Taxi Demand Prediction Using Parallel Multi-Task Learning Model

Chizhan Zhang, Fenghua Zhu, *Member, IEEE*, Xiao Wang, *Member, IEEE*, Leilei Sun, *Member, IEEE*, Haina Tang, and Yisheng Lv, *Senior Member, IEEE*

***Abstract*— Accurate and real-time taxi demand prediction can help managers pre-allocate taxi resources in cities, which assists drivers quickly finding passengers and reduce passengers' waiting time. Most of the existing studies focus on mining spatial-temporal characteristics of taxi demand distributions, while lacking in modeling the correlations between taxi pick-up demand and the drop-off demand from the perspective of multi-task learning. In this article, we propose a multi-task learning model containing three parallel LSTM layers to co-predict taxi pick-up and drop-off demands, and compare the performance of single demand prediction methodology and that of two demands' co-prediction methodology. Experimental results on real-world datasets demonstrate that the pick-up demand and the drop-off demand do depend on each other, and the effectiveness of the proposed co-prediction methods.**

***Index Terms*— Taxi demand prediction, pick-up/drop-off demand, multi-task learning, LSTM, deep learning.**

I. Introduction

With the rapid development of online taxi-hailing platforms such as Didi and Uber, taxi and car-hailing services have brought great convenience to our daily lives in recent years. However, there is usually an imbalance between supply and demand in this kind of service, especially in big cities. Due to the uncertainty of taxi demand in time and space, drivers have to experience many vacant trips and passengers suffer from long waiting time [1]. Accurate and timely demand prediction is a solution to alleviate these problems via pre-allocating taxi resources to match passengers' demand in different regions of cities.

There have been many attempts for predicting short-term taxi demands from different perspectives. Autoregressive Integrated Moving Average (ARIMA), a time-series analysis method, was applied to predict taxi demand in [2]. Recently, deep learning based methods are widely developed to predict taxi demand, which exhibit superior performance and have attracted many researchers' interests [3]–[5].

Although deep learning based methods have achieved great success, they still can be improved. Most studies focus on capturing the spatial-temporal relationships of taxi demands among different regions and various time intervals, ignoring correlations between the taxi pick-up demand and the drop-off demand. Intuitively, one passenger's pick-up places and drop-off places are likely the same. Thus taxi pick-up demand and drop-off demand have internal correlations. Multi-task learning (MTL) is one of machine learning paradigms, which leverage useful information shared in multiple related tasks to help improve the performance of the model [6]. Considering taxi pick-up demand prediction and drop-off demand prediction as two tasks, we can use multi-task learning techniques to handle both of these two prediction tasks. In this article, we propose a novel multi-task learning model for predicting taxi pick-up and drop-off demands simultaneously. The main contribution of this article is listed as follows:

- We design a deep learning based multi-task learning model to simultaneously predict taxi pick-up and drop-off demands, which can utilize the shared information of the two demands.
- We develop a neural network based taxi demand classifier to indirectly embed time features in taxi demand prediction models.
- We compare the performance of the proposed methods on taxi pick-up/drop-off demand prediction with those of single demand prediction models. Experiments show the effectiveness of the proposed model.

The rest of this article is organized as follows. Section II briefly introduces the related works on taxi demand prediction. Section III describes some preliminaries and definitions.

Manuscript received February 19, 2020; revised June 17, 2020; accepted July 30, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1004803; in part by NSFC under Grant U1811463, Grant U1909204, Grant 61773381, and Grant 61876011; in part by the Guandong Grant 2019B1515120030; in part by the China Railway under Grant N2019G020; and in part by the Open Program of the Zhejiang Lab under Grant 2019KE0AB03. The Associate Editor for this article was P. Wang. *(Corresponding author: Yisheng Lv.)*

Chizhan Zhang is with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China, also with the School of Future Technology, University of Chinese Academy of Sciences, Beijing 100049, China, also with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the State Key Laboratory of Nuclear Power Safety Monitoring Technology and Equipment, China Nuclear Power Engineering Company Ltd., Shenzhen 518172, China (e-mail: zhangchizhan2018@ia.ac.cn).

Fenghua Zhu, Xiao Wang, and Yisheng Lv are with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the Qingdao Academy of Intelligent Industries, Qingdao 266109, China (e-mail: fenghua.zhu@ia.ac.cn; x.wang@ia.ac.cn; yisheng.lv@ia.ac.cn).

Leilei Sun is with the State Key Laboratory of Software Development Environment (SKLSDE), Beihang University, Beijing 100083, China, and also with the Big Data Brain Computing Laboratory (BDBC Lab), Beihang University, Beijing 100083, China (e-mail: leileisun@buaa.edu.cn).

Haina Tang is with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: hntang@ucas.ac.cn).

Digital Object Identifier 10.1109/TITS.2020.3015542

The proposed methodology is demonstrated in Section IV. Experimental details and results are provided in Section V. Finally, Section VI gives the conclusion of this article.

II. Brief Related Works

The development of sensing techniques provides great opportunities for real-time taxi demand prediction by mining the patterns in the data [7]–[11]. Zhao *et al.* [12] analyzed millions of New York City (NYC) taxi pick-up records and demonstrated the predictability of taxi demand at high spatial resolution. Earlier studies mainly applied traditional time-series analysis methods and statistical machine learning methods to predict taxi demand. Moreira-Matias *et al.* [2] aggregated GPS streaming data to forecast 30-minute taxi-passenger demand by using ARIMA and Poisson models. Wei *et al.* [13] built a hybrid Zero-Grid Ensemble SpatioTemporal (ZEST) model to predict passengers' demand for chauffeured car service with four different predictors. Tong *et al.* [14] proposed a linear regression model named LinUOTD to predict large-scale taxi demand. Faghih *et al.* [15] introduced a novel spatial-temporal autoregressive model to predict short-term uber demand in New York City.

Recently, with the great success of deep learning techniques [16]–[18], many deep learning based methods have been proposed to deal with traffic prediction tasks. Lv *et al.* [19] first applied deep learning methodology to traffic flow prediction problem by using a stacked autoencoder (SAE) model. Xu *et al.* [3] adopted an LSTM model with the Gaussian kernel to predict taxi demand in different areas of New York City. Jiang and Zhang [4] processed geospatial data as images and applied residual networks to simultaneously predict taxi pick-up and drop-off demands. Zhang *et al.* [5] took historical taxi demand data in multiple time scales as input, and proposed a deep spatial-temporal residual network for city-wide crowd flow prediction. Some researchers combined CNN and LSTM to predict short-term taxi demand, such as Ke *et al.* [20], Chu *et al.* [21] and Yao *et al.* [22]. Such works usually apply CNN and LSTM to extract spatial and temporal information, and sometimes consider external factors like weather or big events into the models. In addition, Ye *et al.* [23] proposed an auto-encoder and LSTM based spatial-temporal neural network for pick-up and drop-off demand co-prediction of taxi and city bike, which is an attempt to predict two transportation modes' demands simultaneously.

