Short-Term Prediction of Passenger Demand in Multi-Zone Level: Temporal Convolutional Neural Network With Multi-Task Learning

Kunpeng Zhang, Zijian Liu, and Liang Zheng

***Abstract*—Accurate short-term passenger demand prediction contributes to the coordination of traffic supply and demand. This paper proposes an end-to-end multi-task learning temporal convolutional neural network (MTL-TCNN) to predict the short-term passenger demand in a multi-zone level. Along with a feature selector named spatiotemporal dynamic time warping (ST-DTW) algorithm, this proposed MTL-TCNN is quite qualified for the multi-task prediction problem with the consideration of spatiotemporal correlations. Then, based on the car-calling demand data from Didi Chuxing, Chengdu, China, and taxi demand data from the New York City, the numerical results show that the MTL-TCNN outperforms both classic methods (i.e., historical average (HA), *v*-support vector machine (*v*-SVM), and XGBoost) and the state-of-the-art deep learning approaches [e.g., long short-term memory (LSTM) and convolutional LSTM (ConvLSTM)] in both the single task learning (STL) and multi-task learning (MTL) scenarios. In summary, the proposed MTL-TCNN with the ST-DTW algorithm is a promising method for short-term passenger demand prediction in a multi-zone level.**

***Index Terms*—Short-term passenger demand prediction, multi-task learning, deep learning, convolutional neural network.**

I. Introduction

THE short-term prediction of passenger demand benefits the real-time dispatch of vacant cars and the service level of the transportation system. To accurately predict short-term passenger demand over multi-zones, the spatiotemporal correlativity has to be captured and modeled inherently. Specifically, the passenger demand in the target zone for the next prediction cycle has some correlations with the historical and current passenger demands in the target zone and its neighboring zones. The passenger demand information from the neighboring zones, which has the similar pattern to that in the target zone, is more valuable, and the adjacent zones have stronger correlations with the target zone than the distant zones [1]. Moreover, the passenger demand has strong periodicities (e.g., daily and weekly) and thus the prediction accuracy depends on not only the short-term historical variables but also the long-term historical attributes [2].

However, limited research efforts have been given to address spatiotemporal dependencies at the early stage of short-term passenger demand prediction. For example, Ke *et al.* [2] and Zhou *et al.* [3] employed convolutional long short-term memory (ConvLSTM) to capture spatial and temporal relations for the short-term passenger demand prediction. Yao *et al.* [4] predicted the passenger demand with a deep multi-view spatial-temporal network (DMVST-NET), in which the spatial and temporal views were modeled respectively by long short-term memory (LSTM) and convolutional neural networks (CNN). Wang *et al.* [5] proposed an end-to-end framework called deep supply-demand (DeepSD) to automatically discover complicated supply-demand patterns across different spatiotemporal attributes. Tong *et al.* [6] proposed a unified linear regression model with high-dimensional features to capture spatiotemporal dependencies during the passenger demand prediction. Inspired by these successful applications aforementioned, this study will propose a novel deep learning structure named multi-task learning temporal convolutional neural network (MTL-TCNN) with residual mappings [7] to predict the short-term passenger demand in a multi-zone level. Notably, the residual mapping from the original input to the intermediate output makes MTL-TCNN become substantially deeper (i.e., with more neural network layers). Moreover, based on the correlation of passenger demand patterns over various zones, a spatiotemporal dynamic time warping (ST-DTW) algorithm is explored to select the most informative features for MTL-TCNN. This helps to exclude irrelevant features and reduce the computational complexity.

Therefore, the main contributions of this study cover three aspects: 1) The MTL-TCNN characterizes the spatiotemporal correlations for end-to-end short-term passenger demand prediction in a multi-zone level; 2) The ST-DTW algorithm builds the concise and effective input vector for MTL-TCNN; 3) Based on real-world passenger demand data, numerical experiments will validate the outperformance of the MTL-TCNN compared with other counterpart methods in terms of prediction accuracy and training time.

The remainder of this paper is organized as follows. Section II reviews the related studies on the short-term passenger demand prediction and highlights the state-of-the-art deep

Manuscript received October 22, 2018; revised January 26, 2019; accepted March 31, 2019. Date of publication April 15, 2019; date of current version March 27, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 51475152, Grant 71871227, and Grant 71501191 and in part by the Innovation Driven Plan of Central South University under Grant 20180016040002. The Associate Editor for this paper was Y. Lv. *(Corresponding authors: Zijian Liu; Liang Zheng.)*

K. Zhang and Z. Liu are with the State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body, College of Mechanical and Vehicle Engineering, Hunan University, Changsha 410082, China (e-mail: zijianliu@hnu.edu.cn).

L. Zheng is with the School of Traffic and Transportation Engineering, Central South University, Changsha 410083, China (e-mail: zhengliang@csu.edu.cn).

Digital Object Identifier 10.1109/TITS.2019.2909571

learning approaches for traffic prediction. Section III presents the structure of MTL-TCNN with residual mappings and the ST-DTW method. In Section IV, numerical experiments are conducted with the car-calling demand data from Didi Chuxing in Chengdu city, China, and taxi demand data from New York City. Section V draws some interesting conclusions.

II. Literature Review

As one type of passenger demand prediction, the taxi demand prediction has drawn extensive attention in the past decades. These related studies can provide valuable insights for the car-calling (e.g., Didi Chuxing, Uber, and Lyft) demand prediction due to their similarity. For example, Yuan *et al.* [8] presented a probabilistic approach to model the temporal dependences of taxi behaviors (e.g., picking-up and dropping-off) for taxi demand prediction. Moreira-Matias *et al.* [9] incorporated the time varying passion model and autoregressive integrated moving average (ARIMA) to predict the taxi demand. Zhang *et al.* [10] utilized an exponential weighted moving average (EWMA) method to predict passenger demand hotspots so as to provide the recommendations for taxi drivers. Xu *et al.* [11] employed LSTM and mixture density networks to carry out the citywide taxi demand prediction.

What deserves our attention is that deep learning based traffic prediction approaches have gained a fast development due to their capability of capturing the complex correlations (e.g., spatiotemporal correlations) in rich traffic data sources. For example, Lv *et al.* [12] proposed a deep learning based stacked autoencoder (SAE) model to capture spatial and temporal correlations in the traffic flow prediction. Huang *et al.* [13] predicted traffic flow via a two-layer deep learning structure, i.e., a deep belief network (DBN) as the bottom and a multi-task learning (MTL) model as the top. Koesdwiady *et al.* [14] incorporated a DBN and a decision-level data fusion scheme to predict traffic flow with weather information. Yang *et al.* [15] applied a stacked autoencoder Levenberg-Marquardt model to build a deep learning architecture for the traffic flow prediction. Polson and Sokolov [16] predicted traffic flow by a deep learning model composed of a linear model and a sequence of tanh layers, which is able to capture spatiotemporal dependencies of traffic flow in an urban area.

Moreover, many studies applied CNN to learn the local and global spatial correlations by converting network-wide traffic states as images. For example, Ma *et al.* [17] converted traffic speed dynamics to images, based on which CNN was employed to predict the network-wide traffic speed. Based on images converted from the citywide crowd flow data, Zhang *et al.* [18] utilized a residual CNN to predict the citywide crowd flow. However, it is not a trivial problem to explicitly convert traffic data into images without risking information mismatch. Even more, applying CNN for the entire network-wide traffic states would inevitably contain some irrelevant features, which could worsen the prediction accuracy and increase the computational complexity. In terms of temporal dependencies, Ma *et al.* [19] employed LSTM to capture the temporal characteristics for the traffic speed prediction. Yu *et al.* [20] applied LSTM and a stacked autoencoder to describe the temporal relations for the traffic state prediction under extreme conditions (e.g., peak-hour and post-accident scenarios). However, these two studies fail to account for spatial correlations.

Fortunately, numerous attempts were made to capture spatial and temporal correlations simultaneously in recent years [21], [22]. With a modified ordinary Kriging model, Thajchayapong and Barria [23] found that spatiotemporal characteristics could improve the estimation accuracy of lane-level flow and occupancy. Zheng *et al.* [24] proposed a feature selection based method to predict the urban traffic speed with the consideration of spatiotemporal traffic pattern. Yu *et al.* [25] predicted the network-wide traffic speed with a spatiotemporal recurrent convolutional network, in which a CNN layer and two LSTM layers were combined to capture spatiotemporal correlations. Wu *et al.* [26] proposed a deep learning based traffic flow prediction model with a CNN layer to learn spatial features and two Gated Recurrent Unit (GRU) layers [27] to capture temporal features of traffic flow. However, these models are based on either classic methods or deep learning structures with only several stacked neural network layers, which may restrict their prediction capacities. Meanwhile, the recurrent neural network (RNN) or its variants (e.g., LSTM and GRU) would cause high computational costs during training these models.

In this context, this study will propose an MTL-TCNN structure with residual mappings to construct a deeper learning model. It can not only capture spatiotemporal correlations inherently from multi-zone passenger demand data but also occupy a high training efficiency. Moreover, based on the correlations between various prediction tasks, an ST-DTW algorithm is presented to select the most informative features for MTL-TCNN.

