Co-Prediction of Multiple Transportation Demands Based on Deep Spatio-Temporal Neural Network

Junchen Ye[1], Leilei Sun[1,*], Bowen Du[1], Yanjie Fu[2], Xinran Tong[1], Hui Xiong[3]
[1]SKLSDE and BDBC Lab, Beihang University, Beijing 100083, China,
[2]Department of Computer Science, University of Central Florida, FL, USA,
[3]Department of Management Science and Information Systems, Rutgers University
[1]{yjchen,leileisun,dubowen,xinrantong}@buaa.edu.cn, [2]yanjiefoo@gmail.com, [3]hxiong@rutgers.edu

ABSTRACT

Taxi and sharing bike bring great convenience to urban transportation. A lot of efforts have been made to improve the efficiency of taxi service or bike sharing system by predicting the next-period pick-up or drop-off demand. Different from the existing research, this paper is motivated by the following two facts: 1) From a micro view, an observed spatial demand at any time slot could be decomposed as a combination of many hidden spatial demand bases; 2) From a macro view, the multiple transportation demands are strongly correlated with each other, both spatially and temporally. Definitely, the above two views have great potential to revolutionize the existing taxi or bike demand prediction methods. Along this line, this paper provides a novel Co-prediction method based on Spatio-Temporal neural Network, namely, CoST-Net. In particular, a deep convolutional neural network is constructed to decompose a spatial demand into a combination of hidden spatial demand bases. The combination weight vector is used as a representation of the decomposed spatial demand. Then, a heterogeneous Long Short-Term Memory (LSTM) is proposed to integrate the states of multiple transportation demands, and also model the dynamics of them mixedly. Last, the environmental features such as humidity and temperature are incorporated with the achieved overall hidden states to predict the multiple demands simultaneously. Experiments have been conducted on real-world taxi and sharing bike demand data, results demonstrate the superiority of the proposed method over both classical and the state-of-the-art transportation demand prediction methods.

CCS CONCEPTS

• **Information systems → Spatial-temporal systems**; • **Computing methodologies → Neural networks**.

KEYWORDS

Demand Prediction, Spatio-Temporal Analysis, Sharing Economy, Deep Neural Network

∗ Corresponding Author.

KDD '19, August 4–8, 2019, Anchorage, AK, USA

ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330887

ACM Reference Format:
Junchen Ye, Leilei Sun, Bowen Du, Yanjie Fu, Xinran Tong, Hui Xiong. 2019. Co-Prediction of Multiple Transportation Demands Based on Deep Spatio-Temporal Neural Network In *The 25th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (KDD'19), August 4–8, 2019, Anchorage, AK, USA.* ACM, NY, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330887

1 INTRODUCTION

Recent years have witnessed the rapid development of sharing economy, in which sharing transportation occupies a dominant position. However, the advent of a large number of transportation providers (Uber, Didi, Dida, Mobike, OFO, etc.) results in a competitive market. For example, the OFO, prevalent not long ago, has been indebted heavily and is going to be closed down. In this situation, the importance of precise management has been realized by many providers. Demand prediction is exactly one of the most important issues towards precise operation and maintenance.

To predict the transportation demand, many prediction methods have been proposed from different perspectives. Classical prediction methods mainly focus on the temporal characteristics of transportation demands, in which a studied area is first divided into many grids, then the transportation demand of each grid is predicted separately. In fact, this type of demand prediction can be formalized as a general time series prediction problem. Therefore, Autoregressive Integrated Moving Average (ARIMA) has been widely used in the existing literature [12]. Moreover, the next-period transportation demand could also be estimated by historical demands and external environmental features [10]. However, an intrinsic disadvantage of these methods is that the spatial relationship of transportation demands, such as dependence of neighborhood demands, has not been taken into account.

