems. Section 3 describes the key components of our framework, including the processes of transforming geospatial data to images and the utilization of state-of-the-art deep-learning methodologies. Section 4 presents the New York taxi pick-up/drop-off forecasting problem as a case study and shows how we applied our framework to this problem. We present our conclusions in Section 5.

2 Related Work

2.1 Deep-learning basics

Deep learning is a class of machine learning techniques, where many layers of information processing stages in hierarchical supervised architectures are exploited for unsupervised feature learning and pattern analysis/classification. The essence of deep learning is to compute hierarchical features or representations of the observational data, where the higher-level features or factors are defined from the lower-level ones[38].

In this section, we briefly introduce Artificial Neural Networks (ANNs), CNNs, and Recurrent Neural Networks (RNNs), which are basic and widely used deep-learning concepts. A systematic introduction of deep learning is given in Refs. [38–40].

2.1.1 Artificial neural network

Artificial neural networks are learning models inspired by biological neural networks that approximate functions that depend on a large number of inputs (features or data representation)[41]. A computing unit or an ANN neuron usually consists of two components: the aggregation function, which calculates the sum of the inputs, and the activation function, which generates

the outputs. Popular choices of the activation function include the logistic sigmoid, hyperbolic tangent (tanh), and Rectified Linear Units (ReLU). A specific example of ANNs is given in Fig. 1. If the neural network has the same number of nodes in the input and output layers, it is also called an autoencoder. The ANN shown in Fig. 1 is also an autoencoder.

A Deep Neural Network (DNN) is an ANN with multiple hidden layers between the input and output layers[42, 43]. A DNN can be trained with the standard back-propagation algorithm[43], which is a method of calculating the gradient of the loss function (to produce the cost associated with a given state) and set the weights iteratively. While DNNs can produce superior results, they are prone to overfitting because of the multiple hidden layers, and regularization methods are applied to combat overfitting[44]. Added layers also result in more training parameters, and various methods to speed up computation have been applied, such as batching (computing the gradient on several training examples at once rather than individual examples)[45]. In addition, the increased chip processing abilities (e.g., GPUs) alleviate the computation problem, as the matrix and vector computations required are well-suited for GPUs[43].

2.1.2 Convolutional neural network

Convolutional neural networks[36] are designed for processing two-dimensional images (and other two-dimensional representations), and the convolutional layer functions as a hidden layer, in which each group of neurons (also called filters) performs a convolution operation in the image and each neuron in a filter is connected to a different region of the image but the neurons share the same weights.

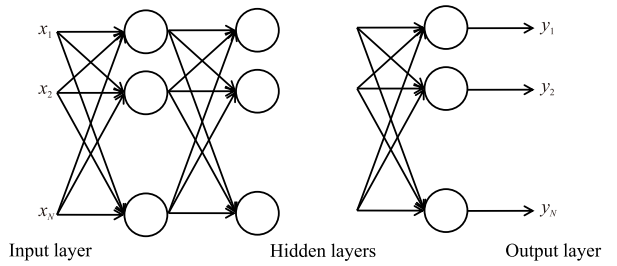

Fig. 1 An example of artificial neural network. Each circle represents a neuron.

Because the input image may have a large resolution, max-pooling[46] is often used in CNNs to reduce the original size. Max-pooling applies a max filter to non-overlapping regions of the initial image. The process of applying convolution and max-pooling operations is illustrated in Fig. 2.

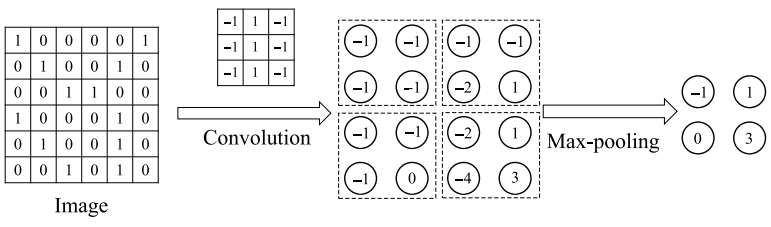

Fig. 2 Convolutional and max-pooling operations of a convolutional neural network using a 3 × 3 filter (adapted from a deep-learning tutorial[47]).

2.1.3 Recurrent neural network

RNNs[48] are neural networks with loops. They can be trained by the reverse mode of automatic differentiation[49], or the back-propagation algorithm by "unfolding" the network through time and constraining some of the connections to always hold the same weights[50].

Long Short-Term Memory (LSTM) networks[51] are RNNs that solve the vanishing gradient problem[52], when the gradients of some of the weights start to shrink or enlarge if the neural network is unfolded too many times. In an LSTM, the hidden layer is replaced by recurrent gates called forget gates (Fig. 3). Compared to RNNs, LSTM has been proved to generate superior results in many problems.

2.2 Deep learning for traffic forecasting

2.2.1 Traditional methods

Short-term traffic forecasting has been an active research area for over sixty years. Before the popularity of deep learning, traffic forecasting-related problems have been addressed with many models, which include the following:

- Time-series models, such as ARIMA[54], in which the future value of time series is a linear combination of previous values and residuals, and differentiation is used to obtain stationarity from the nonstationary time series[55]; Vector AutoRegressive (VAR), which captures the linear interdependencies among interrelated time series[54]; and seasonal ARIMA, which applies the additional seasonal difference to seasonal time series before using ARIMA[56].

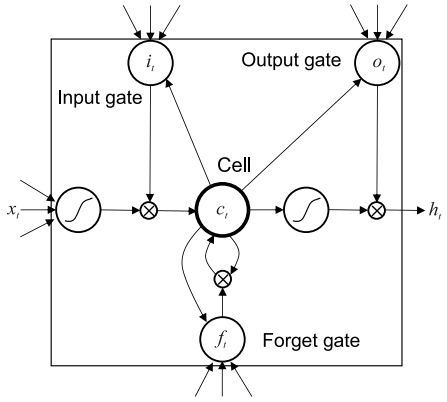

Fig. 3 Long short-term memory network (adapted from Ref. [53]).