III. Methodology

This section first describes the structure of the temporal convolutional neural network (TCNN) and its training efficiency, and then builds an MTL-TCNN model with residual mappings for short-term passenger demand prediction. Finally, the ST-DTW algorithm is proposed to select the most informative features based on the quantified correlations between various prediction tasks.

A. Temporal Convolutional Neural Network

Compared with RNN, which is specialized for processing time series data by introducing memory to retain information, CNN extracts data information for fixed size contexts, which makes CNN less common for time series modeling due to the lack of memory for a long sequence [28]. However, recent evidence indicates that some novel CNN based architectures can outperform RNN based ones in the fields of audio synthesis and machine translation [29], [30]. Encouraged by these successful applications, a modified TCNN is introduced to predict short-term passenger demand, which incorporates dilated causal convolutions and residual mappings.

The main component of TCNN is dilated causal convolutions. In the dilated causal convolutions, causal convolution

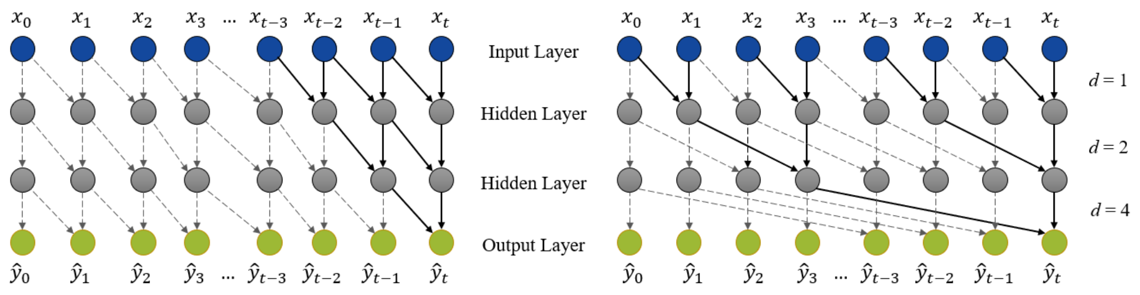

x_0 x_1 x_2 x_3 ... x_{t-3} x_{t-2} x_{t-1} x_t
Input Layer
Hidden Layer
Hidden Layer
Output Layer
$\hat{y}_0$ $\hat{y}_1$ $\hat{y}_2$ $\hat{y}_3$ ... $\hat{y}_{t-3}$ $\hat{y}_{t-2}$ $\hat{y}_{t-1}$ $\hat{y}_t$
$d = 1$
$d = 2$
$d = 4$

Fig. 1. Receptive fields of a stack of standard causal convolution layers and a stack of dilated causal convolution layers. *Note:* The dashed line and the solid line stand for computational procedures in parallel. (a) Standard causal convolution stack. (b) Dilated causal convolution stack.

layers [29] enable an output at timestamp t to be impacted only by the input from the current timestamp t and the past timestamps (i.e., *t-1, t-2, t-3, …*). Moreover, the receptive field is introduced to represent the capability of CNN to cover the inputs in the input sequence [31]. A larger receptive field indicates a broader range of inputs from the input layer. Fig. 1(a) shows the receptive field of a standard causal convolution stack. It is known that the output (i.e., $\hat{y}_t$) depends only on the inputs from the current timestamp (i.e., x_t) and the earlier timestamps (e.g., x_{t-1}, x_{t-2}, and x_{t-3}). Then, for an input sequence of one-dimensional time series X and a filter *f:{0,…, k-1}*, the convolution operation F for a standard causal convolution stack at the timestamp t is defined as

$$F(t) = \sum\nolimits_{i=0}^{\mathrm{k}-1} f(i) X_{t-i} \tag{1}$$

where $f(i)$ is the i^{th} filter of the corresponding layer (i.e., the hidden layer in Fig. 1(a)), k is the filter size and *t-i* is the timestamp of the past.

As for the standard causal convolution stack with the filter size $k = 2$ and the network depth $n = 4$ in Fig. 1(a), only 4 inputs impact the output due to the limited receptive field. This reveals a major disadvantage of the standard causal convolution stack. That is, a very large filter size k or an extremely deep network is needed to maintain a large receptive field, which helps to make an output be influenced by long and effective historical inputs. However, a large filter size k could cause the model non-convergence problem [30], which will worsen the prediction accuracy. Meanwhile, a deeper network will hurt the training stabilization and cause a degradation problem [7].

To address this first issue, TCNN employs the dilated causal convolutions (c.f., Fig. 1(b)), in which the dilation factor d increases exponentially with the network depth. In this way, TCNN could not only effectively expand the receptive field without requiring a large filter size k, but also achieve the high computational efficiency. For example, in Fig. 1(b), the output $\hat{y}_t$ depends on all inputs with few computational procedures. Then, the convolution operation F for a dilated causal convolution stack at the timestamp t is defined as

$$F(t) = \sum\nolimits_{i=0}^{\mathrm{k}-1} f(i) X_{t-d\cdot i} \tag{2}$$

With a proper filter size, the training stabilization of the deeper TCNN becomes critical to achieving a desirable prediction performance. To address this issue, the residual mapping is applied as a form of a residual block in the deeper TCNN. As illustrated in Fig. 2(a), a residual block contains a shortcut connection (i.e., the black curve) to perform the residual mapping from the input X_i to the transformation $F(X_i)$. This mapping process can be formulated as

$$X_{i+1} = \Phi(Z_i + X_i) \tag{3}$$

where $\Phi(.)$ denotes a nonlinearity activation operation and Z_i is the output of $F(X_i)$.

Besides, the residual block also encloses the weight normalization layer, activation layer, pooling layer, and dropout layer. Specifically, the weight normalization layer reparametrizes the weight vectors to speed up the convergence. A leaky rectified linear unit [32] (i.e., *LeakyReLU* in Fig. 2(a)) is utilized as the activation layer here. The pooling layer is a nonlinear down-sampling layer to reduce the spatial size of the representation and the computation complexity, as well as to avoid over-fitting. A spatial adaptive max pooling [33] (i.e., *AdaptiveMaxPool* in Fig. 2(a)) is employed here to provide a more flexible pooling function that allows for a larger pooling region. Moreover, a spatial dropout layer is applied to randomly dismiss neurons at each training step, and a convolutional layer is incorporated between two residual blocks to further improve the learning capability of the proposed model (c.f., Fig. 2(b, c)).

B. Training Efficiency Analysis of TCNN

TCNN outperforms RNN and its variants in training efficiency from two aspects. Firstly, as shown in Fig. 1(b), TCNN could carry out the predictions in parallel, and maintain a sufficient receptive field with a high computational efficiency. This enables TCNN to train a long data sequence as a whole, instead of processing these data sequentially as RNN and its variants. Thus, numerous variables can be handled in one process, which helps to reduce the training time dramatically. Secondly, RNN and its variants need to read and write intermediate results during the learning process. This routine not only requires a high memory but also increases computational costs. In contrast, TCNN does not suffer this problem because the layers in TCNN are mapped directly, which allows the current layer directly accesses the results of the previous layer without the help of intermediate results. This novel design enables TCNN to complete the learning process with lower memory usage and time consumption.