In recent years, deep neural networks have been explored to capture the spatial and temporal characteristics of demand distributions simultaneously. For example, Wei et al. provided an ensemble spatio-temporal neural network to predict passenger demand for chauffeured car service [14]. Zhang et al. employed a deep residual network to model the spatial correlation of neighborhood demands for citywide crowd flow prediction [22]. Ye et al. presented a multi-view spatio-temporal network to predict the taxi demands from three different views [18]. The above research demonstrates the effectiveness of deep spatio-temoral neural network in transportation demand prediction. However, there are still two important issues that have not been discussed ever before: 1) From a micro view, a snapshot spatial demand observed at any time slot could be actually decomposed as a combination of a series of hidden fixed spatial demand bases, see Figure 1 (*b*). The spatial demand bases can not

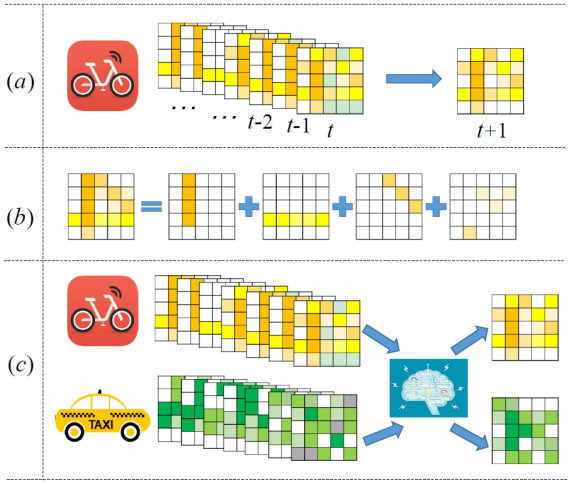

Figure 1: (*a*) The existing demand prediction method based on spatio-temporal analysis. (*b*) From a micro view, an observed spatial demand at time t can be decomposed into many hidden fixed spatial demand bases. (*c*) From a macro view, the multiple transportation demands are strongly correlated with each other.

only help us predict more precisely, but also present a reasonable explanation to transportation demand distribution. 2) From a macro view, the multiple transportation demands are strongly correlated with each other, both spatially and temporally. For example, both car and taxi are used to transport passengers from residential area to office area in the workday morning. The peaks of taxi pick-up demand in a workday are usually accompanied with bike pick-up peaks. Obviously, the above two points could be exploited to improve the performance of the existing transportation demand prediction methods.

Three challenges are faced to incorporate the above two ideas in transportation demand prediction. First, classical decomposition methods such as principal component analysis, non-negative matrix factorization, etc. can only capture the linear combination. Additionally, these methods cannot handle the spatial neighborhood correlation. Second, it is difficult to model the interactions of multiple transportation demands as they can affect each other both spatially and temporally. None of the existing demand prediction methods was designed for multiple demands co-prediction. Last, it is a non-trivial endeavor to take the impact of environmental factors into account, because the impact of them on different transportation services may be quite different.

To address these challenges, this paper proposes a Co-prediction method based on Spatio-Temporal neural Network (CoST-Net). First, a particular convolutional neural network is explored to discover the hidden spatial patterns of demand distribution. Based on this, any snapshot spatial demand could be encoded as a deep and nonlinear combination of the discovered spatial bases. Then, a general heterogeneous LSTM is studied to model the multiple transportation demands simultaneously. The hidden states of heterogeneous LSTM are driven by the dynamics of multiple transportation demands. Last, external environmental features are embedded to incorporate with the unifying hidden state of heterogeneous LSTM, and a fully connected feedforward neural network is designed to predict the multiple transportation demands jointly. In summary, this paper has the following contributions:

- A deep convolutional AutoEncoder is designed to implement the spatial decomposition of transportation demand. Based on the network, a snapshot transportation demand can be encoded as a combination of spatial demand bases.
- A heterogeneous LSTM is proposed to model the dynamics of multiple transportation demands simultaneously, and also integrate them into an unifying hidden state of the spatio-temporal neural network.
- A co-prediction module is proposed to predict multiple transportation demands according to the unifying hidden state and the external environmental factors.

2 PRELIMINARIES

In this section, we first formally introduce the preliminaries used in this paper, and then briefly revisit the transportation demand prediction problem.

2.1 Notations and definitions

Table 1 lists the notations that will be used throughout the paper.

Table 1: Notations and comments.