• State-space models, such as the spatiotemporal random effect model[57], which use a set of inputs, outputs, and state variables to describe a system by a set of first-order differential equations.

• Non-parametric models, such as Bayesian network[58] and spatialtemporal weighted k-nearest neighbor[59], which are data-driven and approximate the relationship between the dependent and independent variables using a training dataset.

The traditional methods are adequately studied and described in Ref. [60].

2.2.2 Summary of previous works

With the developments of deep learning and more open geospatial datasets in recent years, deep-learning models have been trained with big data and then applied to different traffic forecasting problems. In Table 1, we list some recent works in which deep-learning

Table 1 A summary of previous works.

Problem	Dataset	Deep-learning method	Baseline method
Bus ridership forecasting	Smart card data[9]	DBN[9]	
Crowd flow forecasting	Taxicab GPS trajectory data[11], NYC bike sharing data[11]	DNN[10], CNN[11], RNN[11]	HA[11], ARIMA[10,11], SARIMA[10,11], VAR[10,11], ANN[11], DNN[11], CNN[10]
Extreme condition traffic forecasting	Traffic data collected by highway loop detectors[12], accident data[12]	LSTM[12]	ARIMA[12], RW[12], HA[12], SARIMA[12]
Human trajectories forecasting	Data usage detail records[13], multi-object tracking datasets[14]	CNN[13], STAM[14]	LSTM[14]
Supply-demand forecasting for online car-hailing service	Car-hailing service data[15], weather data[15], traffic data[15]	RNN[15]	HA[15], LASSO[15], GBDT[15], RF[15]
Taxi destination forecasting	Taxicab GPS trajectory data[16]	MLP[16], RNN[16], CNN[17]	MLP[17]
Traffic accident forecasting	GPS records[18], traffic accident data[18]	SAE[18]	DT[18], LR[18], SVM[18]
Traffic accident severity forecasting	Traffic accident data[19]	LSTM[19]	MLP[19], BLR[19]
Traffic congestion forecasting	Traffic condition data[20]	LSTM[20]	MLP[20], DT[20], SVM[20]
Traffic flow forecasting	Traffic flow data collected by freeway station's detectors[21,22,24,26,27], weather data[26], traffic flow data collected by observation stations with cameras, induction coils and velocity radars[28]	SAE[22,24], DPN[26], LSTM[23,25,27,28], GRU[25], DBN[21]	RF[22], RW[23,27], SVM[22,23,27,28], RBF[22,28], ARIMA[21,25,26,28], SAE[23,27,28]
Traffic speed forecasting	GPS traces[33], traffic message channel data[29], remote microwave sensor data[30]	CNN[32,33], RNN[32], DBN[31], LSTM[30]	OLS[33], kNN[33], ANN[33], RF[33], SAE[33], RNN[33], LSTM[33], ARIMA[30–32], SVR[30,32], SAE[32]
Travel time forecasting	Simulated traffic volume distribution[34], traffic flow data collected by station detectors[35]	SAE[34], DBN[35]	ARIMA[35]

methodologies were applied to traffic forecasting problems, and for clarity, the full terminologies for the abbreviations in Table 1 are given in Table 2. Because of space constraints, only typical methods and the most recent works are listed, as traffic forecasting is a very wide research area.

In Table 1, we categorize the previous works by the problems they address, and then list the used datasets, deep-learning methods, and baseline methods. For the deep-learning methods, we list the specific family of models being used, such as LSTM, where possible; otherwise, we list a more general concept, such as RNN. For the baseline methods, we do not only list the traditional methods used, but also some deep-learning methods, providing they were proposed by the authors and compared with the traditional methods. In some works, the performance between deep-learning methods may be compared, considering different parameters; in these cases no baseline methods are listed.

Table 2 Abbreviations used in Table 1 and their corresponding terminologies (in alphabetical order).

ANN	Artificial neural network
ARIMA	Autoregressive integrated moving average
BLR	Bayesian logistic regression
CNN	Convolutional neural network
DBN	Deep belief network
DNN	Deep neural network
DT	Decision tree
GBDT	Gradient boosting decision tree
GRU	Gated recurrent units
HA	Historical average
LASSO	Least absolute shrinkage and selection operator
LR	Logistic regression
LSTM	Long short-term memory
MLP	Multilayer perceptron
OLS	Ordinary least squares
RBF	Radial basic function
RF	Random forest
RNN	Recurrent neural network
RW	Random walk
SAE	Stacked autoencoder
SARIMA	Seasonal ARIMA
STAM	Spatial-temporal attention model
SVM	Support vector machine
SVR	Support vector regression
VAR	Vector autoregressive
kNN	k-nearest neighbor

Although many studies have covered various forecasting problems (Table 1), deep-learning methodologies can still be much applied to solve traffic forecasting problems for these three reasons. First, deep learning is still a very active field, and new models and algorithms are continuously proposed. Second, more traffic data are becoming publicly available. With the emergence of new research areas, such as autonomous driving, which involves both large amount of datasets and forecasting problems, deep-learning methods have wide potential applications. Third, deep-learning models are computationally expensive to train and demand a huge amount of training time, which makes deep learning not very flexible in real systems; thus, methods of implementing and deploying deep learning-based systems are yet to be studied.

3 Geospatial Data to Images: Our Framework

Based on insights from previous works, we propose a deep-learning framework that transforms geospatial data to images before utilizing state-of-the-art deep-learning methodologies. Our framework can be summarized as shown in Fig. 4. Our framework has two inputs: geospatial data and external factors. For the geospatial data, we perform data transformation and image preprocessing to transform geospatial data to images. For the external factors, we extract their features. Then both the images and features are fed into the deep-learning models to yield the predicted results.