Zhou *et al.* [24] used attention based conventional LSTM to predict multi-step citywide passengers' demand. Liu *et al.* [25] designed a contextual spatial-temporal network to forecast taxi origin-destination demand. With the dramatic use of social media, many researchers start mining useful traffic information from social media texts [26], for example, Chen *et al.* [27] extracted traffic related information from Sina Weibo by using deep learning methods. Social media texts are also used to predict taxi demand, an example is that Rodrigues *et al.* [28] combined web textual data and time-series taxi record data from New York city to predict passengers' taxi demand in event areas.

III. Preliminary

In this section, we firstly introduce some definitions, then formalize the taxi demand prediction problem.

A. Definitions

1) Taxi Zone: The taxi demand prediction task aims to predict the future taxi demand of many small regions in a city. Researchers usually choose a city area and divide it into many grids, and each grid is defined as a taxi zone. An example can be found in [5]. The authors select a rectangular city area as a research object, and split it into $I \times J$ equal grids based on the longitude and latitude. A grid in the i_{th} row and j_{th} column can be denoted as $G(i, j)$, where $1 \leq i \leq I$ and $1 \leq j \leq J$. Each point in the area belongs to a specific grid, and each grid is a taxi zone. There are many ways to define the taxi zone. In this article, R_n is used to denote each taxi zone, where $n \in N$ and N is the number of all taxi zones.

2) Time Interval: The continuous time is partitioned into sequential and equal time segments. Each segment is defined as a time interval, which is denoted as: T_k, $1 \leq k \leq K$, where K is the number of time intervals during a continuous period of time. For example, if we use 30 minutes as the length of a time interval, then one month with 30 days can be divided into 48×30 segments, and K is set to be 1440.

3) Taxi Transaction Event: A taxi transaction event is defined as a four-tuple: $\mathrm{e} = (pt, dt, pl, dl)$, where pt and dt indicate the pick-up and drop-off time, respectively, pl and dl represent the start and end location of a single taxi transaction record, respectively.

4) Taxi Demand: Taxi demand consists of the pick-up demand and the drop-off demand. These two kinds of taxi demands can be denoted as:

$$D_t^{p,n} = |\{(pt, dt, pl, dl) : pt \in T_t \wedge pl \in R_n\}|, \quad (1)$$

$$D_t^{d,n} = |\{(pt, dt, pl, dl) : dt \in T_t \wedge dl \in R_n\}|, \quad (2)$$

where n indicates the index of a taxi zone, and t is the index of a time interval, $|\cdot|$ is the cardinality of a set, $D_t^{p,n}$ and $D_t^{d,n}$ denote the number of pick-up and drop-off taxis in taxi zone R_n at time interval T_t, respectively. In addition, $D_t^p \in \mathbb{R}^N$ and $D_t^d \in \mathbb{R}^N$ are used to represent taxi pick-up and drop-off demands of all taxi zones at t^{th} time interval. The two-tuple (D_t^p, D_t^d) represents the overall taxi demand in the whole selected city area at t^{th} time interval, which can be abbreviated as $D_t \in \mathbb{R}^{2 \times N}$.

B. Problem Formalization

Taxi demand prediction can be described as a time series forecasting problem. Given the historical taxi demand denoted by $D_{t-s}, D_{t-s+1}, \ldots, D_{t-1}$, the taxi demand prediction task is to forecast the taxi demand at t^{th} time interval, which can be written as:

$$\hat{D}_t = \mathcal{F}(D_{t-s}, D_{t-s+1}, \ldots, D_{t-1}), \quad (3)$$

where s denotes the number of historical time steps, and $\mathcal{F}$ is a prediction function. $\mathcal{F}$ takes the historical data as input and

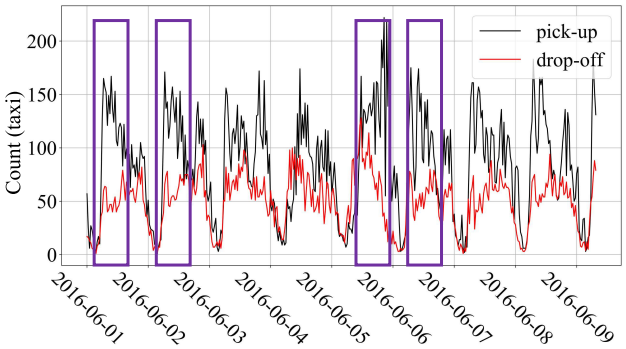


Fig. 1. Comparison of taxi pick-up demand and drop-off demand in a region of Manhattan. The two curves in purple frames have different changing patterns.

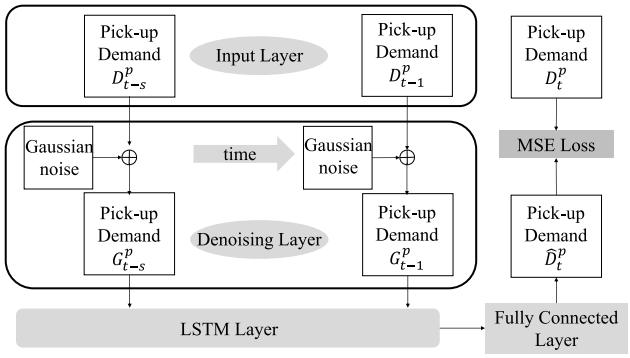


Fig. 2. Structure of single demand prediction model. Here taxi pick-up demand prediction is exhibited as an example.

outputs the predicted value. The goal is to find a function $\mathcal{F}$ that can yield a relatively accurate prediction, in other words, making $\hat{D}_t$ close to D_t.

IV. Methodology

In this section, we introduce the proposed model in detail. We firstly present the single demand prediction, and then describe the time feature extraction. Lastly, we introduce the fused co-prediction model.

A. Single Demand Prediction

A typical way for taxi demand prediction is to divide the city area into grids, and treat taxi pick-up/drop-off demand at each time interval as a two-channel image denoted by a three-dimensional matrix. Then CNN and LSTM can be applied to extract the spatial and temporal relationships of the two demands [20]–[22]. However, these two demands have inconsistent periodic characteristics when analyzing their changing pattern. As shown in Fig.1, there exist time periods that taxi pick-up demand is decreasing while taxi drop-off demand increasing. Simply fusing the data of these two demands in a shared layer may erase their own specific characteristics, thereby possibly reducing the prediction accuracy. Some researchers have attempted to catch the change modes hidden in traffic flow data [29].