Notation	Comments
$\mathbb{L}, \mathbb{H}$	The set of locations, time intervals.
i, j, t	i and j are the index of locations, t is the indicator of time interval.
$\mathbb{E}, \mathbb{O}$	The set of external factors, orders.
$z, \boldsymbol{A}$	z is the coefficient of spatial demand bases, and $\boldsymbol{A}$ represents spatial demand bases.
$\boldsymbol{X}$	The demands of different transport, including pick-up demand and drop-off demand.
$\boldsymbol{Y}$	The ground truth.
q, n, k	The index of external features, orders, and spatial demand bases.

Definition 2.1. Time Interval, Location and External Factors: According to previous study [22], we split the whole city into an $I \times J$ grid map which consists of I rows and J columns. We define the locations as $\mathbb{L} = \{l_{0,0}, l_{0,1}, ...l_{i,j}, ..., l_{I,J}\}$. The set of time intervals is defined as $\mathbb{H} = \{h_1, h_2, ..., h_t, ...h_T\}$, and we set the 30 minutes as a time interval in this paper. Bike demand is susceptible to many external factors. For example, pick-up demands of bike in weekday are totally different from the demands in weekend, and the usage of bike decreases obviously with a sudden rain occurring. We take the external features into consideration, which are defined as $\mathbb{E}$, and e_t^q represents the q-th external feature in time interval t.

Definition 2.2. Order: According to previous work [18], o represents an order, and a bike demand order is defined as a 5-tuple

$o_B = (o_B^d.t, o_B^d.l, o_B^p.t, o_B^p.l, o_B.n)$, where o_B^d and o_B^p indicate the drop-off and pick-up of bike, t, l represent time interval and location, and n is the number of the order. o_C represents the taxi demand order, which consists of five components as well, $o_C = (o_C^d.t, o_C^d.l, o_C^p.t, o_C^p.l, o_C.n)$.

Definition 2.3. Demand: The demand of bike is defined as the total number of bikes getting into or leaving the grid region in a time interval. Here we have the definition of the demand:

$$\begin{aligned} x_t^{B,d,i,j} &= |\{o_B^d : o_B^d.t \in h_t \wedge o_B^d.l \in l_{i,j}\}| \\ x_t^{B,p,i,j} &= |\{o_B^p : o_B^p.t \in h_t \wedge o_B^p.l \in l_{i,j}\}|, \end{aligned} \tag{1}$$

where i, j represent the location (i, j), which lies at the i^{th} row and j^{th} column, and t is the time interval. The $||$ denotes the cardinality of a set, so $x_t^{B,d,i,j}$ and $x_t^{B,p,i,j}$ represent the number of bike drop-off and pick-up of grid (i, j) at time interval t. Demands in all regions at a time interval form a demand graph. $x_t^{C,d,i,j}$ and $x_t^{C,p,i,j}$ are taxi drop-off demand and taxi pick-up demand, which is defined as the same as the bike.

2.2 Problem Formalization

Given a set of historical observations until time interval t, the traffic volume prediction problem aims to forecast the drop-off and pick-up demands of bike and taxi at time interval t+1. External factors are also taken into consideration. We define external factors for region (i, j) at time interval t as a vector $e_t^l \in \mathbb{Q}^q$, where q is the number of factors. Thus, the prediction goal is defined as:

$$\hat{\mathbf{Y}}_{t+1}^L = \mathcal{F}(\mathbf{X}_{t-h,\ldots,t}^{B,L}, \mathbf{X}_{t-h,\ldots,t}^{C,L}, \boldsymbol{E}_t^L), \tag{2}$$

where L indicates all grids and $\mathbf{X}_{t-h,\ldots,t}^{B,L}$ represents the historical pick-up and drop-off demands of bike. Equally, historical pick-up and drop-off demands of taxi are $\mathbf{X}_{t-h,\ldots,t}^{C,L}$. The $\boldsymbol{E}_t^L$ is the historical external features at time interval t, and $\mathcal{F}$ is a predicting function.

3 METHODOLOGY

In the section, we introduce the details of CoST-Net. Figure 2 shows the architecture of our proposed method.