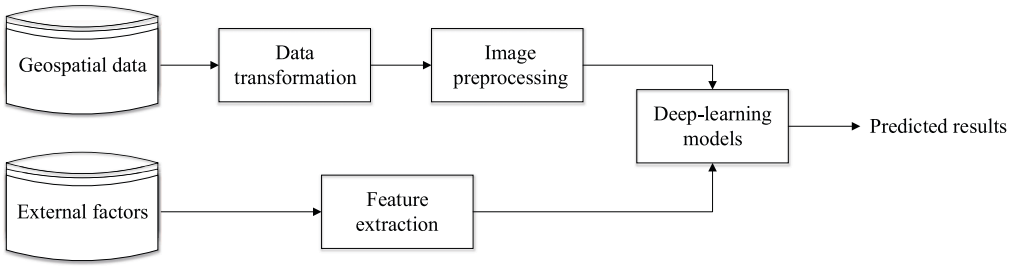

Fig. 4 Structure of our deep-learning framework.

3.1 Data transformation

In this section, we introduce two methods of transforming geospatial data to images: *individual-based data transformation* and *crowd-based data transformation*. The choice between these two data transformations depends on the specific application. For both methods of data transformation, the geospatial data need to be transformed to gray images. To effect this, we have to grid the whole map based on the location information of the geospatial data, and then each grid cell is seen as a pixel. Intuitively, a smaller gap between grids would generate an image with a higher resolution. We also have to partition the time domain into small time intervals. Because we are dealing with forecasting problems, the common scenario is to predict the situation of the next time interval based on the data collected in the previous time intervals. Then using the data within each cell, we can transform a cell into a single value by adding the data or taking the average, or other application-centric methods.

To explain the transformation process, we use an example of check-ins collected from two individuals, within a location range of $[0, 3) \times [0, 3)$ and time range of $[0, 3)$, as shown in Figs. 5 and 6.

3.1.1 Individual-based data transformation

For individual-based data transformation, the geospatial data are grouped by different individuals and transformed to an image per individual. As shown in Fig. 5, the value in each cell represents a 0/1 binary variable which indicates whether the individual has visited the cell or not. With other geospatial data, it may also be a continuous variable, such as the moving speed of the individual in the cell.

Individual-based data transformation is suitable for individual-based applications, such as human trace forecasting and taxi destination forecasting.

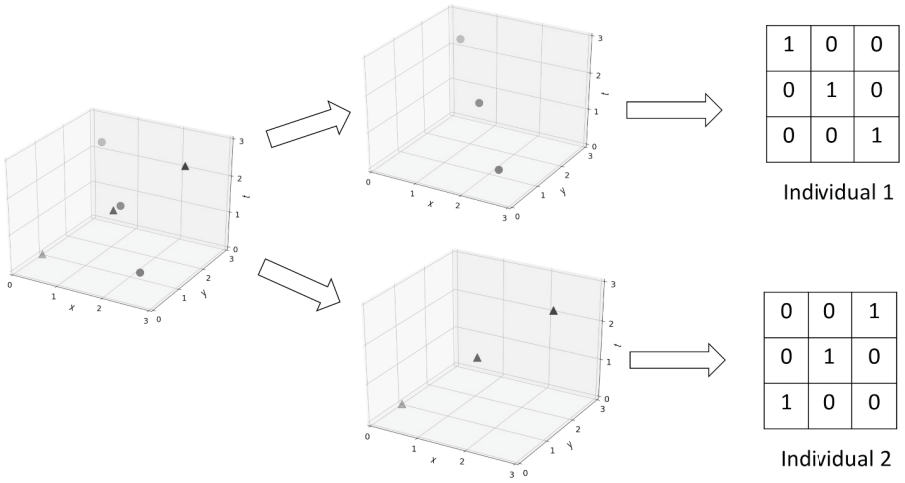

Fig. 5 Individual-based data transformation, with each image representing the movement of an individual.

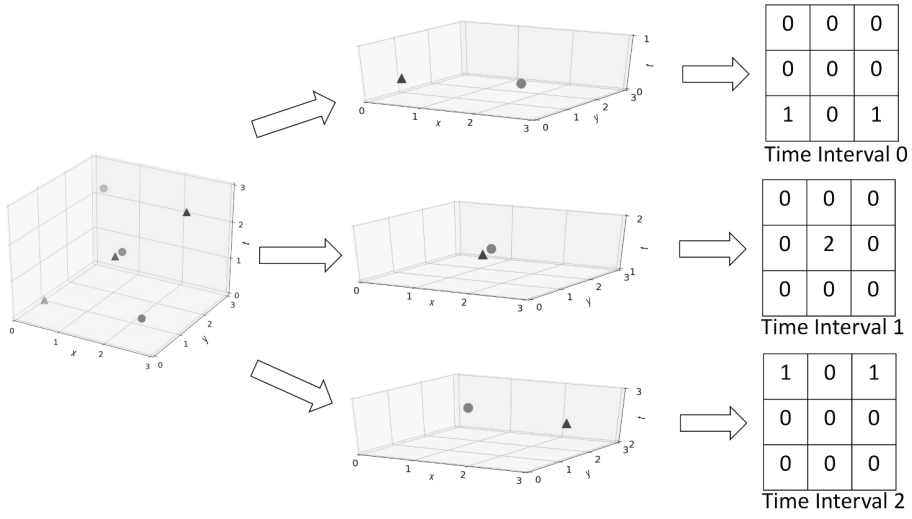

Fig. 6 Crowd-based data transformation, with each image representing the number of check-ins in a time interval.