Motivated by catching different changing patterns of taxi pick-up and drop-off demands, we build a single demand prediction model to forecast each one. CNN is widely used to extract spatial relationships [4], [21], [22], but it can only handle data in Euclidean space, that is to say, the taxi zones should be gridded, which means it is not suitable for irregular and discrete taxi zones. In addition, taxi demand of two distant regions may exist high correlations, while CNN mainly extracts local characteristics. Therefore, we only adopt LSTM neural network for taxi demand prediction, yet we also conduct experiments combing both CNN and LSTM for comparison. Furthermore, inspired by stacked denoising autoencoders [30] and its application in traffic data imputation [31], we embed a denoising layer following the input layer to make the model yield better generalization ability. Take taxi pick-up demand for example, Fig. 2 displays the structure of single demand prediction model, where a denoising layer is used to handle the random noise hidden in the input data, and an LSTM layer is applied to model the temporal relationship of pick-up demand among different time intervals. Similarly, the prediction model on taxi drop-off demand can be built. The model for single type taxi demand prediction is explained in detail as follows. Suppose $D^{type}_{t-s}, D^{type}_{t-s+1}, \ldots, D^{type}_{t-1}$ are the observed historical data, and D^{type}_t is the observed value at time step t, where $type$ denotes either taxi pick-up demand or drop-off demand. At first, a proportional gaussian random noise is added to the input data as a denoising operation, which can be denoted as:

$$G^{type}_{t-k} = D^{type}_{t-k} + \lambda * D^{type}_{t-k} * \epsilon, \quad k = 1, 2, \ldots, s, \tag{4}$$

where λ is a proportional factor, and ϵ is a random matrix with the same shape as D^{type}_{t-k}. Each element of ϵ subjects to standard normal distribution. This operation is only used for the training dataset, in other words, there is no need to add such a noise to the input data of validation and test datasets. The output of denoising layer is fed into the LSTM layer, which is defined as:

$$L^{type}_t = LSTM(G^{type}_{t-s}, G^{type}_{t-s+1}, \ldots, G^{type}_{t-1}), \tag{5}$$

where LSTM denotes long short-term memory neural network operation, and L^{type}_t is the last step output of LSTM layer. A fully connected layer following the LSTM layer takes L^{type}_t as input and output the single demand prediction value:

$$\hat{D}^{type}_t = FC(L^{type}_t), \tag{6}$$

where FC denotes the operation of fully connected layers, and $\hat{D}^{type}_t$ is the output of the model. The loss is defined as the mean square error between D^{type}_t and $\hat{D}^{type}_t$, formulated as:

$$Loss = MSE(\hat{D}^{type}_t, D^{type}_t), \tag{7}$$

where MSE is the mean square error and D^{type}_t is the true value. The model is trained by error back propagation algorithm [32], which is a widely used methodology in training artificial neural networks.

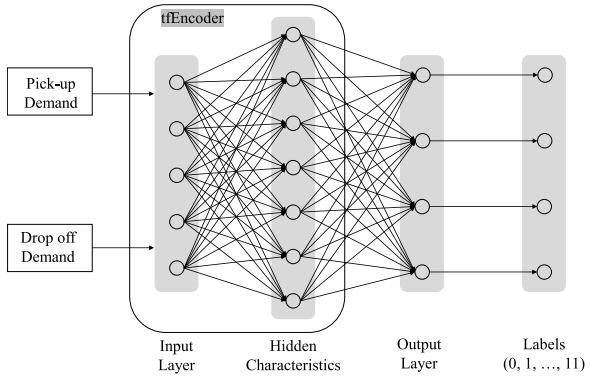


Fig. 3. Structure of taxi demand classifier. The first several layers can be regarded as a time feature encoder.

B. Time Feature Extraction

Fig. 1 shows that taxi demand is similarly periodical on a daily scale. Due to the changing trend of taxi demand varies at different time periods of a day, time of day information is incorporated into taxi demand prediction models [22], [23], and one-hot vectors are used to denote the time information. Since taxi demand and the time of a day have an inherent connection, we suppose time information is hidden in the structure of the input data. It might be an alternative way to extract temporal characteristics from the mapping between demand data and time order in a day. Thus, we build a simple taxi demand classifier by using artificial neural networks to capture the correlations between them.

The relationship between demand data and time can be modeled as a classification problem, that is, to classify the taxi demand in chronological order of a day. Firstly, 24 hours one day are partitioned into S sequential time segments; for example, we set S to 12 in this article and each segment denotes two hours. Then each taxi demand D_t is labeled according to the segment order it belongs to. For example, a taxi demand during $0:00am-2:00am$ is labeled as 0, and that during $2:00am-4:00am$ is labeled as 1, etc. The taxi demand D_t and the label $0 \sim 11$ are treated as input and output, respectively; therefore, it becomes a multi-class classification problem.

The structure of the taxi demand classifier is illustrated in Fig. 3, where multiple fully connected layers are used to build the classification model. There is a ReLU activation function following each fully connected layer except the last one, which is followed by a softmax function. After training the model, we gain a time feature encoder (tfEncoder) of taxi demand at each time interval by removing several last layers. The tfEncoder extracts hidden time characteristics of the taxi demand data, which can be seen as a representation of time information and embedded in taxi demand prediction model.

C. Fused Co-Prediction Model

It is rational to assume that taxi pick-up demand and drop-off demand are influenced by each other. We want to catch the correlations between these two kinds of demands; meanwhile, we hope to preserve their own special characteristics. A parallel multi-task learning model based on three LSTM layers (pmlLSTM) is proposed to address the above issues.

As Fig. 4 displays, the model architecture includes three parts. The first part is two single demand prediction modules, which lie at the top and bottom of the figure, respectively. They can capture the independent trend of pick-up and drop-off demand, where each single prediction module just takes one kind of demand as input and output. The second part is time feature embedding module, which applies tfEncoder from the pre-trained taxi demand classifier to extract time features of taxi demand. The third part is the fused co-prediction module, fusing all input data and time features to predict the two demands simultaneously. Other shared features of taxi pick-up and drop-off demands like weather or big event could also be added in this module.

Suppose using historical demand $D_{t-s}, D_{t-s+1}, \ldots, D_{t-1}$ to predict demand D_t. At first, taxi pick-up demand and drop-off demand are separately processed by denoising opeartion:

$$G^p_{t-k} = D^p_{t-k} + D^p_{t-k} * \lambda * \epsilon, \quad k = 1, 2, \ldots, s, \tag{8}$$

$$G^d_{t-k} = D^d_{t-k} + D^d_{t-k} * \lambda * \epsilon, \quad k = 1, 2, \ldots, s. \tag{9}$$

The time features of input taxi demand are extracted by the time encoder:

$$E_{t-k} = Encode(D_{t-k}), \quad k = 1, 2, \ldots, s. \tag{10}$$

Then two parallel LSTM layers are respectively applied to predict pick-up demand and drop-off demand, and a fused LSTM layer is introduced to simultaneously predict these two demands. The output of denoising layers and that of time encoder are merged as input of fused LSTM layer. The aforementioned operation is denoted as:

$$\hat{D}^{sp}_t = FC^p(LSTM^p(G^p_{t-s}, G^p_{t-s+1}, \ldots, G^p_{t-1})), \tag{11}$$

$$\hat{D}^{sd}_t = FC^d(LSTM^d(G^d_{t-s}, G^d_{t-s+1}, \ldots, G^d_{t-1})), \tag{12}$$

$$C_{t-k} = Concat(G^p_{t-k}, G^d_{t-k}, E_{t-k}), \quad k = 1, 2, \ldots, s, \tag{13}$$

$$\hat{D}^f_t = FC^f(LSTM^f(C_{t-s}, C_{t-s+1}, \ldots, C_{t-1})), \tag{14}$$

where p is for the pick-up and d is for the drop-off; Concat is concatenating operation; $\hat{D}^{sp}_t, \hat{D}^{sd}_t, \hat{D}^f_t$ are the outputs of single pick-up demand prediction module, single drop-off demand prediction module and fused two demands co-prediction module, respectively. The output of co-prediction module, $\hat{D}^f_t$, is regarded as the model's final prediction.