3.1 Spatial Demand Decomposition

In order to capture spatial and temporal sequential dependency, most of the exiting works combine CNN and LSTM to model two relationships simultaneously, but we have a totally different framework from them. They have missed the high-level correlation between demand graphs in the different time intervals. In order to capture the correlation, we assume that there are some spatial demand bases. They are invariant in different time intervals and every basis has its unique representation of the city. Moreover, the demands of all regions in per time interval can be viewed as the spatial bases' combination, which are defined as:

$$\boldsymbol{X}_t^L = \sum_{k=1}^{K} z_{t,k} \cdot \boldsymbol{A}_k^L, \tag{3}$$

where $\boldsymbol{X}_t = \{x_t^{i,j}\}$ is a type of spatial demand matrix, $x_t^{i,j}$ denotes the demand of region (i, j). Equation (3) means that a spatial demand matrix observed at time t could be seen as a combination of many hidden spatial demand bases as shown in Figure 1. Therefore, the spatial demand matrix $\boldsymbol{X}_t$ can be encoded as the combination coefficients of bases, that is, $\boldsymbol{Z}_t = \{z_{t,k}\},\ k = 1, 2, \cdots, K$.

To ensure the representation capacity of spatial demand bases, the combination of bases is designed in a deep and nonlinear manner. Therefore, the combination coefficients can be embedded in many folds:

$$\boldsymbol{Z}_t^{r+1} = \Psi(\boldsymbol{Z}_t^r), \tag{4}$$

where r is the number of embedding layers, Ψ is the function which transforms the low-level decomposition of snapshot sptial demand into high-level representation.

Correspondingly, we can get a nonlinear decomposition of $\boldsymbol{X}_t$ by bases of each layer. Take the r-th layer for example,

$$\boldsymbol{X}_t^L = f\left(\sum_{k=1}^{K^r} g_k\left(z_{t,k}^r \cdot \boldsymbol{A}_k^{L,r}\right)\right), \tag{5}$$

where $\{\boldsymbol{A}_1^r, \boldsymbol{A}_2^r, \cdots, \boldsymbol{A}_{K^r}^r\}$ are spatial demand bases of r-th layer, K^r is the number of bases.

3.1.1 Decompose to Obtain Spatial Demand Bases. In data mining techniques and empirical statistical methods, there are many methods to decompose to get essential bases. PCA, principal components analysis, which transforms a set of possibly correlated variables into a set of value of variables which is linearly uncorrelated, performing well in classical machine learning. In a sense, ICA(Independent Component Correlation Algorithm), clustering, etc., can also solve this problem. In recent years, deep learning has shown its advantages and has an excellent performance in image classification and natural language processing. Hinton and Salakhutdinov [2] proposed a method based on deep learning to obtain bases. They compared the results with PCA, and their method showed advantages in image restructuring which means that their method captured more typical bases. In 2013, Le [7] employed a huge autoencoder network to capture high-level representations and proved that the bases are multilayered. In the different layer of deep learning model, the obtained information is in different category. For example, in image recognition, the low layers might catch the outline shape of the object, and it is possible for the neuron in a high layer to recognize the human face or human body. We propose to use a deep convolutional AutoEncoder to obtain multilayered bases in the city transportation. The model is defined as:

$$\boldsymbol{Z}_t^L = f_{en}(\boldsymbol{X}_t^L * \boldsymbol{W}_{en} + \boldsymbol{b}_{en}), \tag{6}$$

$$\hat{\boldsymbol{X}}_t^L = f_{de}(\boldsymbol{Z}_t^L * \boldsymbol{W}_{de} + \boldsymbol{b}_{de}). \tag{7}$$

Equation (6) describes the encoder, where the feature graph $\boldsymbol{Z}_t^L$ is the output of the encoder. $\boldsymbol{W}_{en}, \boldsymbol{b}_{en}$ are trainable parameters of the whole encoder. The output of the encoder is employed as the input of the decoder. Equation (7) describes the decoder of the model, which has the symmetrical structure of the encoder. $\hat{\boldsymbol{X}}_t^L$, the restructuring of $\boldsymbol{X}_t^L$, is the output of the decoder. The depth of the encoder is R, as same as the decoder. The structure of the convolutional AutoEncoder will be elaborated in the section 3.1.2.