3.1.2 Crowd-based data transformation

For crowd-based data transformation, the geospatial data are grouped by different time intervals and transformed to an image per time interval. As shown in Fig. 6, the value in each cell represents the number of check-ins within a cell in a specific time interval. With other geospatial data, it may also represent other crowd-based characteristics, such as the size of the flow, the number of individuals, and the supply and demand of transportation services.

Crowd-based data transformation is suitable for crowd-based applications, such as traffic flow forecasting and crowd flow forecasting.

3.2 Image preprocessing

Before feeding the images into the deep-learning models, some preprocessing operations may be applied, such as *image fusion* and *image selection*, to choose the appropriate images to use.

3.2.1 Image fusion

Image fusion has been used in computer vision area to combine two or more images from different sources into a single image[61]. One advantage of image fusion is that the combined image could provide more information than any of the input images. In addition, the errors in a single image can be avoided by comparing with other images, if the images overlap in spatial and temporal domains.

3.2.2 Image selection

While forecasting for crowd, a practical method is to choose a sliding lookback window and use the data within this lookback window to predict the future case. This corresponds to an image selection process where only the most recent images are used to predict the near future. Based on the temporal properties, different images may be selected, such as images in the past few hours and images in the same time interval of the previous weekday.

3.3 Feature extraction

Traffic conditions are affected by external factors other than the geospatial data, such as weather data and holiday information. It has been known that traffic situations are highly affected by the weather, and the traffic patterns in workdays and holidays are different; for example, there is an obvious daily commute pattern on workdays. We need to incorporate this effect into our predictions.

3.4 Deep-learning models

The choice of deep-learning models involves domain knowledge as well as trial and error. Although different deep-learning models or their combinations can be tried and applied for forecasting problems, CNNs and RNNs are widely used to capture the spatial and temporal dependencies among the geospatial data. The performances of the deep-learning models may depend on the specific problems. In the next section, we give a specific example of the deep-learning model used in the New York taxi pick-up/drop-off forecasting problem.

4 New York Taxi Pick-up/Drop-off Forecasting Problem

In this section, we define the taxi pick-up/drop-off forecasting problem based on a New York trip record data[3], which we use to formulate the problem and validate our framework. The taxi pick-up/drop-off in one area may be affected by the neighborhood; for example, when a taxi drops off a passenger, it may search for passengers in nearby areas. The pick-up/drop-off events of taxis are also affected by the most recent time intervals and may exhibit a periodical pattern; the pick-up/drop-off during morning/evening rush hours may be similar on consecutive workdays or the same day of the previous week.

For this problem, we supported our deep-learning model with CNNs and residual networks and compared it with time-series models, such as HA model and ARIMA, which capture temporal dependencies. Based on our experiments, our deep-learning model achieved a remarkable improvement over HA model and ARIMA.

4.1 Dataset description

In our experiments, we used the NYC taxi trip record data provided by the NYC Taxi and Limousine Commission, covering both yellow and green taxis. Unlike yellow taxis, green taxis can only pick up passengers in upper Manhattan and the outer boroughs. Each taxi trip record includes fields capturing pick-up and drop-off dates/times, pick-up and drop-off locations (longitudes and latitudes), trip distances, itemized fares, rate types, payment types, and driver-reported passenger counts. We mainly used pick-up/drop-off timestamps and locations of taxi trips from July 2015 to June 2016, which contain 138 413 407 yellow taxi trips and 18 355 786 green taxi trips.

4.2 Data transformation and problem description

We used crowd-based data transformation in our taxi pick-up/drop-off forecasting problem. To realize this, we partitioned the New York map into an $M \times N$ grid map based on the location information of the geospatial data, such as the longitude and latitude, where each square formed by grids is referred to as a cell. For the high-level feature, we aimed to predict the taxi pick-up/drop-off number within cell (i, j) in time interval k, based on the previous data.

Formally, we denote a taxi trip record r as $(t_r^O, t_r^D, x_r^O, y_r^O, x_r^D, y_r^D)$, where t_r^O and t_r^D represent the pick-up and drop-off timestamps, respectively; x_r^O and y_r^O represent the pick-up longitude and latitude, respectively; x_r^D and y_r^D represent the drop-off longitude and latitude, respectively. For each cell, the longitude and latitude ranges of cell (i, j) were represented as $[x_i, x_{i+1}) \times [y_j, y_{j+1})$. Time interval k was defined as $[t_k, t_{k+1})$. We define the taxi pick-up/drop-off numbers within cell (i, j) in time interval k as follows:

$$\begin{aligned} v_{i,j,k}^{\text{pick-up}} &= |\{r | r \in R, t_k \leqslant t_r^O < t_{k+1}, \\ &\qquad x_i \leqslant x_r^O < x_{i+1}, y_j \leqslant y_r^O < y_{j+1}\}|, \\ v_{i,j,k}^{\text{drop-off}} &= |\{r | r \in R, t_k \leqslant t_r^D < t_{k+1}, \\ &\qquad x_i \leqslant x_r^D < x_{i+1}, y_j \leqslant y_r^D < y_{j+1}\}|, \end{aligned}$$

where $|\cdot|$ represents the cardinality of a set, r represents a trip record, and R represents the set of all the trips (including yellow and green taxi trips).

We denote the taxi pick-up/drop-off numbers within time interval k as a tensor $\boldsymbol{V}_k \in \mathbf{R}^{2 \times M \times N}$, where $(\boldsymbol{V}_k)_{0,i,j} = v_{i,j,k}^{\text{pick-up}}$, $(\boldsymbol{V}_k)_{1,i,j} = v_{i,j,k}^{\text{pick-up}}$. The taxi pick-up/drop-off forecasting problem can be stated as given $\{\boldsymbol{V}_k | k = 0, 1, \ldots, n-1\}$, predict $\boldsymbol{V}_n$.