The loss of the model also involves three parts. The first part is the mean square error between the output of single prediction module and the true value, enabling the model to focus on each specific demand. The second part is the mean square error between the co-prediction result and the true value, using the information of those two demands. The third part is the mean square error between the output of single prediction module and that of the co-prediction module, in order to combine the independent features and shared characteristics of these two demands to make a better prediction.

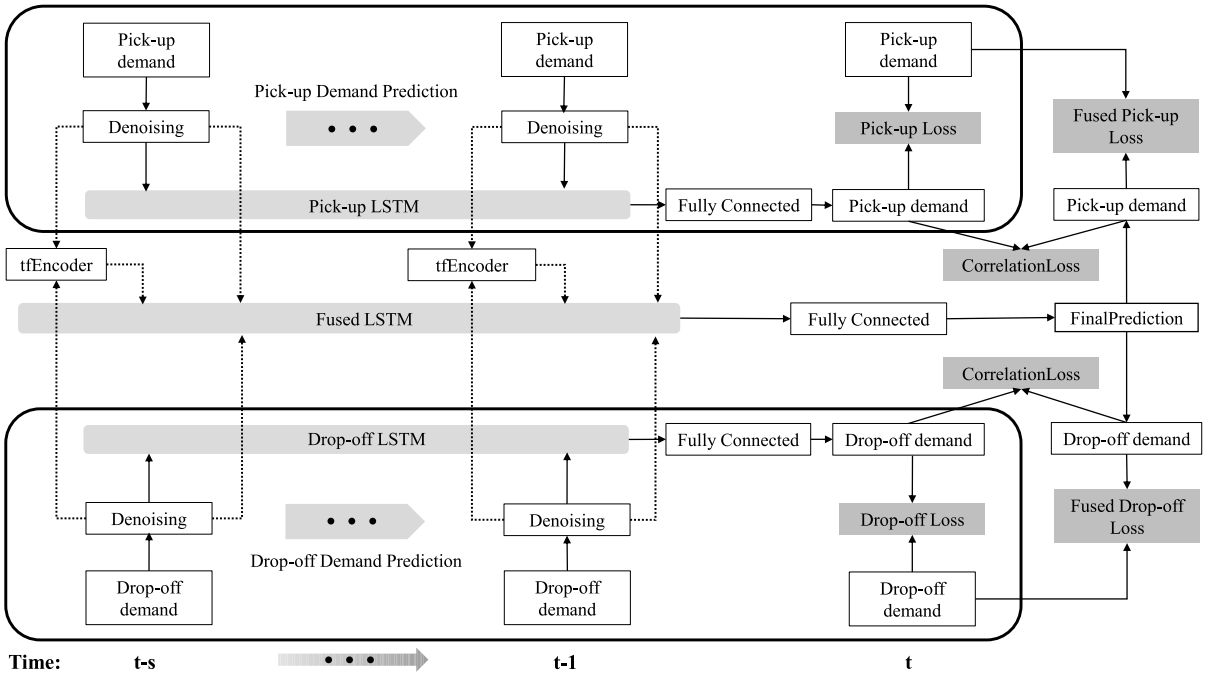


Fig. 4. Architecture of pmlLSTM model. Taxi pick-up demand prediction and taxi drop-off demand prediction are regarded as two tasks.

Algorithm 1 Preparing and Training pmlLSTM Model

Require: Taxi demand at each time interval $D_t \in \mathbb{R}^{N\times 2}$, the hyperparameters of taxi demand classifier and pmlLSTM model

Step 1: Training taxi demand classifier

1: Process taxi demand data into labeled classification samples according to the time period order in a day
2: Initialize the parameters of taxi demand classifier
3: **while** non-convergent **do**
4: Forward propagation to compute the outputs
5: Calculate the cross-entropy loss
6: Update the parameters through error back propagation
7: **return** Pre-trained taxi demand classifier network

Step 2: Training pmlLSTM model

1: Process taxi demand data into labeled regression samples according to the length of history steps
2: Use the first several layers of pre-trained taxi demand classifier as tfEncoder to build pmlLSTM network
3: Initialize the parameters of pmlLSTM
4: **while** non-convergent **do**
5: Forward propagation to compute the outputs of 3 parallel layers
6: Calculate the total loss which involves 3 parts
7: Update the parameters through error back propagation
8: **return** Well-trained pmlLSTM network

The loss is formulated as:

$$\hat{D}_t^s = (\hat{D}_t^{sp}, \hat{D}_t^{sd}), \tag{15}$$

$$Loss = MSE(\hat{D}_t^s, D_t) + MSE(\hat{D}_t^f, D_t) + MSE(\hat{D}_t^s, \hat{D}_t^f). \tag{16}$$

As shown in Algorithm 1, there are two steps to prepare and train the pmlLSTM model. Taxi demand classifier is firstly trained by cross-entropy loss and error back propagation algorithm. Then the first several layers in taxi demand classifier are used as tfEncoder to build pmlLSTM network. The single demand prediction values and the fused two demands co-prediction values are simultaneously obtained from three parallel layers by forward propagation following Equation (8)-(14), and the total loss is calculated according to Equation (15)-(16). The model is trained through error back propagation algorithm until convergence. In practice, we early-stop the training procedure according to the forecasting accuracy of the model on the validation dataset.

The single demand prediction module in pmlLSTM model can catch single pick-up or drop-off demand characteristics by minimizing the pick-up loss and drop-off loss, and the fused co-prediction module can capture the correlations between the two demands by minimizing the fused pick-up/drop-off loss. By minimizing the correlation loss between single demand prediction value and fused co-prediction value, pmlLSTM model can learn not only single demand's features, but also the shared features hidden in the two demands. The time of day information is indirectly embedded into the forecasting model by tfEncoder, thus it can learn the tendency of taxi demand in different time periods of a day. In addition, weather or other factors can also be embedded using the same way. By this design, the model can reduce generalization error and yield better prediction accuracy.

V. Experiments

In this section, we present the dataset firstly. Then we introduce baselines and give the evaluation metrics of the task. Lastly, we report the experimental results.

A. Dataset

Experiments are conducted on two real-world datasets to validate the effectiveness of the proposed method. Both of the

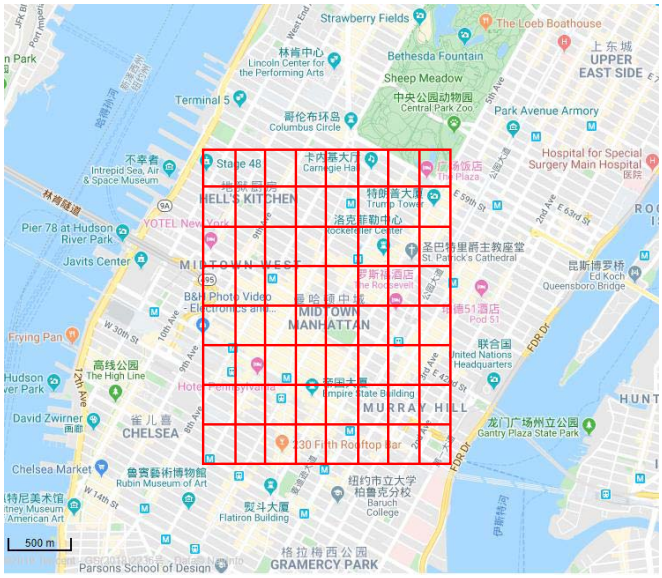


Fig. 5. Schematic diagram of dataset 1 research area. The red frame denotes our selected region.