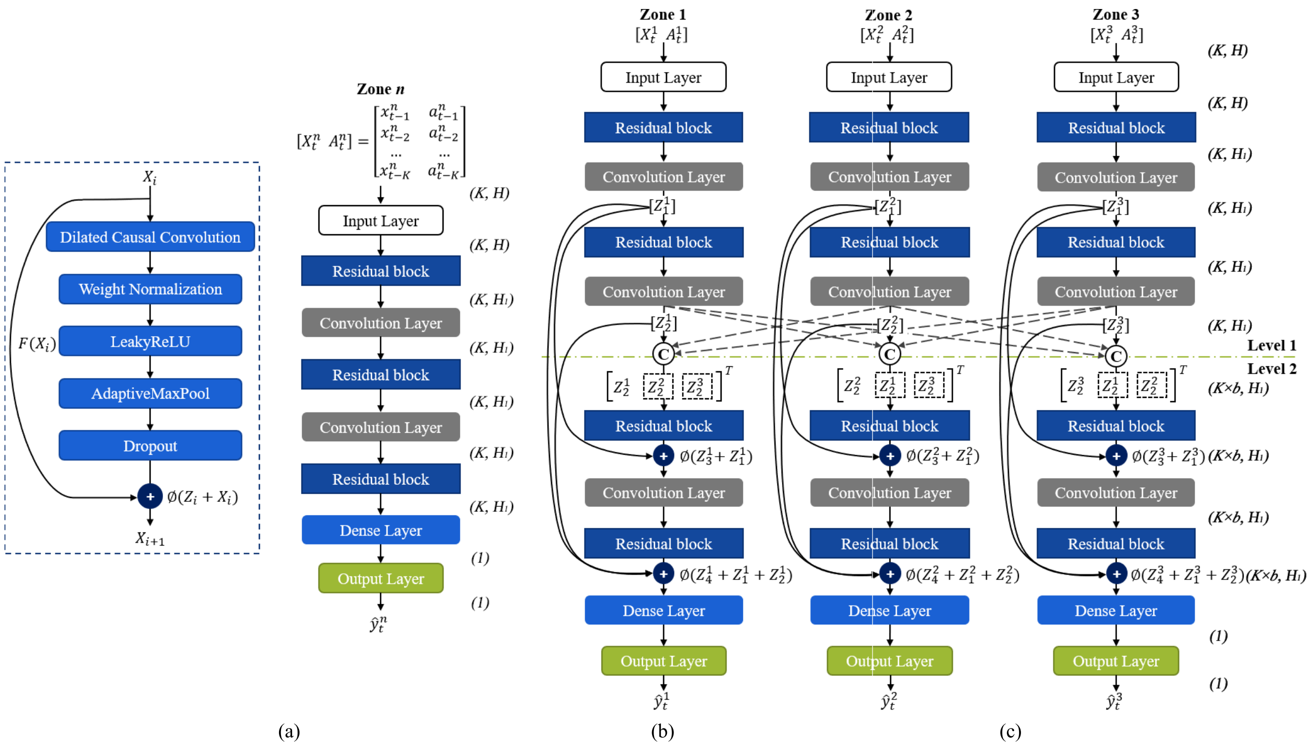


Fig. 2. Internal structures of STL-TCNN model and MTL-TCNN model with residual mappings. *Note:* © denotes a concatenation operation. The values in italic parentheses are the shapes of inputs or outputs of each layer. H_1 is the number of the hidden units of the corresponding layer. (a) Structure of a residual block. (b) Single task learning. (c) Multi-task learning.

C. Multi-Task Learning Temporal Convolutional Neural Network

Multi-task learning can take advantage of useful information from multiple related tasks. It is found that learning these tasks jointly can improve the prediction performance compared with learning them individually [35]. This section proposes an MTL model with TCNN to improve the generalization performance of all prediction tasks.

Taking the passenger demand prediction for the n^{th} zone as an example, Fig. 2(b) details the inner structure of an STL-TCNN model, the input of which is $[X_t^n A_t^n](n = 1, 2, \dots, N)$. N is the number of investigated zones in a study site. $x_{t-k}^n(k = 1, 2, \dots, K)$ represents the k^{th} previous passenger demand observation and a_{t-k}^n is the auxiliary variable corresponding to x_{t-k}^n. The shape of $[X_t^n A_t^n]$is *(K, H)*, where K is the look-back time window and H is the sum of the dimensions of x_{t-k}^n and a_{t-k}^n. $\hat{y}_t^n$ is the predicted passenger demand at time t. This STL-TCNN model carries out the prediction task for the n^{th} zone without information (e.g., spatiotemporal correlations) from other tasks. Fig. 2(c) details the inner structure of an MTL-TCNN model with three prediction tasks as an example, which allows to improve the prediction performance for each task by considering the spatiotemporal correlations with other tasks. Compared with the STL model in Fig. 2(b), the MTL model contains not only the multiple prediction branches but also two levels of the learning process. In the first level (i.e., Level 1), each prediction task is trained individually to extract temporal features from the raw input data (i.e., $[X_t^n A_t^n](n = 1, 2, 3)$). Then, the extracted features (i.e., $[Z_2^n]$) are concatenated (© in Fig. 2(c)) and then fed into the residual blocks in Level 2, which helps to capture the spatiotemporal correlations among various prediction tasks in the form of high-level features. Moreover, the residual mapping indicated by the black curve in Fig. 2(c) creates a shortcut connection between these two levels. Along with the residual mapping in the residual block of TCNN, the network of the MTL model can be very deep without suffering the degradation problem. Notably, the dashed line denotes the conditional connection, which means only the most related tasks are connected. That is, the features from other tasks (c.f., dashed boxes in Fig. 2(c)) will be included (excluded) if these tasks are (not) sufficiently correlated with the target task. b is the number of correlated tasks for each target task (c.f., Fig. 2(c)). As b increases, the computational complexity will grow exponentially, especially for a deeper network. Moreover, the features from uncorrelated tasks could worsen the prediction accuracy. Thus, based on the dynamic time warping (DTW) algorithm [36], the following section will introduce an ST-DTW algorithm to quantify the task correlation, which helps to select the most informative features.

D. Spatiotemporal Dynamic Time Warping

The DTW algorithm is originally developed to find an optimal alignment between two time series with the consideration of their temporal heterogeneity. In this study, the original DTW is referred as the temporal DTW, which is introduced as follows. Two time series X and Y, which represent the passenger demand of zone I and zone II respectively can be given as

$$X = x_1, x_2, \ldots, x_i \ldots, x_I \tag{4}$$

$$Y = y_1, y_2, \ldots, y_j, \ldots, y_J \tag{5}$$

To find the optimal alignment between X and Y, a warping path W is introduced, which can be given as

$$W = w_1, w_2, \ldots w_l, \ldots w_L \tag{6}$$

where $max(I, J) \leq L < I + J$, L is the length of W and $w_l = (i, j)$ is the l^{th} element of W. The warping path W defines a mapping between X and Y, which satisfies three conditions. *Boundary condition:* $w_1 = (1, 1)$ and $w_L = (I, J)$, which guarantee that every index of X and Y is considered. *Monotonicity condition:* Given $w_l = (i, j)$ then $w_{l-1} = (i', j')$ where $i\text{-}i' \geq 0$ and $j\text{-}j' \geq 0$. This ensures no backward path in W. *Continuity condition:*Given $w_l = (i, j)$ then $w_{l-1} = (i', j')$ where $i\text{-}i' \leq 1$ and $j - j' \leq 1$. This restricts only adjacent indexes are considered. Under these conditions, the optimal alignment between X and Y can be found in the form of the optimal warping path, which has minimal warping cost among all possible warping paths. In the warping path w_l, a warping cost measures the distance of the i^{th} element of X and the j^{th} element of Y, which is calculated by

$$T_{Dist}(i, j) = (x_i - y_j)^2 \tag{7}$$

With the distance $T_{Dist}(i, j)$, the optimal warping path can be found to represent the temporal DTW distance between X and Y by calculating the following recurrence.

$$\begin{aligned} T_D(i, j)_{I,II} = T_{Dist}(i, j) + \min[&T_D(i-1, j)_{I,II}, \\ &T_D(i, j-1)_{I,II}, T_D(i-1, j-1)_{I,II}] \end{aligned} \tag{8}$$

where $T_D(i, j)_{I,II}$ indicates the temporal DTW distance between zone I and zone II. It is the sum of the distance $T_{Dist}(i, j)$ calculated at the current timestamp and the minimum value of the temporal DTW distance at the adjacent timestamps.

However, the temporal DTW algorithm does not take into account the spatial distance, and thus is incapable to capture the spatial diversity. Inspired by the First Law of Geography - "near things are more related than distant things" [37], this study considers spatial impacts when calculating the correlations between various tasks. Then, the spatiotemporal DTW (ST-DTW) algorithm is given as

$$\begin{aligned} &ST_D(i, j)_{I,II} \\ &\quad = T_{Dist}(i, j) + \min[ST_D(i-1, j)_{I,II} + S_D(i-1, j), \\ &\qquad ST_D(i, j-1)_{I,II} + S_D(i, j-1), \\ &\qquad ST_D(i-1, j-1)_{I,II} + S_D(i-1, j-1)] \end{aligned} \tag{9}$$

where $ST_D(i, j)_{I,II}$ represents the ST-DTW distance between zone I and zone II. It includes the distance $T_{Dist}(i, j)$ measured at the current timestamp, and the minimum value for the sum of the ST-DTW distance and the corresponding spatial distance S_D at the adjacent timestamps. Notably, the spatial location information is needed to calculate S_D. For example, the spatial distance between x_i and y_j (i.e., $S_D(i, j)$) can be calculated

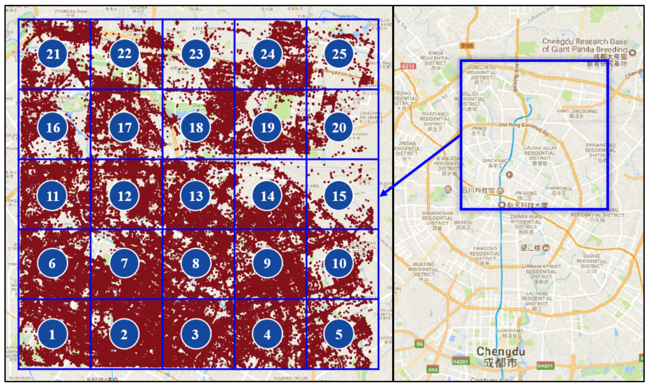


Fig. 3. 5 × 5 zones of study site.

with the longitude and latitude of x_i and y_j, which is expressed as

$$\begin{aligned} &S_D(i, j) \\ &= 2r \cdot \arcsin\left(\sqrt{\sin^2\left(\frac{\varphi_j - \varphi_i}{2}\right) + \cos(\varphi_i)\cos(\varphi_j)\sin^2\left(\frac{\lambda_j - \lambda_i}{2}\right)}\right) \end{aligned} \tag{10}$$