We used Root Mean Square Error (RMSE) as our evaluation metric, which can be formulated as

$$\text{RMSE} = \sqrt{\frac{1}{S} \sum_i (v_i - \hat{v}_i)^2},$$

where $\hat{v}$ and v are the predicted and true values, respectively, and S is the total number of the values to predict.

In our experiments, we partitioned the New York map into a 25×25 grid map and evenly split a day into 48 time intervals. We used the taxi trip record from July 2015 to May 2016 as the training set and the taxi trip record in June 2016 as the testing set. We evaluated the RMSE for pick-up and drop-off predictions separately.

To prepare the inputs for the deep-learning model, we conducted two more preprocessing operations:

- We used the log function to transform the scale of the original pick-up/drop-off numbers, as the supply and demand for taxis was unbalanced in different areas, and thus, the distribution of these numbers was highly skewed. Most cells had a small number of pick-ups or drop-offs, and only few cells had an extremely large number;
- Then we used a min-max normalization process to scale the numbers into the range $[-1, 1]$, as required by the tanh activation function of the deep-learning model.

For evaluation, the reversed processes were performed before the RMSEs were calculated.

4.3 Image preprocessing

Since we lacked other sources of data which also reflects the taxi pick-up and drop-off situations, image fusion technologies were not used. In the image selection process, to predict the pick-up and drop-off numbers $\boldsymbol{V}_k \in \mathbf{R}^{2 \times 25 \times 25}$ in interval k, we chose the following pick-up and drop-off numbers as inputs:

- $\boldsymbol{V}_{k-\ell}, \ldots, \boldsymbol{V}_{k-1}$, where ℓ represents the length of the lookback window;
- $\boldsymbol{V}_{k-48}$, which represents the pick-up/drop-off numbers in the same time interval a day earlier;
- $\boldsymbol{V}_{k-7\times48}$, which represents the pick-up/drop-off numbers in the same interval a week earlier;
- External factors, including weather, wind speed, and holiday data.

We compared the effect of different lengths of the lookback window (i.e., $\ell = 2, 4, 6$).

While our target was to predict the overall taxi pick-up/drop-off situation, there were two types of taxis: yellow taxis and green taxis. Although we could easily add the pick-up/drop-off numbers from these two types of taxis, it would be interesting to investigate if a better prediction result can be achieved by separately using the pick-up/drop-off images from the yellow taxis and green taxis separately. This would be considered in the future.

4.4 Feature extraction

External factors, such as weather and wind speed, may affect the demand for taxis. On a rainy or windy day, the demand for taxis may increase as those who usually use bicycles may find it inconvenient to ride in a bad weather.

To prepare the external factors as inputs for the deep-learning model, we used one-hot coding to transform the holiday data, and weather conditions into binary

vectors, where 1 indicates the day is a holiday, and 0 indicates otherwise. In addition, we had a 25-bit vector for the 25 types of weather conditions, such as rain, snow, and fog. The temperature and wind speed were scaled to the range $[-1, 1]$ with the min-max normalization. Because we lacked data on the weather condition, temperature, or wind speed of the future time interval k we aimed to predict, we only used the information of the most recent time interval $k-1$. All these vectors and scaled values were then concatenated to a long vector and used as the input for the deep-learning model.

4.5 Deep-learning model

In our deep-learning model, we combine CNNs and residual networks. Residual networks have been used to build extremely deep models and achieve a tremendous success on tasks such as image recognition[37].

In our implementation of the deep-learning model, we used Theano[62], which is a Python framework for fast computation of mathematical expressions, and Keras[63], which is a high-level neural network application programming interface written in Python. For convenience, we used the Keras layers to present our deep-learning model as shown in Fig. 7. Each layer used in the model is explained as follows:

- InputLayer: the layer that specifies the input;
- Conv2D: the two-dimensional convolutional layer, in which each filter has a size of 3×3;
- Activation: the layer that applies an activation function to an output, e.g., the ReLU function and tanh function;
- Merge: the layer that merges a list of inputs;
- Dense: the regular densely-connected neural network layer;
- Reshape: the layer that reshapes an output to a certain shape.

While the input images were independently processed at first, a merge layer was used to fuse the influence of different images as well as the external factors. In this way, the influence of different images, which represents the pick-up/drop-off situation in the past few hours, a day earlier, and a week earlier, could be learned separately and their degrees of influence on the time interval k could be quantified and adjusted by the merge layer.

Since we were predicting the taxi pick-up/drop-off numbers within the same cells with the inputs, we did not use subsampling or max-pooling in the CNN for this specific problem, but only convolutions. For the residual network, we used an Activation layer and a Conv2D layer as an RNU, as shown in Fig. 7. We studied the effects of different numbers of RNUs used in the model.

The loss function to be minimized in the training process is the mean squared error between the predicted pick-up/drop-off numbers and the true pick-up/drop-off