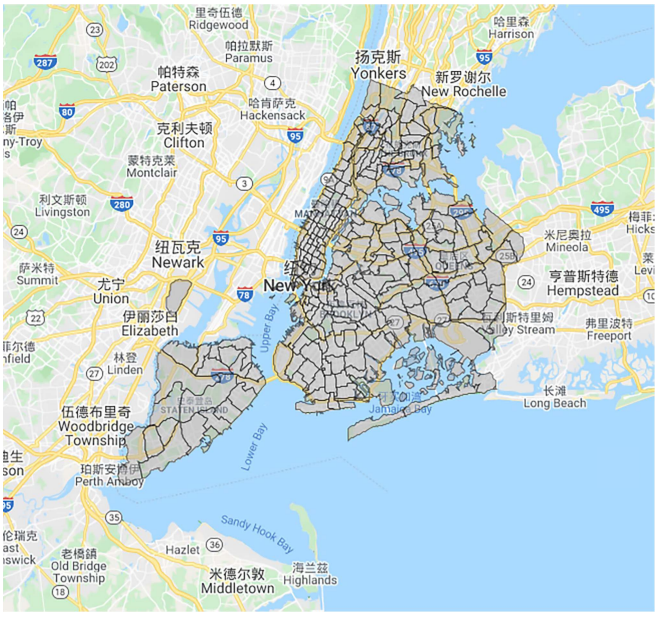


Fig. 6. New York City taxi zones division of dataset 2.

two datasets are constructed by NYC (New York City) Taxi and Limousine Commission (TLC) and are available on NYC OpenData platform.[1]

1) Dataset 1: This dataset is 2016 NYC yellow taxi trip data, which has an average of 380 thousand NYC yellow taxi trip records every day. Each record corresponds to a taxi transaction event including the following information: pick-up time, pick-up latitude, pick-up longitude, drop-off time, drop-off latitude, drop-off longitude, trip distance, passenger count, etc. The data in the first six months is used. We pre-process the original data with a series of procedures. Firstly, we choose a relatively large area containing approximately 93% of the records in the dataset. Secondly, we divide the area into 20×20 grids and put taxi trip records into each grid respectively. Then we calculate the total number of records in each grid and analyze the data distribution. We find that most of the records are concentrated near Manhattan Midtown. Therefore, as Fig. 5 shows, we finally select an area near Manhattan Midtown as a case study. It is divided into equal grids according to longitude and latitude, and each grid is a taxi zone.

2) Dataset 2: This dataset is a combination of 2018 NYC yellow-taxi/green-taxi/for-hire-vehicle trip data, which has an average of 1.2 million taxi trip records every day, and the data in the first 10 months is used. It is almost the same as dataset 1, but the longitude and latitude in a taxi transaction event are replaced with NYC taxi zone numbers. As Fig. 6 shows, NYC government divides the whole city into 263 irregular zones.[2] Instead of forecasting the taxi demand of all zones, we select a subset of taxi zones based on the Average Origin-Destination Number (AODN), which is the average amount of trip records between every origin-destination pair per time interval. Let ζ denotes an AODN threshold, then a set of origin-destination pairs is determined by checking whether the AODN is bigger than ζ. Finally, all of the taxi zones are extracted from the origin-destination pairs set. We only forecast the taxi demand of these regions simultaneously, for passengers tend to travel among them by taxis, indicating they are highly related.

[1]https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
[2]https://data.cityofnewyork.us/Transportation/NYC-Taxi-Zones/d3c5-ddgc

B. Baselines

We compare the proposed model with the following baseline methods. We carefully tune the parameters of each model and run ten times of each method in order to get reliable results.

- **Historical Average (HA)**: HA refers to predicting the future demand of a region by averaging the demand of that region at the same time interval in previous several days. For example, predicting the taxi demand of region $G(i, j)$ during $10{:}00am$-$10{:}30am$ on Monday can be done by averaging the demand of this region at the same time of each day in last week.
- **Autoregressive Integrated Moving Average (ARIMA)**: ARIMA is a widely used and well-known method for analyzing time-series data.
- **Multiple Layer Perceptron (MLP)**: A feedforward neural network containing multiple fully connected layers is used to predict taxi pick-up/drop-off demand.
- **Single Demand LSTM (sLSTM)**: An LSTM based single demand prediction model is used to forecast taxi pick-up demand and drop-off demand, respectively.
- **Single Demand CNN-LSTM (sCLSTM)**: A CNN layer is added before LSTM layer in sLSTM model. This baseline is only used for dataset 1.
- **Multiple Demands LSTM (mLSTM)**: We change the input and output of sLSTM from one kind of taxi demand to both taxi pick-up and drop-off demands; therefore, the sLSTM is transformed into a multiple demands co-prediction model.

- **Multiple Demands CNN-LSTM (mCLSTM)**: A CNN layer is added before LSTM layer in mLSTM model. This baseline is only used for dataset 1.
- **Parallel Multi-task Learning CNN-LSTM (pml-CLSTM)**: A CNN layer is added before each LSTM layer in pmlLSTM model. This baseline is only used for dataset 1.

C. Experimental Settings

We choose thirty minutes as a time interval for both of the two datasets. The following hyperparameters are selected for they enable the model to yield the best performance on validation dataset. All taxi demand data is normalized to 0 - 1 by Min-Max normalization, and the time window size of LSTM is set to 8. The total number of training epochs is 200, and the batch size is set to 100. For LSTM based models, the proportional factor λ in denoising layer is set to 0.15. The optimizer is Adam [33], and the learning rate is set to 0.001 for all of the deep learning based models. To increase the number of training samples, we use a time shift skill following [23]. For example, if we have 4 minutes shift, the time intervals would become 0:04am to 0:34am, 0:34am to 1:04am, etc. By shifting the time with 0, 4, 8, 12, 16, 20, 24, 28 minutes, we get a much larger dataset, which helps improve the performance of deep learning models. The well-trained network with the best performance on validation dataset is used to calculate the MAE and RMSE on test dataset as an evaluation of its performance. Deep learning models are implemented by pytorch framework. All experiments are conducted on a computer with an Intel Core i7-9700 CPU and an Nvidia GeForce GTX 1660ti Graphics Card. More detailed settings are listed as follows.

1) Dataset 1 Setting: The selected city area is a rectangle region, 40°44'34.8"N to 40° 46'1.2"N and 73°59'45.6"W to 73°58'19.2"W. It is split into 8×8 grids, i.e., 64 taxi zones. The data of the first four months is used to train the model, and that of the fifth and sixth months is used to validate and test the model, respectively.