3.1.2 CNN Demand Model. Convolutional Neural Network(CNN) has shown advantages in image classification for its excellent ability in capturing spatial correlation. For each time interval t, we

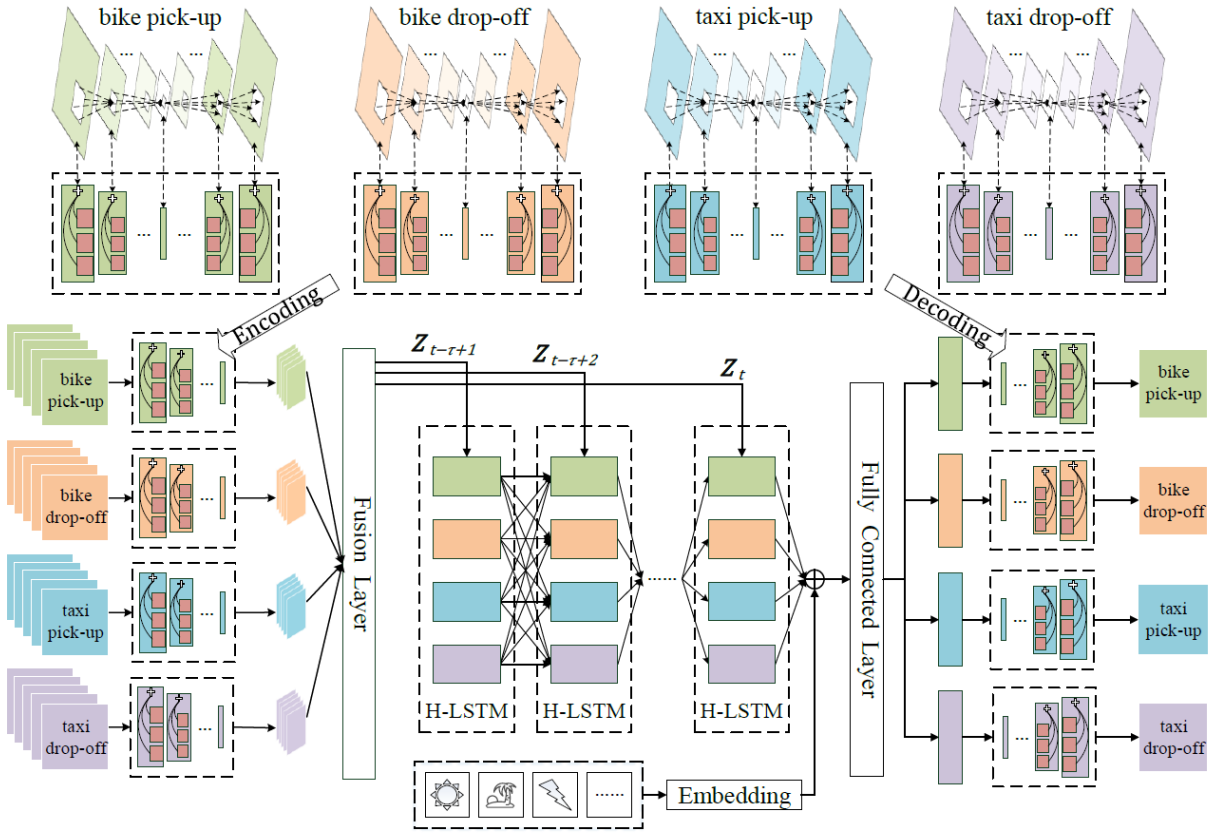

Figure 2: Overview of Co-prediction method based on Spatio-Temporal neural Network (CoST-Net). Four colors, Green, orange, blue, and purple, represent four kinds of transportation demands respectively. We employ the deep convolutional AutoEncoder to decompose the spatial demand into the combination of hidden demand bases, which is also used to encode a snapshot demand into a hidden state. Heterogeneous LSTM is used to mix the hidden states of multiple transportation demands. Additionally, the external factors are incorporated with the unifying hidden state of CoST-Net. A fully connected neural network is used to predict the next-period transportation demands together according to the fused information.