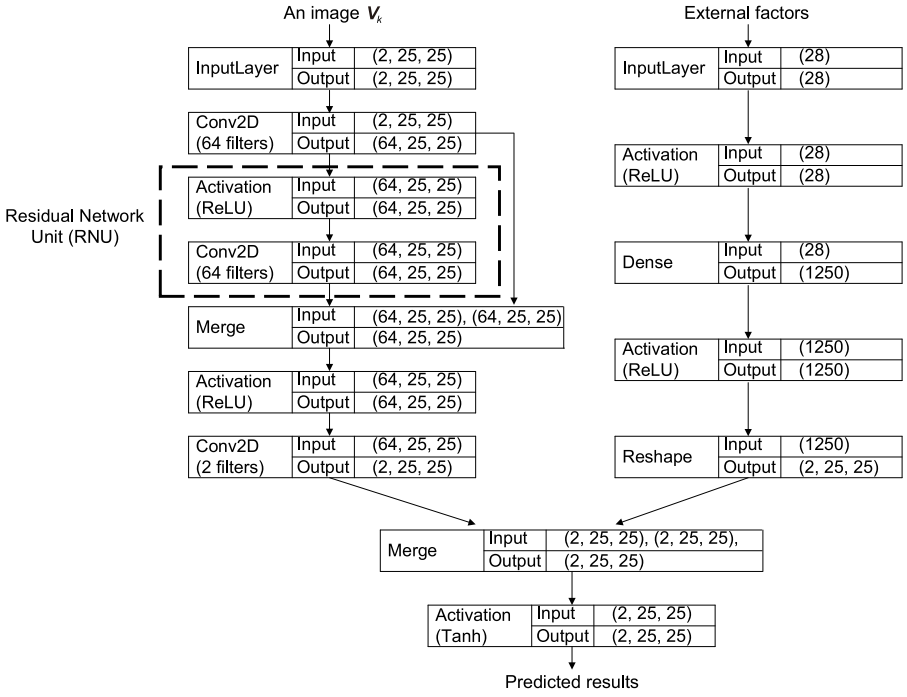

Fig. 7 Workflow of our deep-learning model. Here we illustrate the case with one image and one residual network unit.

numbers:

$$\mathcal{L}(\theta) = ||\hat{\boldsymbol{V}}_k - \boldsymbol{V}_k||_2^2 \quad (1)$$

where θ denotes all learnable parameters in our deep-learning model, $\hat{\boldsymbol{V}}_k$ denotes the predicted pick-up/drop-off numbers and $\boldsymbol{V}_k$ denotes the true pick-up/drop-off numbers, both in time interval k.

Algorithm 1 outlines our training process. To accelerate the training process, we use the mini-batch technology[64] (we use 25 as the batch size) and Adam[65] as the optimizer.

4.6 Baseline methods

We compared our deep-learning model with two baseline methods:

• Historical Average. In our implementation of the HA method, we used the weekly average values of pick-ups and drop-offs in each time interval.

• ARIMA. We used a Python module *statsmodels* for our implementation of ARIMA model and the parameters were optimized by grid searching.

4.7 Experiments and results

Our results of predicting the taxi pick-ups/drop-offs using different methods and parameters are summarized in Table 3.

Our key findings are as follows:

• For the baseline models, HA captured the weekly pattern, while ARIMA used the most recent information. Based on our results, the weekly pattern outstripped the most recent information in the taxi pick-up/drop-off forecasting problem.

• Our deep-learning framework yielded a remarkable improvement over both baseline methods.

• With different lengths of lookback window ℓ, we may achieve different performances for pick-up or drop-off forecasting. As seen from our results, using a smaller ℓ of 2, the best RMSE for pick-up forecasting was achieved, and using a bigger ℓ of 6, the best RMSE for drop-off forecasting was achieved.

• Using more RNUs does not always result in better performance. When the length of lookback window ℓ is fixed as 4, the best performance was achieved by using only one RNU.

Algorithm 1 Training of our deep-learning model

Input: Historical observations: $\{\boldsymbol{V}_0, \ldots, \boldsymbol{V}_{n-1}\}$;
external factors: $\{E_0, \ldots, E_{n-1}\}$;
length of lookback window: ℓ.
Output: a deep-learning model $\mathcal{M}$
1: initialize the training dataset $\mathcal{D} \leftarrow \varnothing$
2: **for** all available time interval k $(7 \times 48 \leqslant k \leqslant n-1)$ **do**
3: $I_k = [\boldsymbol{V}_{k-\ell}, \ldots, \boldsymbol{V}_{k-1}, \boldsymbol{V}_{k-48}, \boldsymbol{V}_{k-7\times48}]$
4: put an training instance $(\{I_k, E_{k-1}\}, \boldsymbol{V}_k)$ into $\mathcal{D}$
5: **end for**
6: initialize the parameter θ
7: **repeat**
8: randomly select a batch of training instances $\mathcal{D}_b$ from $\mathcal{D}$
9: find θ by minimizing the objective (1) with $\mathcal{D}_b$
10: **until** stopping criteria is met
11: **return** the trained model $\mathcal{M}$

5 Conclusion

In this study, we focus on the application of deep-learning technologies in a series of traffic forecasting problems based on geospatial data. In addition, we propose a preliminary framework for traffic forecasting, which transforms the geospatial data to images and leverages the state-of-the-art deep-learning technologies. Our experiments on New York taxi pick-up/drop-off forecasting problem demonstrate that our framework greatly outperforms traditional methods.

Acknowledgment

This work was funded by Shenzhen Municipal Development and Reform Commission, Shenzhen Engineering Laboratory for Data Science and Information Technology (No. SDRC [2015]1872).

Table 3 RMSE results.

Model			Pick-up RMSE	Drop-off RMSE
Historical average			10.127	8.516
ARIMA			12.210	11.209
Deep learning	Length of lookback window ℓ	Number of residual network units		
	2	1	**6.685**	7.893
	4	0	8.744	8.167
	4	1	7.914	6.552
	4	2	8.185	6.625
	4	3	8.251	7.037
	6	1	8.033	**6.542**

References

[1] Y. Zheng, L. Wang, R. Zhang, X. Xie, and W. Ma, GeoLife: Managing and understanding your past life over maps, in *Proc. 9th International Conference on Mobile Data Management*, Beijing, China, 2008, pp. 211–212.

[2] California Department of Transportation Caltrans Performance Measurement System (PeMS) database, http://pems.dot.ca.gov/, 2017.

[3] NYC Taxi & Limousine Commission TLC Trip Record Data, http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml, 2017.

[4] C. Szegedy, A. Toshev, and D. Erhan, Deep neural networks for object detection, in *Advances in Neural Information Processing Systems*, Lake Tahoe, NV, USA, 2013, pp. 2553–2561.