2) Dataset 2 Setting: The AODN threshold is set to 10, and 86 taxi zones are extracted. The data of the first six months is used to train the model, and that of the last four months is used to validate and test the model respectively.

3) HA Setting: The length of history days is set to 7 for both of the two datasets.

4) ARIMA Setting: There are two ARIMA models trained for each taxi zone, one for pick-up demand, and the other for drop-off demand. The p, q, d tuple of ARIMA model ranges from 1 to 4 for each parameter, and the order with the best performance on training dataset is used to test.

5) MLP Setting: The input dimension is 8×128 for dataset 1 and 8×172 for dataset 2. The structure of the MLP model is the same for both of the two datasets, which has three hidden layers, and the amounts of neurons in each layer are 200, 100, 50. The output dimension is 128 for dataset 1 and 172 for dataset 2.

6) sLSTM Setting: The input dimension is 8×64 for dataset 1 and 8×86 for dataset 2. The dimension of LSTM layer is set to 200. The fully connected layer following the LSTM layer has an input dimension of 200, and an output dimension of 64 for dataset 1, 86 for dataset 2, respectively.

7) sCLSTM Setting: This model is only used for dataset 1. The input dimension is 8×64. In CNN layer, the kernel size of the filter is 3×3, and the numbers of filters are set to 16, with a stride of 1. The activation function is ReLU. In pooling layer, the kernel size of the filter is 2×2. The dimension of LSTM layer is set to 200, followed by a fully connected layer, with an input dimension of 200 and an output dimension of 128.

8) mLSTM Setting: The input dimension is 8×128 for dataset 1 and 8×172 for dataset 2. The dimension of LSTM layer is 200. The fully connected layer following the LSTM layer has an input dimension of 200, and an output dimension of 128 for dataset 1, 172 for dataset 2.

9) mCLSTM Setting: This model is only used for dataset 1. The input dimension is 8×128. The dimension of LSTM layer is set to 200, and the CNN setting is the same as that of sCLSTM. The fully connected layer following the LSTM layer has an input dimension of 200, and an output dimension of 128.

10) pmlLSTM Setting: The input dimension is 8×128 for dataset 1 and 8×172 for dataset 2. The dimension of LSTM layer in single demand prediction module is set to 50, and that of the fused LSTM layer is set to 100, for both of the two datasets. The last step output of LSTM is fed into a fully connected layer to match the output dimension. For the taxi demand classifier, we partition 24 hours into 12 sequential time segments, and each segment denotes two hours. In dataset 1, we use four fully connected layers and the number of neurons for each layer is set to 64, 64, 32, 12, respectively. In dataset 2, we use three fully connected layers and the number of neurons for each layer is set to 100, 50, 12, respectively. The first two layers in taxi demand classifier are used as tfEncoder.

11) pmlCLSTM Setting: This baseline almost has the same setting as pmlLSTM. The only difference is that there is a CNN before each LSTM layer, and the CNN setting is the same as that of mCLSTM.

D. Evaluation Metrics

We use Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) to measure the performance of our model as in [20], which can be defined as follows:

$$MAE = \frac{1}{M}\sum_{i=1}^{M}|\hat{X}_i - X_i| \tag{17}$$

$$RMSE = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(\hat{X}_i - X_i)^2} \tag{18}$$

where $\hat{X}_i$ and X_i are the prediction value and true value, respectively, and M is the number of prediction values.

E. Results and Discussion

Experimental results are displayed in Table I, II, III, Fig. 7 and Fig. 8. Each method is run 10 times and the average values are calculated as the final results. Deep learning based methods yield lower MAE and RMSE, exhibiting better performance than ARIMA and HA methods.

TABLE I

PERFORMANCE COMPARISON OF TAXI PICK-UP/DROP-OFF DEMAND PREDICTION AMONG DIFFERENT METHODS ON DATASET 1

Methods	MAE		RMSE	
	pick-up	drop-off	pick-up	drop-off
HA	8.870	8.378	13.805	13.236
ARIMA	7.960	7.199	12.150	10.560
MLP	6.555	5.807	9.998	8.590
sCLSTM	6.532	5.773	9.911	8.457
mCLSTM	6.543	5.797	9.931	8.511
pmlCLSTM	6.299	5.623	9.575	8.229
sLSTM	6.359	5.630	9.671	8.248
mLSTM	6.448	5.676	9.735	8.260
pmlLSTM	**6.202**	**5.533**	**9.399**	**8.130**

TABLE II

PERFORMANCE COMPARISON OF TAXI PICK-UP/DROP-OFF DEMAND PREDICTION AMONG DIFFERENT METHODS ON DATASET 2

Methods	MAE		RMSE	
	pick-up	drop-off	pick-up	drop-off
HA	42.019	38.550	69.324	62.146
ARIMA	21.423	20.154	33.592	30.245
MLP	17.852	16.147	29.013	25.239
sLSTM	16.529	14.941	26.813	22.984
mLSTM	16.515	14.877	26.776	22.765
pmlLSTM	**16.176**	**14.499**	**26.311**	**22.411**

TABLE III

PERFORMANCE COMPARISON OF WITH/WITHOUT ADDING TFENCODER TO PMLLSTM MODEL

Method	MAE		RMSE	
	pick-up	drop-off	pick-up	drop-off
without tfEncoder (dataset 1)	6.256	5.560	9.463	8.143
with tfEncoder (dataset 1)	**6.202**	**5.533**	**9.399**	**8.130**
without tfEncoder (dataset 2)	16.304	14.652	26.577	22.597
with tfEncoder (dataset 2)	**16.176**	**14.499**	**26.311**	**22.411**

As Table I and Table II show, LSTM is better than MLP for taxi pick-up/drop-off demand prediction, which benefits from its ability to capture long-term dependencies. However, when CNN is added to the model, the performance becomes worse in dataset 1. The reason may be that taxi demand in a region is influenced not only by its adjacent regions, but also by the longer-distance regions. CNN may not be suitable to deal with such a situation.