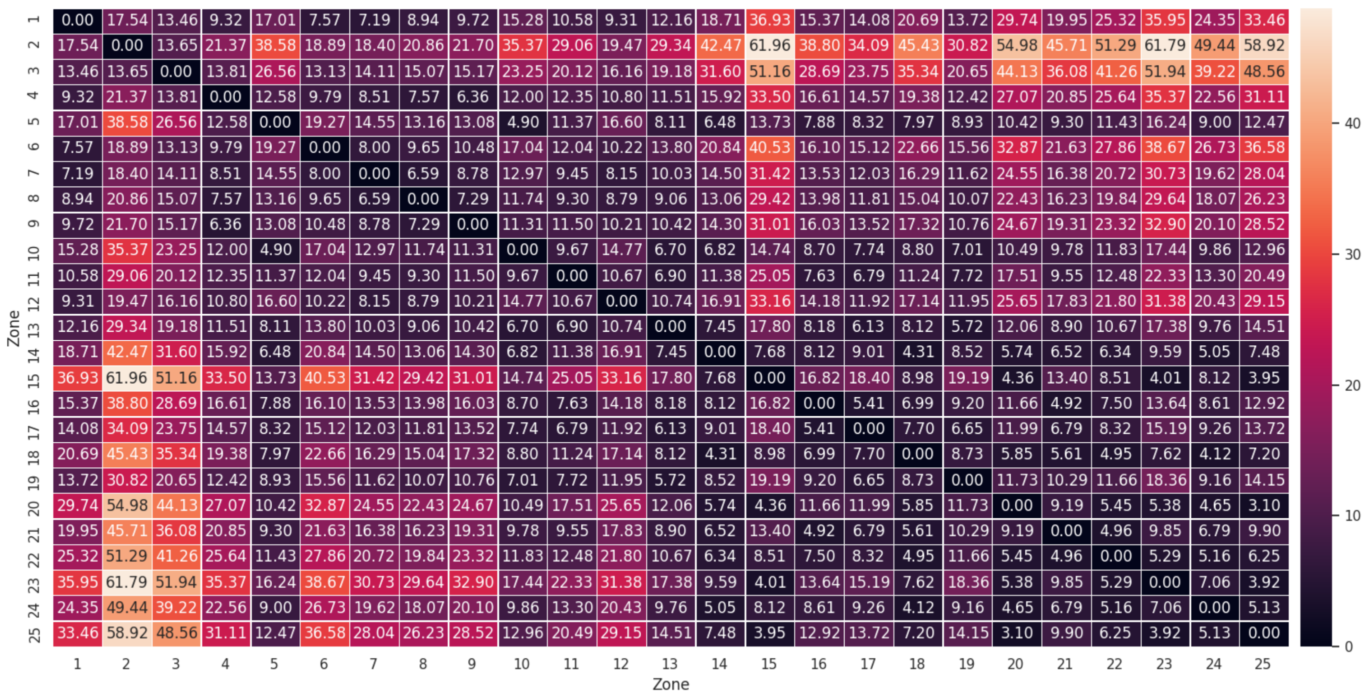
where r is the radius of the Earth, φ_i and φ_j are the latitudes of x_i and y_j, λ_i and λ_j are the longitudes of x_i and y_j.

As an extension of the DTW algorithm, the ST-DTW algorithm can test the ST-DTW distance between two time series provided their spatial location information available. Fortunately, the passenger demand data from the interest area include the location information (i.e., longitude and latitude) (c.f., Section IV. *A*), which makes the ST-DTW algorithm applicable in this study.

IV. Experiments and Results

A. Study Site Selection and Data Description

The study site is located in the downtown of Chengdu city, China, starting from 104.043°E to 104.130°E in longitude, and from 30.653°N to 30.726°N in latitude. This study site is partitioned into 5 × 5 zones and each zone is a rectangle with the length and the width both about 1.6 km, c.f., Fig. 3. The passenger demand datasets of these 25 zones are extracted from Didi Chuxing during the period from November 1 to November 30, 2016 [38]. These datasets contain about 120,000 car-calling requests per day, and each request includes the pick-up time, pick-up longitude, pick-up latitude, drop-off time, drop-off longitude and drop-off latitude. The passenger demand data in a zone for a time interval can be obtained by summing the car-calling requests from that zone during that time interval. This data is further separated into two groups: those for the first 23 days are treated as the training dataset and those for the last 7 days as the test dataset. In this study, the prediction cycle is set as 15 minutes, and the prediction step as 1, that is, the short-term passenger demand prediction is performed for every 15 minutes. Then, the time interval to count the passenger demand data is also set as 15 minutes, resulting in 96 data samples per day for each zone.



Zone	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	0.00	17.54	13.46	9.32	17.01	7.57	7.19	8.94	9.72	15.28	10.58	9.31	12.16	18.71	36.93	15.37	14.08	20.69	13.72	29.74	19.95	25.32	35.95	24.35	33.46
2	17.54	0.00	13.65	21.37	38.58	18.89	18.40	20.86	21.70	35.37	29.06	19.47	29.34	42.47	61.96	38.80	34.09	45.43	30.82	54.98	45.71	51.29	61.79	49.44	58.92
3	13.46	13.65	0.00	13.81	26.56	13.13	14.11	15.07	15.17	23.25	20.12	16.16	19.18	31.60	51.16	28.69	23.75	35.34	20.65	44.13	36.08	41.26	51.94	39.22	48.56
4	9.32	21.37	13.81	0.00	12.58	9.79	8.51	7.57	6.36	12.00	12.35	10.80	11.51	15.92	33.50	16.61	14.57	19.38	12.42	27.07	20.85	25.64	35.37	22.56	31.11
5	17.01	38.58	26.56	12.58	0.00	19.27	14.55	13.16	13.08	4.90	11.37	16.60	8.11	6.48	13.73	7.88	8.32	7.97	8.93	10.42	9.30	11.43	16.24	9.00	12.47
6	7.57	18.89	13.13	9.79	19.27	0.00	8.00	9.65	10.48	17.04	12.04	10.22	13.80	20.84	40.53	16.10	15.12	22.66	15.56	32.87	21.63	27.86	38.67	26.73	36.58
7	7.19	18.40	14.11	8.51	14.55	8.00	0.00	6.59	8.78	12.97	9.45	8.15	10.03	14.50	31.42	13.53	12.03	16.29	11.62	24.55	16.38	20.72	30.73	19.62	28.04
8	8.94	20.86	15.07	7.57	13.16	9.65	6.59	0.00	7.29	11.74	9.30	8.79	9.06	13.06	29.42	13.98	11.81	15.04	10.07	22.43	16.23	19.84	29.64	18.07	26.23
9	9.72	21.70	15.17	6.36	13.08	10.48	8.78	7.29	0.00	11.31	11.50	10.21	10.42	14.30	31.01	16.03	13.52	17.32	10.76	24.67	19.31	23.32	32.90	20.10	28.52
10	15.28	35.37	23.25	12.00	4.90	17.04	12.97	11.74	11.31	0.00	9.67	14.77	6.70	6.82	14.74	8.70	7.74	8.80	7.01	10.49	9.78	11.83	17.44	9.86	12.96
11	10.58	29.06	20.12	12.35	11.37	12.04	9.45	9.30	11.50	9.67	0.00	10.67	6.90	11.38	25.05	7.63	6.79	11.24	7.72	17.51	9.55	12.48	22.33	13.30	20.49
12	9.31	19.47	16.16	10.80	16.60	10.22	8.15	8.79	10.21	14.77	10.67	0.00	10.74	16.91	33.16	14.18	11.92	17.14	11.95	25.65	17.83	21.80	31.38	20.43	29.15
13	12.16	29.34	19.18	11.51	8.11	13.80	10.03	9.06	10.42	6.70	6.90	10.74	0.00	7.45	17.80	8.18	6.13	8.12	5.72	12.06	8.90	10.67	17.38	9.76	14.51
14	18.71	42.47	31.60	15.92	6.48	20.84	14.50	13.06	14.30	6.82	11.38	16.91	7.45	0.00	7.68	8.12	9.01	4.31	8.52	5.74	6.52	6.34	9.59	5.05	7.48
15	36.93	61.96	51.16	33.50	13.73	40.53	31.42	29.42	31.01	14.74	25.05	33.16	17.80	7.68	0.00	16.82	18.40	8.98	19.19	4.36	13.40	8.51	4.01	8.12	3.95
16	15.37	38.80	28.69	16.61	7.88	16.10	13.53	13.98	16.03	8.70	7.63	14.18	8.18	8.12	16.82	0.00	5.41	6.99	9.20	11.66	4.92	7.50	13.64	8.61	12.92
17	14.08	34.09	23.75	14.57	8.32	15.12	12.03	11.81	13.52	7.74	6.79	11.92	6.13	9.01	18.40	5.41	0.00	7.70	6.65	11.99	6.79	8.32	15.19	9.26	13.72
18	20.69	45.43	35.34	19.38	7.97	22.66	16.29	15.04	17.32	8.80	11.24	17.14	8.12	4.31	8.98	6.99	7.70	0.00	8.73	5.85	5.61	4.95	7.62	4.12	7.20
19	13.72	30.82	20.65	12.42	8.93	15.56	11.62	10.07	10.76	7.01	7.72	11.95	5.72	8.52	19.19	9.20	6.65	8.73	0.00	11.73	10.29	11.66	18.36	9.16	14.15
20	29.74	54.98	44.13	27.07	10.42	32.87	24.55	22.43	24.67	10.49	17.51	25.65	12.06	5.74	4.36	11.66	11.99	5.85	11.73	0.00	9.19	5.45	5.38	4.65	3.10
21	19.95	45.71	36.08	20.85	9.30	21.63	16.38	16.23	19.31	9.78	9.55	17.83	8.90	6.52	13.40	4.92	6.79	5.61	10.29	9.19	0.00	4.96	9.85	6.79	9.90
22	25.32	51.29	41.26	25.64	11.43	27.86	20.72	19.84	23.32	11.83	12.48	21.80	10.67	6.34	8.51	7.50	8.32	4.95	11.66	5.45	4.96	0.00	5.29	5.16	6.25
23	35.95	61.79	51.94	35.37	16.24	38.67	30.73	29.64	32.90	17.44	22.33	31.38	17.38	9.59	4.01	13.64	15.19	7.62	18.36	5.38	9.85	5.29	0.00	7.06	3.92
24	24.35	49.44	39.22	22.56	9.00	26.73	19.62	18.07	20.10	9.86	13.30	20.43	9.76	5.05	8.12	8.61	9.26	4.12	9.16	4.65	6.79	5.16	7.06	0.00	5.13
25	33.46	58.92	48.56	31.11	12.47	36.58	28.04	26.23	28.52	12.96	20.49	29.15	14.51	7.48	3.95	12.92	13.72	7.20	14.15	3.10	9.90	6.25	3.92	5.13	0.00