treat the entire demand in the city as an image. Then the problem has been transformed into the field at which CNN has shown excellent performance. Given a sequence of demand, we feed every demand graph to the deep convolutional AutoEncoder. The pick-up and drop-off demands of different transport have different spatial demand bases. In order not to hurt the performance of capturing high-level representation, the demand graphs are encoded separately. The encoder consists of K layers of CNNs which take its predecessor's output as input. The decoder has the symmetrical structure of the encoder and the CNN layers are replaced by Transposed CNN. The CNN layers and Transposed CNN layers are defined as:

$$Z_t^{L,r} = \delta_{en}\left(Z_t^{L,r-1} * W_{en}^{r-1} + b_{en}^{r-1}\right), \tag{8}$$

$$Z_t^{L,r-1} = \delta_{de}\left(Z_t^{L,r} * W_{de}^{r} + b_{de}^{r}\right). \tag{9}$$

If $r = 1$, $Z_t^{L,0} = X_t^L$, then Equation (8) and (9) reduce to Equation (6) and (7), where r is the number of layers. Equation (8) describes the CNN, where $Z_t^{r,L}$ is the output of r^{th} layer of CNN. W_X^{r-1} and b_X^{r-1} are two trainable parameters. Equation (9) represents the r^{th} layer of Transpose CNN. For each time interval t, we treat the entire city as a channel $I \times J$ map, $x_t^{B,p,L} \in \mathbb{R}^{1\times I\times J}$($x_t^{B,d,L}$, $x_t^{C,p,L}$, etc.). In this paper, we have not used any padding for images. The size of images will be reduced by CNN, and the feature graph Z_t^L is the high-level representation of X_t^L. The reduced size will be restored by Transposed CNN. After encoding, we get the map with shape $I' \times J'$. Then we feed all encoded results into LSTM.

3.2 Heterogeneous LSTM Neural Network

After the training of deep convolutional AutoEncoder is completed, we freeze the parameters of the encoder and decoder. Given a sequence of demand graphs, we encode the sequence to obtain high-level representations by the pre-trained encoders. Because each feature graph can be viewed as the spatial demand bases' combination, the feature map sequence can be viewed as coefficients' changing of spatial bases. The problem is transformed as predicting

the coefficient of spatial demand bases in time interval t+1, given historical coefficient in time interval $t-\tau+1, t-\tau+2, ..., t$. It can be solved by RNN. The basic RNN is defined as:

$$\boldsymbol{s}_t = \boldsymbol{W}_s[\boldsymbol{s}_{t-1}, \boldsymbol{Z}_t] + \boldsymbol{b}_s. \tag{10}$$

It updates the hidden state $\boldsymbol{s}_t$ by combining the current input $\boldsymbol{Z}_t$ and the previous state $\boldsymbol{s}_{t-1}$. $\boldsymbol{W}_s$ and $\boldsymbol{b}_s$ represent the weight and the bias. The RNN we employed should have plural hidden states, which can be defined as:

$$\begin{aligned}
\boldsymbol{s}_t^{B,p} &= f(\boldsymbol{W}_*^{B,p}[\boldsymbol{Z}_t, \boldsymbol{s}_{t-1}] + \boldsymbol{b}_*^{B,p}) \\
\boldsymbol{s}_t^{B,d} &= f(\boldsymbol{W}_*^{B,d}[\boldsymbol{Z}_t, \boldsymbol{s}_{t-1}] + \boldsymbol{b}_*^{B,d}) \\
\boldsymbol{s}_t^{C,p} &= f(\boldsymbol{W}_*^{C,p}[\boldsymbol{Z}_t, \boldsymbol{s}_{t-1}] + \boldsymbol{b}_*^{C,p}) \\
\boldsymbol{s}_t^{C,d} &= f(\boldsymbol{W}_*^{C,d}[\boldsymbol{Z}_t, \boldsymbol{s}_{t-1}] + \boldsymbol{b}_*^{C,d}),
\end{aligned} \tag{11}$$

where $\boldsymbol{s}_t^{B,p}, \boldsymbol{s}_t^{B,d}, \boldsymbol{s}_t^{C,p}, \boldsymbol{s}_t^{C,d}$ represent the hidden states of bike pick-up, bike drop-off, taxi pick-up, taxi drop-off and the f denotes all functions to update the hidden states. The $\boldsymbol{Z}_t$ can be divided into separate inputs, which means we denote $\boldsymbol{Z}_t$ as:

$$\boldsymbol{Z}_t = [\boldsymbol{Z}_t^{B,p}, \boldsymbol{Z}_t^{B,d}, \boldsymbol{Z}_t^{C,d}, \boldsymbol{Z}_t^{C,p}]. \tag{12}$$