Table I shows that mLSTM is worse than sLSTM on dataset 1; however, Table II exhibits that mLSTM is slightly better than sLSTM on dataset 2. One possible reason is that we forecast the taxi demand of all regions in a certain area on dataset 1, while only highly related regions on dataset 2. If the regions have few correlations, it is hard to extract their spatial characteristics by forecasting their taxi demands simultaneously. Simply fusing the input data of these two

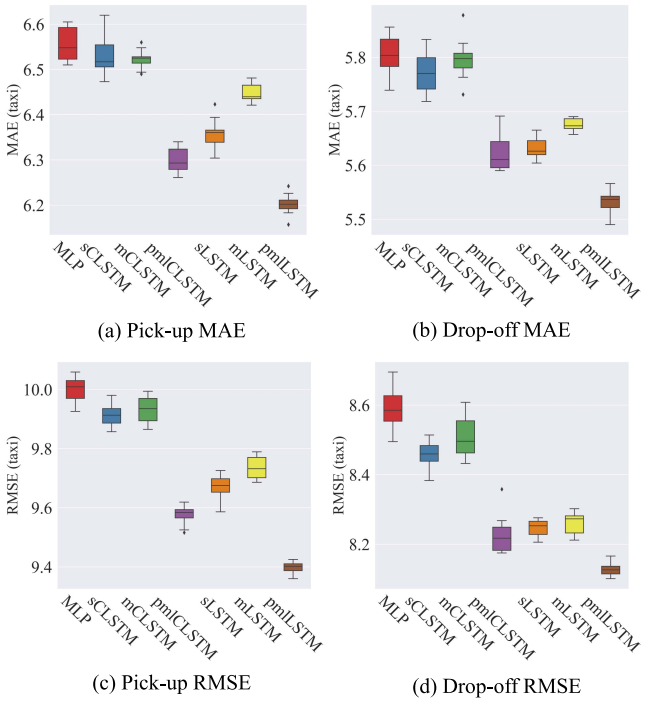


Fig. 7. Box-plot of 10 experimental results on dataset 1.

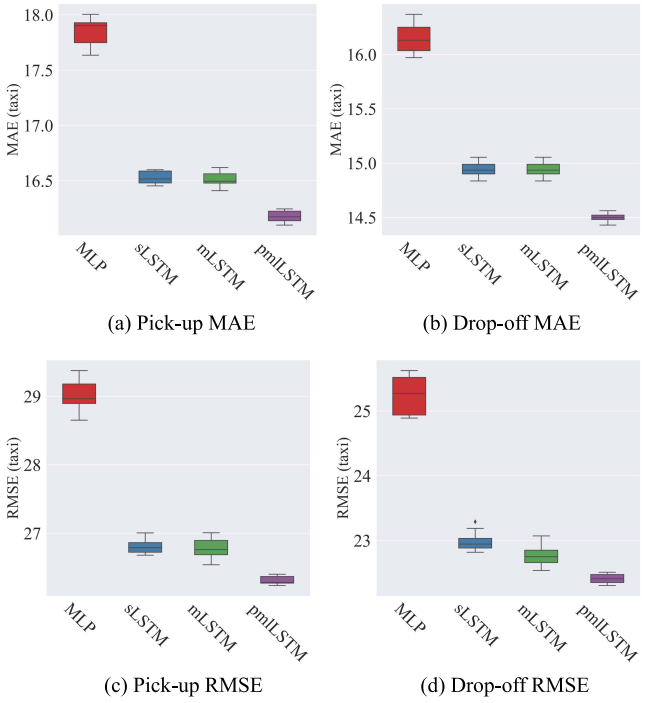


Fig. 8. Box-plot of 10 experimental results on dataset 2.

demands may not fully utilize their shared information, thus it may not yield better forecasting accuracy compared with single demand prediction methodology.

As shown in Table I and Table II, our model yields lower MAE and RMSE than both sLSTM and mLSTM on two different datasets, demonstrating that it can make better use of the correlations between taxi pick-up demand and drop-off demand. Fig. 7 and Fig. 8 exhibit the box-plot of

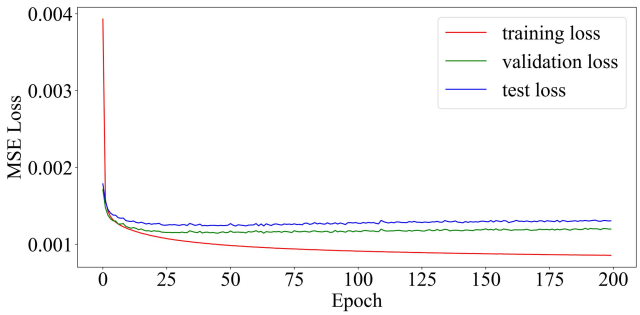
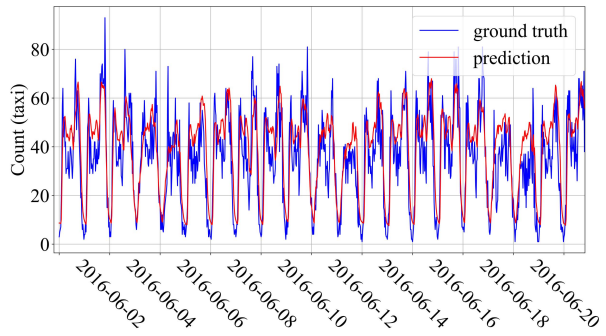
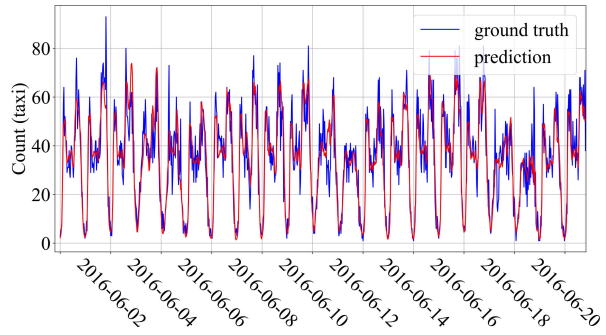


Fig. 9. Training loss of pmlLSTM model on dataset 1. Original data has been normalized to 0 - 1. Red, green and blue curves denote training loss, validation loss and test loss, respectively.



(a) Grid(1, 0) Pick-up Demand Prediction After Training 1 Epoch



(b) Grid(1, 0) Pick-up Demand Prediction After Training 200 Epochs

Fig. 10. Examples of pmlLSTM testing performance on dataset 1 in different training epochs.

10 experimental results for deep learning methods, where the results are consistent for each model. Fig. 9 displays the change of MSE loss when training pmlLSTM model on dataset 1. Validation loss and testing loss are consistent, and the model has converged without overfitting. The taxi demand prediction performances of pmlLSTM model without tfEncoder and with tfEncoder are illustrated in Table III. After tfEncoder is added to the model, the MAE and RMSE losses of taxi demand prediction decrease, which demonstrates the effectiveness of our time feature extraction model.

VI. Conclusion

In this article, we focus on the co-prediction of taxi pick-up and drop-off demands, and propose a parallel multi-task learning model, which can deal with shared features of multiple tasks simultaneously. In addition, we design a novel taxi demand classifier to extract the time information hidden in the data, which embeds the time of a day feature into the forecasting model. The proposed methods are evaluated on NYC taxi datasets. We performed extensive experiments, which show the effectiveness of the proposed co-prediction methods. Experimental results also demonstrate that taxi pick-up and drop-off demands have natural corrections, but only by carefully designing deep learning models can we best utilize them to improve the forecasting performance. For future work, it would be interesting to investigate the effectiveness of adding weather and events information into the taxi demand prediction model. It is also interesting to apply multi-task learning techniques to other traffic co-prediction problems.

Acknowledgment

The authors would like to thank the anonymous referees for their invaluable insights.

References

[1] L. Zhang *et al.*, "A taxi order dispatch model based on combinatorial optimization," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 2151–2159.

[2] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1393–1402, Sep. 2013.

[3] J. Xu, R. Rahmatizadeh, L. Boloni, and D. Turgut, "Real-time prediction of taxi demand using recurrent neural networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2572–2581, Aug. 2018.