[5] A. Graves and J. Schmidhuber, Offline handwriting recognition with multidimensional recurrent neural networks, in *Advances in Neural Information Processing Systems*, Hyatt Regency, Canada, 2009, pp. 545–552.

[6] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, et al., Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[7] A. Krizhevsky, I. Sutskever, and G. Hinton, Imagenet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems*, Lake Tahoe, NV, USA, 2012, pp. 1097–1105.

[8] I. Sutskever, O. Vinyals, and Q. Le, Sequence to sequence learning with neural networks, in *Advances in Neural Information Processing Systems*, Montreal, Canada, 2014, pp. 3104–3112.

[9] J. Baek and K. Sohn, Deep-learning architectures to forecast bus ridership at the stop and stop-to-stop levels for dense and crowded bus networks, *Applied Artificial Intelligence*, vol. 30, no. 9, pp. 861–885, 2017.

[10] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, DNN-based prediction model for spatio-temporal data, in *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, San Francisco, CA, USA, 2016, p. 92.

[11] J. Zhang, Y. Zheng, and D. Qi, Deep spatio-temporal residual networks for citywide crowd flows prediction, in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, CA, USA, 2017, pp. 1655–1661.

[12] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu, Deep learning: A generic approach for extreme condition traffic forecasting, in *Proceedings of the 2017 SIAM International Conference on Data Mining*, Houston, TX, USA, 2017, pp. 777–785.

[13] X. Ouyang, C. Zhang, P. Zhou, and H. Jiang, Deepspace: An online deep learning framework for mobile big data to understand human mobility patterns, https://arxiv.org/abs/1610.07009, 2016.

[14] D. Varshneya and G. Srinivasaraghavan, Human trajectory prediction using spatially aware deep attention models, https://arxiv.org/abs/1705.09436, 2017.

[15] D.Wang,W. Cao, J. Li, and J. Ye, Deepsd: Supply-demand prediction for online car-hailing services using deep neural networks, in *Proceedings of IEEE 33rd International Conference on Data Engineering*, San Diego, CA, USA, 2017, pp. 243–254.

[16] A. De Brébisson, É. Simon, A. Auvolat, P. Vincent, and Y. Bengio, Artificial neural networks applied to taxi destination prediction, in *Proceedings of the 2015 International Conference on ECML PKDD Discovery Challenge*, Aachen, Germany, 2015, pp. 40–51.

[17] J. Lv, Q. Li, and X. Wang, Modeling trajectory as image: Convolutional neural networks for multi-scale taxi trajectory prediction, https://arxiv.org/pdf/1611.07635, 2016.

[18] Q. Chen, X. Song, H. Yamada, and R. Shibasaki, Learning deep representation from big and heterogeneous data for traffic accident inference, in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, Phoenix, AZ, USA, 2016, pp. 338–344.

[19] M. Sameen and B. Pradhan, Severity prediction of traffic accidents with recurrent neural networks, *Applied Sciences*, vol. 7, no. 6, p. 476, 2017.

[20] Y. Chen, Y. Lv, Z. Li, and F. Wang, Long short-term memory model for traffic congestion prediction with online open data, in *Proceedings of the IEEE 19th International Conference on Intelligent Transportation Systems*, Rio de Janeiro, Brazil, 2016, pp. 132–137.

[21] W. Huang, G. Song, H. Hong, and K. Xie, Deep architecture for traffic flow prediction: Deep belief networks with multi-task learning, *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2191–2201, 2014.

[22] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang, Traffic flow prediction with big data: A deep learning approach, *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[23] Y. Tian and L. Pan, Predicting short-term traffic flow by long short-term memory recurrent neural network, in *Proceedings of the 2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*, Chengdu, China, 2015, pp. 153–158.

[24] Y. Duan, Y. Lv, and F. Wang, Performance evaluation of the deep learning approach for traffic flow prediction at different times, in *Proceedings of the 2016 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, Beijing, China, 2016, pp. 223–227.

[25] R. Fu, Z. Zhang, and L. Li, Using LSTM and GRU neural network methods for traffic flow, prediction, in *Proceedings of the 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, Wuhan, China, 2016, pp. 324–328.

[26] A. Koesdwiady, R. Soua, and F. Karray, Improving traffic flow prediction with weather information in connected cars: A deep learning approach, *IEEE Transactions on*

Vehicular Technology, vol. 65, no. 12, pp. 9508–9517, 2016.

[27] H. Shao and B. Soong, Traffic flow prediction with long short-term memory networks (lstms), in *Proceedings of the 2016 IEEE Region 10 Conference (TENCON)*, Singapore, 2016, pp. 2986–2989.

[28] Z. Zhao, W. Chen, X. Wu, P. C. Chen, and J. Liu, LSTM network: A deep learning approach for short-term traffic forecast, *IET Intelligent Transport Systems*, vol. 11, no. 2, pp. 68–75, 2017.

[29] J. Lemieux and Y. Ma, Vehicle speed prediction using deep learning, in *Proceedings of the 2015 IEEE Vehicle Power and Propulsion Conference (VPPC)*, Montreal, Canada, 2015, pp. 1–5.

[30] 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, *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015.

[31] Y. Jia, J. Wu, and Y. Du, Traffic speed prediction using deep learning method, in *Proceedings of the IEEE 19^{th} International Conference on Intelligent Transportation Systems*, Rio de Janeiro, Brazil, 2016, pp. 1217–1222.

[32] J. Wang, Q. Gu, J. Wu, G. Liu, and Z. Xiong, Traffic speed prediction and congestion source exploration: A deep learning method, in *Proceedings of the 2016 IEEE 16^{th} International Conference on Data Mining (ICDM)*, Barcelona, Spain, 2016, pp. 499–508.

[33] 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.