Fig. 4. ST-DTW distances between 25 zones.

After that, the Max-Min normalization is utilized to transform the passenger demand values of all zones into the range [-1, 1]. Meanwhile, inspired by the previous studies [2], [5], Time variables (including time-of-day and day-of-week) are incorporated in the model by one-hot encoding, which are also treated as the auxiliary variables (c.f., Fig. 2) to describe the passenger demand that varies across different timestamps of a day and different days of a week. Therefore, when the look-back time window K is set as 6, the shape of the input is $(K = 6, H = 104)$.

B. Spatiotemporal Dynamic Time Warping Test

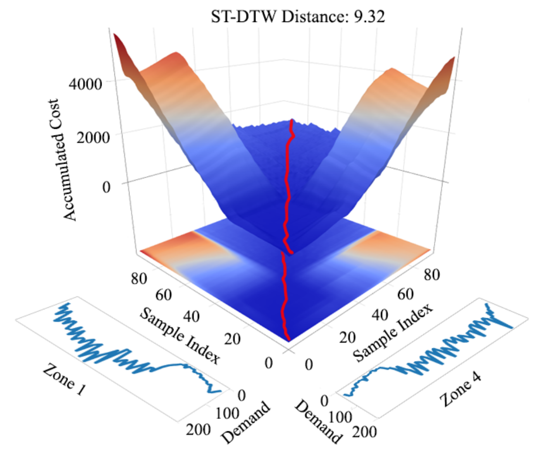
In this study, a task is defined as the short-term passenger demand prediction for one target zone. Due to the research purpose of the passenger demand prediction in a multi-zone level, 25 prediction tasks exist in the study site, c.f., Fig. 3. To select the most informative features for each prediction task, the ST-DTW algorithm is used to evaluate the correlation between the target zone and each of the rest 24 zones based on their short-term passenger demand data. Then, 24 ST-DTW distances are computed for each zone. Fig. 4 illustrates the ST-DTW distances of 25 zones, in which a smaller value indicates a shorter ST-DTW distance, i.e., a higher correlation between two prediction tasks.

Moreover, Fig. 5 demonstrates the ST-DTW distances between zone 1 and zone 4, and between zone 1 and zone 15. Specifically, Fig. 5(a) illustrates a "valley" in the accumulated cost matrix, and the ST-DTW distance is 9.32. Whereas, a sharp uptrend appears in Fig. 5(b) and the ST-DTW distance reaches 36.93. Therefore, compared with the ST-DTW distance between zone 1 and zone 15, a higher correlation between zone 1 and zone 4 exist.

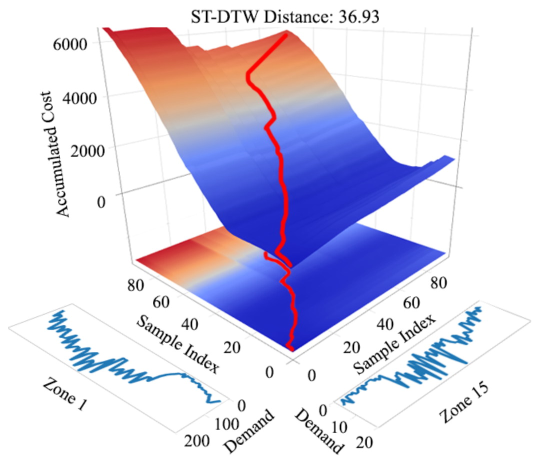
C. Model Comparisons

In order to validate the outperformance of the proposed MTL-TCNN with the ST-DTW test as the feature selector, various counterpart methods are presented and compared in STL and MTL scenarios. In both scenarios, a look-back time window K is selected to determine the number of previous data samples and their Time variables, which are taken as the inputs for the passenger demand prediction for the next 15 minutes.

1) *HA:* The historical average model carries out the prediction by averaging the historical passenger demand values at the same timestamps.
2) *υ-SVM:* SVM is usually applied to classification, regression, signal processing etc. It can handle the problems with the nonlinearity and high dimension due to the structural risk minimization principle and the convex quadratic programming. Inspired by Dong *et al.* [39], υ-SVM is employed here. 25 υ-SVM models are developed to predict passenger demand for 25 zones individually. A set of past values of variables (i.e., the demand data of each zone) are selected and fed into the corresponding υ-SVM model, which only considers temporal correlations.
3) *XGBoost:* XGBoost is a scalable machine learning system for tree boosting [40]. It is widely used in many machine learning challenges with an excellent performance. A set of proper variables (i.e., the passenger demand data and their Time variables) of each zone are reshaped to a vector and then fed into the corresponding XGBoost model for training and prediction.
4) *STL-LSTM:* Its structure is similar to that shown in Fig. 2(b) with a single input and a single output.



(a)



(b)

Fig. 5. 2D and 3D graphical demonstration of accumulated cost matrixes. (a) ST-DTW test between zone 1 and zone 4. (b) ST-DTW test between zone 1 and zone 15.

TABLE I
PREDICTION ERRORS AND EFFICIENCY FOR CHENGDU CITY

Model	RMSE	$MAPE_{10}$	MAE	Time (min)
HA	14.890	25.413%	10.325	0.05
v-SVM	11.960	20.344%	8.374	1.39
XGBoost	11.695	18.465%	8.356	1.12
STL-LSTM	9.541	15.220%	6.812	27.42
STL-ConvLSTM	9.526	15.386%	6.791	21.14
STL-TCNN	9.507	15.248%	6.568	9.36
MTL-LSTM	9.494	15.180%	6.708	15.29
MTL-ConvLSTM	9.417	15.154%	6.774	19.08
FCL-Net	9.480	15.405%	6.816	14.79
FTCNN-Net	9.406	15.292%	6.721	10.82
MTL-TCNN	9.337	14.950%	6.553	10.97
MTL-TCNN (ST-DTW)	**8.935**	**14.515%**	**6.258**	**7.68**

TABLE II
PREDICTION ERRORS AND EFFICIENCY FOR NEW YORK CITY

Model	RMSE	$MAPE_{10}$	MAE	Time (hour)
HA	21.173	36.426%	15.493	0.0003
v-SVM	11.994	20.770%	8.960	2.30
XGBoost	10.982	17.477%	8.146	0.54
STL-LSTM	10.611	17.402%	7.707	6.18
STL-ConvLSTM	10.457	17.417%	7.529	6.22
STL-TCNN	10.306	17.153%	7.396	3.70
MTL-LSTM	10.274	16.808%	7.367	5.39
MTL-ConvLSTM	10.280	16.854%	7.431	5.60
FCL-Net	10.463	17.058%	7.529	5.98
FTCNN-Net	10.167	16.941%	7.406	3.77
MTL-TCNN	9.930	16.672%	7.360	4.01
MTL-TCNN (ST-DTW)	**9.680**	**16.014%**	**7.164**	**3.16**

Instead of the residual block and convolution layer, LSTM is employed as the kernel neural unit in the STL-LSTM model, which will be built for 25 times to predict passenger demand of 25 zones. The passenger demand data and their Time variables of each zone are taken as the input of the corresponding STL-LSTM model to predict passenger demand with the consideration of temporal dependencies.