[4] W. Jiang and L. Zhang, "Geospatial data to images: A deep-learning framework for traffic forecasting," *Tsinghua Sci. Technol.*, vol. 24, no. 1, pp. 52–64, Feb. 2019.

[5] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. 31st AAAI Conf. Artif. Intell.*, Feb. 2017, pp. 1655–1661.

[6] Y. Zhang and Q. Yang, "An overview of multi-task learning," *Nat. Sci. Rev.*, vol. 5, no. 1, pp. 34–47, Jan. 2018.

[7] F.-Y. Wang, "Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 630–638, Sep. 2010.

[8] Z. Huang *et al.*, "Modeling real-time human mobility based on mobile phone and transportation data fusion," *Transp. Res. C, Emerg. Technol.*, vol. 96, pp. 251–269, Nov. 2018.

[9] T. Liu, B. Tian, Y. Ai, and F.-Y. Wang, "Parallel reinforcement learning-based energy efficiency improvement for a cyber-physical system," *IEEE/CAA J. Automatica Sinica*, vol. 7, no. 2, pp. 617–626, Mar. 2020.

[10] A. Almalaq, J. Hao, J. Jason Zhang, and F.-Y. Wang, "Parallel building: A complex system approach for smart building energy management," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 6, pp. 1452–1461, Nov. 2019.

[11] L. Chen, X. Hu, W. Tian, H. Wang, D. Cao, and F.-Y. Wang, "Parallel planning: A new motion planning framework for autonomous driving," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 1, pp. 236–246, Jan. 2019.

[12] K. Zhao, D. Khryashchev, J. Freire, C. Silva, and H. Vo, "Predicting taxi demand at high spatial resolution: Approaching the limit of predictability," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2016, pp. 833–842.

[13] H. Wei, Y. Wang, T. Wo, Y. Liu, and J. Xu, "Zest: A hybrid model on predicting passenger demand for chauffeured car service," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2016, pp. 2203–2208.

[14] Y. Tong *et al.*, "The simpler the better: A unified approach to predicting original taxi demands based on large-scale online platforms," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 1653–1662.

[15] S. S. Faghih, A. Safikhani, B. Moghimi, and C. Kamga, "Predicting short-term uber demand in new york city using spatiotemporal modeling," *J. Comput. Civil Eng.*, vol. 33, no. 3, May 2019, Art. no. 05019002.

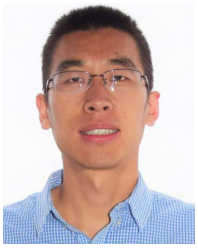
[16] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.
[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
[19] 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.
[20] J. Ke, H. Zheng, H. Yang, and X. Chen, "Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach," *Transp. Res. C, Emerg. Technol.*, vol. 85, pp. 591–608, Dec. 2017.
[21] K. F. Chu, A. Y. S. Lam, and V. O. K. Li, "Travel demand prediction using deep multi-scale convolutional LSTM network," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 1402–1407.
[22] H. Yao *et al.*, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proc. 32nd AAAI Conf. Artif. Intell.*, Feb. 2018, pp. 2588–2595.
[23] J. Ye, L. Sun, B. Du, Y. Fu, X. Tong, and H. Xiong, "Co-prediction of multiple transportation demands based on deep spatio-temporal neural network," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 305–313.
[24] X. Zhou, Y. Shen, Y. Zhu, and L. Huang, "Predicting multi-step citywide passenger demands using attention-based neural networks," in *Proc. 11th ACM Int. Conf. Web Search Data Mining (WSDM)*, 2018, pp. 736–744.
[25] L. Liu, Z. Qiu, G. Li, Q. Wang, W. Ouyang, and L. Lin, "Contextualized Spatial–Temporal network for taxi origin-destination demand prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3875–3887, Oct. 2019.
[26] Y. Lv, Y. Chen, X. Zhang, Y. Duan, and N. L. Li, "Social media based transportation research: The state of the work and the networking," *IEEE/CAA J. Automatica Sinica*, vol. 4, no. 1, pp. 19–26, Jan. 2017.
[27] Y. Chen, Y. Lv, X. Wang, L. Li, and F.-Y. Wang, "Detecting traffic information from social media texts with deep learning approaches," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 8, pp. 3049–3058, Aug. 2019.
[28] F. Rodrigues, I. Markou, and F. C. Pereira, "Combining time-series and textual data for taxi demand prediction in event areas: A deep learning approach," *Inf. Fusion*, vol. 49, pp. 120–129, Sep. 2019.
[29] Y. Lin, X. Dai, L. Li, and F.-Y. Wang, "Pattern sensitive prediction of traffic flow based on generative adversarial framework," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 6, pp. 2395–2400, Jun. 2019.
[30] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 3371–3408, Dec. 2010.
[31] Y. Duan, Y. Lv, Y.-L. Liu, and F.-Y. Wang, "An efficient realization of deep learning for traffic data imputation," *Transp. Res. Part C: Emerg. Technol.*, vol. 72, pp. 168–181, Nov. 2016.
[32] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.
[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

Chizhan Zhang received the bachelor's degree from the University of Chinese Academy of Sciences in 2018, where he is currently pursuing the master's degree in control theory and control engineering with the School of Artificial Intelligence. His research interests include traffic data mining and urban intelligent transportation systems.

Fenghua Zhu (Member, IEEE) received the Ph.D. degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2008. He is currently an Associate Professor with the State Key Laboratory of Management and Control for Complex Systems, China. His research interests include artificial transportation systems and parallel transportation management systems.

Xiao Wang (Member, IEEE) received the bachelor's degree in network engineering from the Dalian University of Technology in 2011 and the Ph.D. degree in social computing from the University of Chinese Academy of Sciences in 2016. She is currently an Assistant Researcher with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. Her research interests include social transportation, cyber movement organizations, artificial intelligence, and social network analysis.



Leilei Sun (Member, IEEE) received the B.S. and M.S. degrees from the School of Control Theory and Control Engineering, Dalian University of Technology, in 2009 and 2012, respectively, and the Ph.D. degree from the Institute of Systems Engineering, Dalian University of Technology, in 2017. He was a Post-Doctoral Research Fellow with the School of Economics and Management, Tsinghua University, from 2017 to 2019. He is currently an Assistant Professor with the State Key Laboratory of Software Development Environment (SKLSDE) and the Big Data Brain Computing Laboratory (BDBC Lab), Beihang University, Beijing, China. His research interests include machine learning and data mining. He has published many articles in the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING (TKDE), *Knowledge and Information Systems* (KAIS), and ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD).

Haina Tang received the Ph.D. degree from Sun Yat-sen University. She is currently an Associate Professor with the School of Artificial Intelligence, University of Chinese Academy of Sciences. Her research interests include network measurement and spatio-temporal data mining.

Yisheng Lv (Senior Member, IEEE) is currently an Associate Professor with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. He is also with the University of Chinese Academy of Sciences. His research interests include artificial intelligence for transportation, intelligent vehicles, and parallel traffic management and control systems.