[34] X. Gang, W. Kang, F. Wang, F. Zhu, Y. Lv, X. Dong, J. Riekki, and S. Pirttikangas, Continuous travel time prediction for transit signal priority based on a deep network, in *Proceedings of the 2015 IEEE 18^{th} International Conference on Intelligent Transportation Systems (ITSC)*, Las Palmas de Gran Canaria, Spain, 2015, pp. 523–528.

[35] C. Siripanpornchana, S. Panichpapiboon, and P. Chaovalit, Travel-time prediction with deep learning, in *Proceedings of the 2016 IEEE Region 10 Conference (TENCON)*, Singapore, 2016, pp. 1859–1862.

[36] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[37] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 770–778.

[38] L. Deng and D. Yu, Deep learning: Methods and applications, *Foundations and Trends in Signal Processing*, vol. 7, nos. 3&4, pp. 197–387, 2014.

[39] R. Rojas, *Neural Networks: A Systematic Introduction*. Springer Science & Business Media, 2013.

[40] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[41] L. M. Rojas-Barahona, Deep learning for sentiment analysis, *Language and Linguistics Compass*, vol. 10, no. 12, pp. 701–719, 2016.

[42] Y. Bengio, Learning deep architectures for AI, *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[43] J. Schmidhuber, Deep learning in neural networks: An overview, *Neural Networks*, vol. 61, pp. 85–117, 2015.

[44] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu, Advances in optimizing recurrent networks, in *Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, Canada, 2013, pp. 8624–8628.

[45] G. Hinton, A practical guide to training restricted boltzmann machines, *Momentum*, vol. 9, no. 1, p. 926, 2010.

[46] J. J. Weng, N. Ahuja, and T. S. Huang, Learning recognition and segmentation of 3-d objects from 2-d images, in *Proceedings of the IEEE Fourth International Conference on Computer Vision*, Berlin, Germany, 1993, pp. 121–128.

[47] H. Lee, Deep learning tutorial, http://speech.ee.ntu.edu.tw/˜tlkagk/slide/Deep\%20Learning\%20Tutorial\%20Complete\%20(v3), 2017.

[48] C. Goller and A. Kuchler, Learning task-dependent distributed representations by backpropagation through structure, in *Proceedings of the IEEE International Conference on Neural Networks*, Washington, DC, USA, 1996, pp. 347–352.

[49] A. Griewank and A. Walther, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. SIAM, 2008.

[50] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning representations by back-propagating errors, *Cognitive Modeling*, vol. 5, no. 3, p. 1, 1988.

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

[52] Y. Bengio, P. Simard, and P. Frasconi, Learning long-term dependencies with gradient descent is difficult, *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[53] A. Graves, Supervised sequence labelling, in *Supervised Sequence Labelling with Recurrent Neural Networks*, 2012, pp. 5–13.

[54] S. R. Chandra and H. Al-Deek, Predictions of freeway traffic speeds and volumes using vector, autoregressive models, *Journal of Intelligent Transportation Systems*, vol. 13, no. 2, pp. 53–72, 2009.

[55] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*. John Wiley & Sons, 2015.

[56] B. L. Smith, B. M. Williams, and R. K. Oswald, Comparison of parametric and nonparametric models for traffic flow forecasting, *Transportation Research Part C: Emerging Technologies*, vol. 10, no. 4, pp. 303–321, 2002.

[57] Y. Wu, F. Chen, C. Lu, B. Smith, and Y. Chen, Traffic flow prediction for urban network using spatio-temporal random effects model, presented at the 91st Annual Meeting of the Transportation Research Board (TRB), Washington, DC, USA, 2012.

[58] C. M. Queen and C. J. Albers, Intervention and causality: Forecasting traffic flows using a dynamic Bayesian network, *Journal of the American Statistical Association*, vol. 104, no. 486, pp. 669–681, 2009.

[59] D. Xia, B. Wang, H. Li, Y. Li, and Z. Zhang, A distributed spatial-temporal weighted model on Mapreduce for short-term traffic flow forecasting, *Neurocomputing*, vol. 179, pp. 246–263, 2016.

[60] G. Chang, S. Wang, and X. Xiao, Review of spatio-temporal models for short-term traffic forecasting, in *Proceedings of the IEEE International Conference on Intelligent Transportation Engineering (ICITE)*, Singapore, 2016, pp. 8–12.

[61] M. B. Haghighat, A. Aghagolzadeh, and H. Seyedarabi, Multi-focus image fusion for visual sensor networks in DCT domain, *Computers & Electrical Engineering*, vol. 37, no. 5, pp. 789–797, 2011.

[62] Theano Development Team, Theano: A Python framework for fast computation of mathematical expressions, https://arxiv.org/abs/1605.02688, 2016.

[63] F. Chollet, Keras, http://keras.io, 2015.

[64] S. Ioffe and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, https://arxiv.org/abs/1502.03167, 2015.

[65] D. Kingma and J. Ba, Adam: A method for stochastic optimization, https://arxiv.org/abs/1412.6980, 2014.

Lin Zhang received the BSc, MSc, and PhD degrees from Tsinghua University, China, in 1998, 2001, and 2006, respectively. He is currently an associate professor in the Department of Electronic Engineering, Tsinghua University and an associate co-director with Tsinghua-Berkeley Shenzhen Institute. He is a recipient of IEEE/ACM IPSN Best Demo Award, 2014, IEEE CASE, Best Application Paper Award, 2013, etc. He has published more than 20 journal and conference papers. His research interests include efficient protocols for sensor networks, statistical learning and data mining algorithms for sensory data processing, and information theory.

Weiwei Jiang received the BSc degree from Tsinghua University, China, in 2013. He is currently pursuing PhD degree in the same affiliation. He has published three conference papers before. His research interests include ridesharing service, incentive mechanism design, and behavior analysis in intelligent transportation system.