5) *STL-ConvLSTM:* It modifies the STL-LSTM model by employing ConvLSTM as the kernel neural unit. STL-ConvLSTM takes the passenger demand and their Time variables of each zone as the input and captures the temporal dependencies.
6) *STL-TCNN:* Its structure is shown in Fig. 2(b). A set of demand data and their Time variables of each zone is used in the corresponding STL-TCNN model to predict passenger demand with temporal correlations considered.

In the MTL scenario, four types of state-of-the-art deep learning approaches are introduced to compare with the proposed MTL-TCNN model without feature selection (i.e., MTL-TCNN in Table I, II) and that with the feature selected by ST-DTW (i.e., MTL-TCNN (ST-DTW) in Table I, II). To ensure the comparable condition, these four approaches to be introduced are also without the feature selection. The passenger demand data and their Time variables of 25 zones are taken as inputs to simultaneously predict the passenger demands of 25 zones with the consideration of spatiotemporal dependencies.

1) *MTL-LSTM:* Its structure is similar to MTL-TCNN shown in Fig. 2(c). Without residual mappings, MTL-LSTM employs LSTM as the kernel neural unit instead of the residual block and convolution layer.
2) *MTL-ConvLSTM:* It modifies the MTL-LSTM model by employing ConvLSTM instead of LSTM.
3) *FCL-Net:* Its structure is similar to that in [2]. That is, a fusion convolutional long short-term memory network (FCL-Net) is developed by employing ConvLSTM as the kernel neural unit.
4) *FTCNN-Net:* It modifies the FCL-Net by employing TCNN instead of ConvLSTM.

All these numerical experiments are carried out in one desktop computer equipped with Core i7-7700 central processing unit, 16 GB memory, and a GeForce GTX 1080 graphics processing unit. The look-back time window K is set as 6 for υ-SVM, XGBoost and deep learning based methods. Random search is employed to optimize the hyper-parameters of all methods, which performs better than other strategies (e.g., grid search and manual search) [41]. Specifically, the structure of the deep learning based model (e.g., the selection of optimizer and hidden layers) is optimized along with the selection of hyper-parameters (e.g., batch size, learning rate, the number of hidden units, and filter size). Meanwhile, with 10% training data as the validation dataset, the deep learning model training will be terminated when the validation error does not change for 10 epochs. As for TCNN based models, the filter size equaled to 7 is sufficient to guarantee a desirable performance. The prediction accuracy is evaluated by three measurements: root mean squared error (RMSE), mean absolute percentage error (MAPE) and mean absolute error (MAE), formulated as

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Y_i - \bar{Y}_i)^2} \tag{11}$$

$$MAPE_{10} = \frac{1}{N}\sum_{i=1}^{N}\frac{|Y_i - \bar{Y}_i|}{Y_i} \tag{12}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|Y_i - \bar{Y}_i| \tag{13}$$

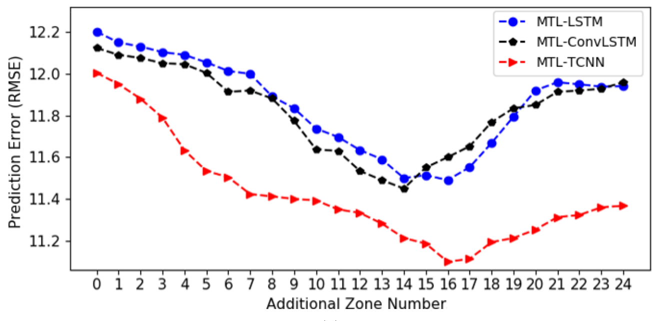
where Y_i and $\bar{Y}_i$ are the i^{th} ground truth and the predicted value of passenger demand respectively, MAPE$_{10}$denotes MAPE with the passenger demand value greater or equal to 10 in the test data samples.

Table I lists the mean prediction errors of 25 zones. It can be found that the proposed MTL-TCNN model outperforms the other counterparts in RMSE, MAPE$_{10}$, and MAE. Moreover, the MTL-TCNN (ST-DTW) model achieves the best prediction accuracy (i.e., RMSE=8.935, MAPE$_{10}$ =14.515%, and MAE=6.258), and reduces about 30% training time compared with the MTL-TCNN model with all features. This great reduction of training time can be explained as follows. With the irrelevant features excluded by ST-DTW, the computation complexity for the MTL-TCNN model is reduced significantly, which results in the great reduced training time. Furthermore, STL-TCNN costs less time than STL-LSTM because it has a superior structure and needs a smaller number of training epochs before stopping the training. Although STL-LSTM takes less time than MTL-LSTM, running STL-LSTM for 25 times would consume more time than MTL-LSTM because of the aggregation effect. On the other hand, due to that STL-TCNN spends much less time than MTL-TCNN, the time to run STL-TCNN for 25 times is still less than MTL-TCNN, although with the aggregation effect. It is also worth noticing that the TCNN models cost a shorter training time than the counterpart LSTM and ConvLSTM models.

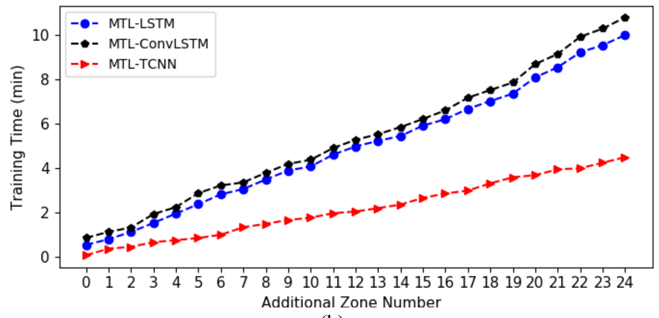
In summary, compared with the STL models, the MTL models with the consideration of spatiotemporal correlations can achieve a better prediction accuracy. In both STL and MTL scenarios, with limited computational costs, the MTL with TCNN can make a more accurate prediction than the conventional methods (i.e., HA, υ-SVM, and XGBoost) and those state-of-the-art neural networks. Furthermore, the ST-DTW algorithm plays a positive role in improving the prediction performance of MTL-TCNN.

D. Performance of MTL-TCNN

This section aims to further compare MTL-TCNN with its two counterparts (i.e., MTL-LSTM and MTL-ConvLSTM) in



(a)



(b)

Fig. 6. Prediction performance of three MTL models. (a) Prediction accuracies of three models for zone 1. (b) Training time of three models for zone 1.

both prediction accuracy and training efficiency. Here, zone 1 is chosen as the target zone for the prediction. Firstly, a benchmark MTL model with a single input and a single output is developed to predict the passenger demand of zone 1. Then, the prediction tasks for the rest 24 zones are cumulatively incorporated into the benchmark model according to their ST-DTW distances with zone 1 (c.f., Fig. 4). The labels of orderly selected zones are 7, 6, 8, 12, 4, 9, 11, 13, 3, 19, 17, 10, 16, 5, 2, 14, 21, 18, 24, 22, 20, 25, 23 and 15. After that, these three MTL models with a various number of prediction tasks are fine-tuned, and their prediction accuracies for zone 1 and training time are collected and demonstrated in Fig. 6. It is known that the MTL-TCNN models obtain the best performance on both prediction accuracy and training time. Specifically, with the increase of additional zone numbers, RMSE first drops and then rises (c.f., Fig. 6(a)). The reason can be analyzed as follows. As more prediction tasks incorporated, the MTL models perform a better prediction due to more spatiotemporal correlations considered. However, the redundant spatiotemporal information resulting from the higher number of additional zones would veil the critical spatiotemporal dependency, which would worsen the prediction accuracies of these three MTL models.

On the other hand, although the training time monotonously grows as additional zone numbers increase (c.f., Fig. 6(b)), the MTL-TCNN models cost the least computational time to obtain the best prediction results. Specifically, when the additional zone number reaches 24, the training time of MTL-LSTM, MTL-ConvLSTM, and MTL-TCNN is 10.00, 10.80 and 4.50 minutes, respectively. It means MTL-TCNN could save about 55% training time compared with MTL-LSTM, and about 58% training time compared with MTL-ConvLSTM. Finally, Fig. 7 shows the best predicted

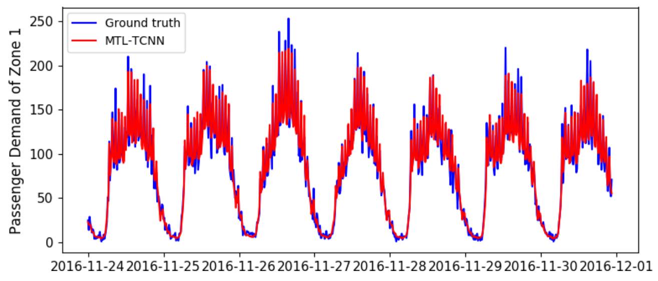


Fig. 7. Passenger demand values predicted for zone 1.