According to the Equation (11) and (12), the four kinds of hidden states can be defined as:

$$\begin{bmatrix} \boldsymbol{s}_t^{B,p} \\ \boldsymbol{s}_t^{B,d} \\ \boldsymbol{s}_t^{C,p} \\ \boldsymbol{s}_t^{C,d} \end{bmatrix} = f\Bigg(\begin{bmatrix} \boldsymbol{W}_*^{B,p} \\ \boldsymbol{W}_*^{B,d} \\ \boldsymbol{W}_*^{C,p} \\ \boldsymbol{W}_*^{C,d} \end{bmatrix} \begin{bmatrix} \boldsymbol{Z}_t^{B,p} \\ \boldsymbol{Z}_t^{B,d} \\ \boldsymbol{Z}_t^{C,p} \\ \boldsymbol{Z}_t^{C,d} \\ \boldsymbol{s}_{t-1}^{B,p} \\ \boldsymbol{s}_{t-1}^{B,d} \\ \boldsymbol{s}_{t-1}^{C,p} \\ \boldsymbol{s}_{t-1}^{C,d} \end{bmatrix} + \begin{bmatrix} \boldsymbol{b}_*^{B,p} \\ \boldsymbol{b}_*^{B,d} \\ \boldsymbol{b}_*^{C,p} \\ \boldsymbol{b}_*^{C,d} \end{bmatrix} \Bigg). \tag{13}$$

In time series prediction problem, basic RNN is gradually abandoned due to the exploding and vanishing gradient. LSTM [4] is proposed to overcome these problems [3]. Because of its ability to learn long and short dependencies of temporal dynamics, it has been widely used and researched in recent years. The basic LSTM takes a temporal sequence $\{\boldsymbol{Z}_{t-\tau+1}, \boldsymbol{Z}_{t-\tau+2}, ..., \boldsymbol{Z}_t\}$ as input. The basic LSTM does not work when it comes to the prediction of t+1 coefficient with heterogeneous temporal sequences. Therefore, we propose a novel heterogeneous LSTM model. According to Equation (13), The mathematical expressions of heterogeneous LSTM are defined as follows:

$$\begin{aligned}
\boldsymbol{a}_t &= \sigma(\boldsymbol{W}_{\boldsymbol{a}}[\boldsymbol{Z}_t^{B,p}, \boldsymbol{Z}_t^{B,d}, \boldsymbol{Z}_t^{C,p}, \boldsymbol{Z}_t^{C,d}, \boldsymbol{s}_{t-1}] + \boldsymbol{b}_{\boldsymbol{a}}) \\
\boldsymbol{\phi}_t &= \sigma(\boldsymbol{W}_{\boldsymbol{\phi}}[\boldsymbol{Z}_t^{B,p}, \boldsymbol{Z}_t^{B,d}, \boldsymbol{Z}_t^{C,p}, \boldsymbol{Z}_t^{C,d}, \boldsymbol{s}_{t-1}] + \boldsymbol{b}_{\boldsymbol{\phi}}) \\
\boldsymbol{u}_t &= \sigma(\boldsymbol{W}_{\boldsymbol{u}}[\boldsymbol{Z}_t^{B,p}, \boldsymbol{Z}_t^{B,d}, \boldsymbol{Z}_t^{C,p}, \boldsymbol{Z}_t^{C,d}, \boldsymbol{s}_{t-1}] + \boldsymbol{b}_{\boldsymbol{u}}) \\
\tilde{\boldsymbol{v}}_t &= tanh(\boldsymbol{W}_{\boldsymbol{v}}[\boldsymbol{Z}_t^{B,p}, \boldsymbol{Z}_t^{B,d}, \boldsymbol{Z}_t^{C,p}, \boldsymbol{Z}_t^{C,d}, \boldsymbol{s}_{t-1}] + \boldsymbol{b}_{\boldsymbol{v}}) \\
\boldsymbol{v}_t &= \boldsymbol{f}_t * \boldsymbol{v}_{t-1} + \boldsymbol{a}_t * \tilde{\boldsymbol{v}}_t \\
\boldsymbol{s}_t &= \boldsymbol{u}_t * tanh(\boldsymbol{v}_t),
\end{aligned} \tag{14}$$