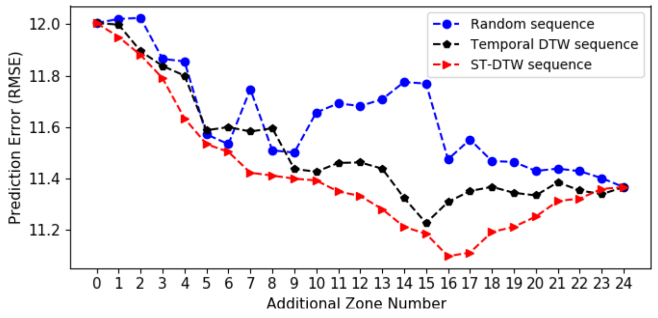


Fig. 8. Prediction accuracies of MTL-TCNN with three feature selection methods.

results for zone 1 by the MTL-TCNN model with 16 additional zones, which agree well with the real-field values.

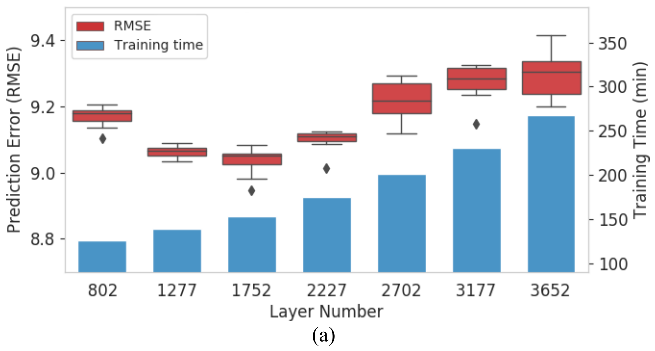
E. Reliability of the ST-DTW Test

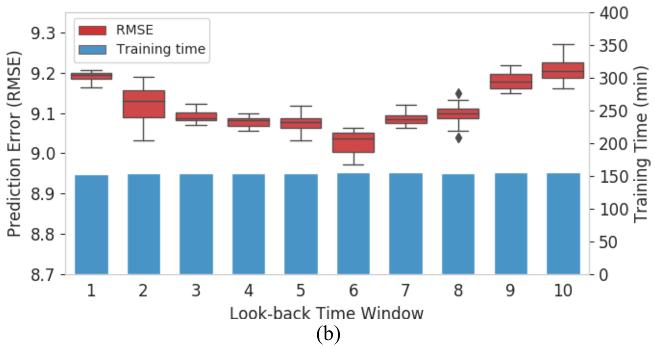
To validate the reliability of the ST-DTW test, this section reuses the MTL-TCNN models developed in Section IV *D* with two other feature selection strategies: random selection and the temporal DTW test. These two strategies are also utilized to select the additional zones cumulatively. That is, the labels of additional zones follow a random sequence (i.e., 16, 17, 12, 7, 20, 13, 4, 23, 14, 2, 24, 9, 15, 8, 11, 3, 22, 21, 19, 18, 6, 10, 25, 5), or follow a temporal DTW sequence according to the temporal DTW test (i.e., 7, 4, 6, 9, 12, 8, 11, 19, 13, 17, 10, 3, 16, 5, 14, 21, 18, 2, 24, 22, 20, 25, 23, 15).

After that, the MTL-TCNN models with a various number of prediction tasks are fine-tuned, and their prediction results for zone 1 are compared with those by MTL-TCNN with the ST-DTW sequence in Section IV *D*. Fig. 8 demonstrates that the MTL-TCNN models with the features selected randomly have an unstable prediction performance. The MTL-TCNN models with the temporal DTW test perform better in terms of performance stability and prediction accuracy. While, the MTL-TCNN models with the ST-DTW test obtain the best prediction performance. This validates that the ST-DTW test is reliable to select the most informative features.

F. Sensitivity Analysis

This section carries out the sensitivity analyses of the MTL-TCNN (ST-DTW) model (in Section IV *C*) on the network depth, look-back time window K, and prediction steps. Firstly, to investigate the impact of the network depth on the prediction performance, the network depth is increased by stacking the residual block and convolutional layer (c.f., Fig. 2(c)), and





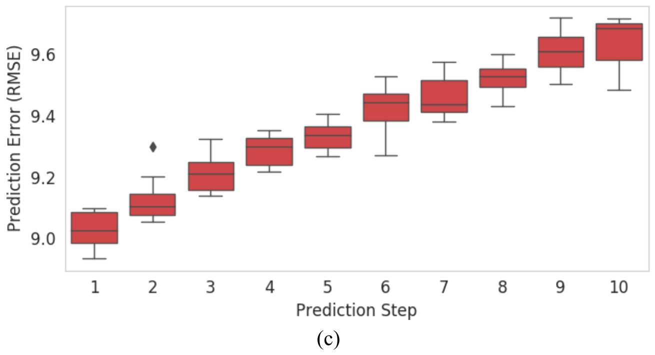


Fig. 9. Sensitivity analyses for MTL-TCNN. (a) Network depth. (b) Look-back time window. (c) Multi-step prediction.

then all neural network layers in the MTL-TCNN structure are counted to obtain the layer number. After fine-tuning, each MTL-TCNN model with a various number of layers is executed for 20 times, and their prediction errors and training time are recorded and demonstrated by the box plot in Fig. 9(a). It is shown that as the number of layers increases, RMSE first drops and then rises, while the computational time monotonously increases. Notably, the MTL-TCNN model with a very large network depth could face the over-fitting issue, which explains the growth of RMSE after the number of layers reaches 1752.

Then, various K values are designed for the MTL-TCNN model with 1752 layers to investigate the impact of the look-back time window on the prediction performance. Each MTL-TCNN model with each K value is fine-tuned and executed for 20 times. Fig. 9(b) demonstrates that RMSE first decreases and then increases with the growth of K value. It is explained as that a longer look-back time window could cover the redundant information, which worsens the prediction performance. It is also worth noticing that the K value has a negligible impact on the training time, which further confirms the well training efficiency of TCNN.

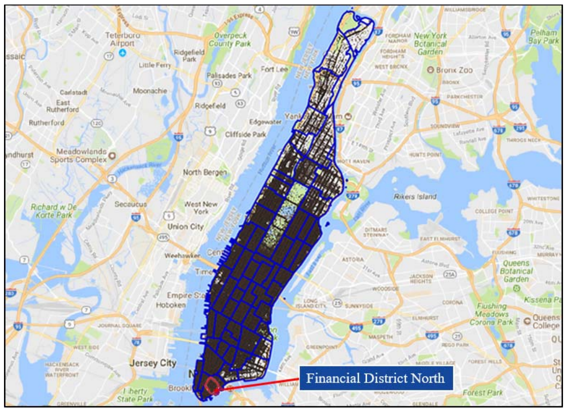


Fig. 10. 63 zones of study site in New York City.

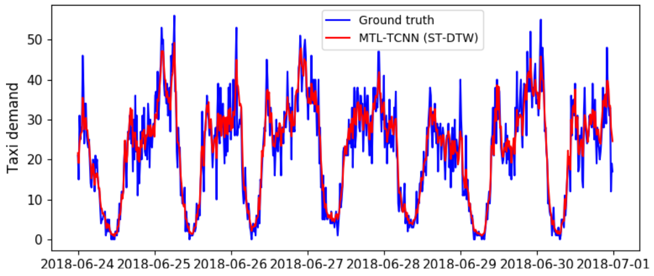


Fig. 11. Taxi demand values predicted for Financial District North.

After that, with a proper network depth (i.e., 1752) and a suitable K (i.e., 6), each MTL-TCNN model with each prediction step is fine-tuned to carry out the prediction for 20 times. Fig. 9(c) illustrates that RMSE gradually increases with the increase of the prediction step. It is explained as that the past information becomes less valuable to facilitate the prediction in the longer future, which causes the reduction of the prediction accuracy. However, MTL-TCNN with 5 prediction steps still outperforms most prediction models in Table I, whose prediction step is 1. This implies that the proposed MTL-TCNN (ST-DTW) model is well applicable to the multi-step prediction.

G. Scalability of MTL-TCNN

To evaluate the scalability of MTL-TCNN, this section performs the short-term taxi demand prediction with the most recent taxi trip data during January 2016 to June 2018 in New York City [42]. Each taxi trip has the pick-up time, pick-up information, drop-off time, and drop-off information. The study region located in Manhattan has 63 zones, c.f., Fig. 10. The taxi demand data of these zones are further separated into three parts: 80% of the data are treated as the training dataset, 10% serve as the validation dataset and the remaining 10% are the test dataset.

Then, the ST-DTW distances between 63 zones are calculated, and the data processing is similar to that in Section IV *A*. With the same model structures but with re-selected hyper-parameters, 12 types of prediction models introduced in Section IV *C* are also employed to predict the taxi demand for the next 15 minutes. Table II lists the mean prediction errors of 63 zones. Obviously, the MTL-TCNN (ST-DTW) model achieves the best prediction accuracy, i.e., RMSE (9.680), MAPE_{10}(16.014%) and MAE (7.164). Notably, compared with the MTL-TCNN model (with all features), the MTL-TCNN (ST-DTW) model reduces RMSE at about 2.5%, and the training time is saved at about 21%. Moreover, all prediction models except XGBoost have a little higher mean prediction errors compared with those in Table I. This can be explained as follows. With limited taxi demand data available in some zones (e.g., the northeast zones) of Manhattan, the demand requests are more random and without a trackable pattern, which results in a poor learning performance of the proposed model for those zones. Finally, Fig.11 illustrates the prediction results by the MTL-TCNN (ST-DTW) model over Financial District North (Wall Street is in this district) for the last 7 days in the test dataset. It can be clearly observed that the prediction results follow the real-field values very well.

V. Conclusions

This study proposes the multi-task learning temporal convolutional neural network to carry out the short-term passenger demand prediction with the consideration of spatiotemporal dependencies. In the MTL-TCNN, a superior TCNN structure is presented by incorporating the dilated causal convolutions and residual mappings to model the time series data with the high accuracy and efficiency. To select the most informative features for the MTL-TCNN model, the ST-DTW algorithm is put forward to quantify the spatiotemporal correlation between two prediction tasks. Then, two datasets are used to perform the numerical experiments. That is, the car-calling demand data from Didi Chuxing in Chengdu city, China, and the taxi trip dataset in New York City. Meanwhile, both classic methods (i.e., HA, υ-SVM, and XGBoost) and state-of-the-art deep learning approaches in both STL and MTL scenarios are introduced as the comparison counterparts. Based on the numerical results, some useful findings are concluded as follows. 1) The MTL-TCNN model outperforms the counterpart models in terms of RMSE, MAPE_{10}, and MAE, implying its good capability of capturing the spatiotemporal dependencies. 2) The ST-DTW algorithm is particularly applicable to choose the most informative features for the prediction model, which helps to improve the prediction accuracy and meanwhile reduce the computational complexity. 3) Compared with LSTM and ConvLSTM, TCNN can dramatically reduce the training time due to its excellent internal structure. 4) The scalability of MTL-TCNN is successfully validated with the larger dataset from 63 zones of New York City during January 2016 to June 2018. In summary, the proposed MTL-TCNN model with the ST-DTW algorithm as the feature selector is promising to predict the short-term passenger demand in a multi-zone level.

References

[1] H. Yang, C. W. Y. Leung, S. C. Wong, and M. G. H. Bell, “Equilibria of bilateral taxi–customer searching and meeting on networks,” *Transp. Res. B, Methodol.*, vol. 44, nos. 8–9, pp. 1067–1083, Sep./Nov. 2010.

[2] 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.

[3] X. Zhou, Y. Shen, Y. Zhu, and L. Huang, “Predicting multi-step citywide passenger demands using attention-based neural networks,” in *Proc. 18th ACM Int. Conf. Web Search Data Min.*, Feb. 2018, pp. 736–744.

[4] H. X. Yao *et al.* (2018). "Deep multi-view spatial-temporal network for taxi demand prediction." [Oline]. Available: https://arxiv.org/abs/1802.08714

[5] D. Wang, W. Cao, J. Li, and J. Ye, "DeepSD: Supply-demand prediction for online car-hailing services using deep neural networks," in *Proc. IEEE 33rd Int. Conf. Data Eng.*, Apr. 2017, pp. 243–254.

[6] 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 Min.*, Aug. 2017, pp. 1653–1662.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[8] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, "T-finder: A recommender system for finding passengers and vacant taxis," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2390–2403, Oct. 2013.

[9] 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.

[10] K. Zhang, Z. Feng, S. Chen, K. Huang, and G. Wang, "A framework for passengers demand prediction and recommendation," in *Proc. IEEE Int. Conf. Services Comput. (SCC)*, Jun./Jul. 2016, pp. 340–347.

[11] J. Xu, R. Rahmatizadeh, L. Bölöni, 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.

[12] 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.

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

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

[15] H.-F. Yang, T. S. Dillon, and Y.-P. P. Chen, "Optimized structure of the traffic flow forecasting model with a deep learning approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2371–2381, Oct. 2016.

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

[17] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang. (2017). "Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction." [Online]. Available: https://arxiv.org/abs/1701.04245

[18] 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.

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

[20] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu, "Deep learning: A generic approach for extreme condition traffic forecasting," in *Proc. SIAM Int. Conf. Data Min.*, Jun. 2017, pp. 777–785.

[21] Z. He, Y. Lv, L. Lu, and W. Guan, "Constructing spatiotemporal speed contour diagrams: Using rectangular or non-rectangular parallelogram cells?" *Transp. B, Transp. Dyn.*, vol. 7, no. 1, pp. 44–60, Apr. 2017.

[22] Z. He, L. Zheng, P. Chen, and G. Wei, "Mapping to cells: A simple method to extract traffic dynamics from probe vehicle data," *Comput. Civ. Infrastruct. Eng.*, vol. 32, no. 3, pp. 252–267, Mar. 2017.

[23] S. Thajchayapong and J. A. Barria, "Spatial inference of traffic transition using micro–macro traffic variables," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 854–864, Apr. 2015.

[24] L. Zheng, C. Zhu, N. Zhu, T. He, N. Dong, and H. Huang, "Feature selection-based approach for urban short-term travel speed prediction," *IET Intell. Transp. Syst.*, vol. 12, pp. 474–484, Aug. 2018.

[25] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," *Sensors*, vol. 17, no. 7, p. 1501, Jun. 2017.

[26] Y. Wu, H. Tan, L. Qin, B. Ran, and Z. Jiang, "A hybrid deep learning based traffic flow prediction method and its understanding," *Transp. Res. C, Emerg. Technol.*, vol. 90, pp. 166–180, May 2018.

[27] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. (2014). "Empirical evaluation of gated recurrent neural networks on sequence modeling." [Online]. Available: https://arxiv.org/abs/1412.3555

[28] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[29] A. C. den Oord *et al.* (2016). "WaveNet: A generative model for raw audio." [online]. Available: https://arxiv.org/abs/1609.03499?context=cs

[30] S. Bai, J. Z. Kolter, and V. Koltun. (2018). "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling." [Online]. Available: https://arxiv.org/abs/1803.01271

[31] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the effective receptive field in deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 4898–4906.

[32] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," *Proc. ICML.*, vol. 30, Jun. 2013, p. 6.

[33] Y. Zhou, X. Sun, D. Liu, Z. Zha, and W. Zeng, "Adaptive pooling in multi-instance learning for Web video annotation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 318–327.

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

[35] K. Zhang, L. Zheng, Z. Liu, and N. Jia, "A deep learning based multitask model for network-wide traffic speed predication," *Neurocomputing*, to be published. [Online]. Available: https://authors.elsevier.com/tracking/article/details.do?aid=20646&jid=NEUCOM&surname=Zheng

[36] J. Kruskall and M. Liberman, "The symmetric time warping problem: From continuous to discrete," in *Time Warps, String Edits, and Macromolecules* (The Theory and Practice of Sequence Comparison). Boston, MA, USA: Addison-Wesley, 1983, pp. 125–161.

[37] W. R. Tobler, "A computer movie simulating urban growth in the detroit region," *Econ. Geogr.*, vol. 46, no. 1, pp. 234–240, Jun. 1970.

[38] *Didi Chuxing*. Accessed: 2018. [Online]. Available: https://gaia.didichuxing.com

[39] N. Dong, H. Huang, and L. Zheng, "Support vector machine in crash prediction at the level of traffic analysis zones: Assessing the spatial proximity effects," *Accid. Anal. Prev.*, vol. 82, pp. 192–198, Sep. 2015.

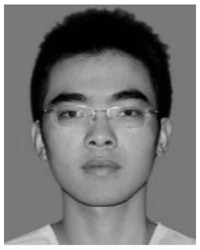
[40] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM Sigkdd Int. Conf. Knowl. Discov. Data Min.*, Aug. 2016, pp. 785–794.

[41] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, Feb. 2012.

[42] *NYC Taxi Limousine Commission. Taxi and Limousine Commission (TLC) Trip Record Data*. Accessed: 2018. [Online]. Available: http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

Kunpeng Zhang received the M.S. degree from the College of Mechanical Engineering, Zhengzhou University, China, in 2014. He is currently pursuing the Ph.D. degree with the College of Mechanical and Vehicle Engineering, Hunan University, China. His research interests include intelligent transportation systems and urban traffic control and management.

Zijian Liu received the M.S. and Ph.D. degrees from the College of Mechanical and Vehicle Engineering, Hunan University, China, in 1984 and 2001, respectively. He is currently a Professor with the College of Mechanical and Vehicle Engineering, Hunan University. His research interests include vehicle body design, process system optimization, and intelligent transportation systems.



Liang Zheng received the B.S. degree from Central South University, Changsha, China, in 2008, and the M.E. and Ph.D. degrees from Tianjin University, Tianjin, China, in 2010 and 2013, respectively. From 2011 to 2012, he spent one year as a Joint Doctoral Student with the University of Wisconsin, Madison, WI, USA. Since 2013, he has been with the School of Traffic and Transportation Engineering, Central South University, as an Associate Professor. His research interests cover macro- and micro-scopic traffic flow modeling and simulation, and data-driven short-term traffic prediction.