where LSTM takes inputs of $\boldsymbol{Z}_t^{B,p}, \boldsymbol{Z}_t^{B,d}, \boldsymbol{Z}_t^{C,p}, \boldsymbol{Z}_t^{C,d}, \boldsymbol{s}_{t-1}, \boldsymbol{v}_{t-1}$. The $\boldsymbol{a}_t$ denotes the input gate in this paper. Forget gate $\boldsymbol{\phi}_t$ can help previous cell state $\boldsymbol{v}_{t-1}$ forget some information. And $\boldsymbol{u}_t$ controls the output of the LSTM network in time interval t. $*$ denotes Hadamard product. In the heterogeneous LSTM, we can infer that predicting $\boldsymbol{s}_t^{B,p}$ can be directly affected by $\boldsymbol{Z}_t^{C,p}, \boldsymbol{Z}_t^{B,d}, \boldsymbol{Z}_t^{C,d}$ and vise versa, which indicates that the heterogeneous sequences will be fully interacted to make a mutual enhancement.

3.3 Fusion and Co-Prediction

Transportation demand is affected easily by many external factors, and it is difficult to give an appropriate explanation for how demands change. Drawing external factors into our prediction model is necessary. The usage of bike decreases obviously with a sudden rain, and demands in weekend are totally different from the demands in weekday. In this model, we mainly consider weather and event. The external factors data are enumeration type and numeric type, and the enumeration type will be encoded by one-hot. After encoding, we use a fully connected layer to extract the correlation and representation over the external factors.

The output of heterogeneous LSTM will be concatenated with fully connected layer, which is defined as:

$$\mathbf{Y}'_{t+1} = \varphi(\boldsymbol{s}_t \oplus \boldsymbol{e}'_t), \tag{15}$$

where $\mathbf{Y}'_{t+1}$ is the concatenated result, $\boldsymbol{s}_t$ is the output of LSTM at time interval t, $\boldsymbol{e}'_t$ is external factors' representations, and φ is a fully connected layer. $\mathbf{Y}'_{t+1}$ can be divided into separate outputs:

$$\mathbf{Y}'_{t+1} = [Y'^{B,p}_{t+1}, Y'^{B,d}_{t+1}, Y'^{C,d}_{t+1}, Y'^{C,p}_{t+1}]. \tag{16}$$

At last, decoders are used to decode $\mathbf{Y}'_{t+1}$ to demand graph $\hat{\mathbf{Y}}_{t+1}$, which is the prediction result. The decoders are not trainable, neither. If we want to predict bike pick-up demand, the parameters of the decoder will be copied from the pre-trained convolutional AutoEncoder which uses bike pick-up demands as input and ground truth. It is defined as:

$$\begin{aligned}
\hat{Y}_{t+1}^{B,p} &= f_{de}^{B,p}(Y'^{B,p}_{t+1}) \\
\hat{Y}_{t+1}^{B,d} &= f_{de}^{B,d}(Y'^{B,d}_{t+1}) \\
\hat{Y}_{t+1}^{C,p} &= f_{de}^{C,p}(Y'^{C,p}_{t+1}) \\
\hat{Y}_{t+1}^{C,d} &= f_{de}^{C,d}(Y'^{C,d}_{t+1}),
\end{aligned} \tag{17}$$

where $\hat{Y}_{t+1}^{B,p}, \hat{Y}_{t+1}^{B,d}, \hat{Y}_{t+1}^{C,p}, \hat{Y}_{t+1}^{C,d}$ are the prediction values, and $f_{de}^{B,p}$, $f_{de}^{B,d}, f_{de}^{C,p}, f_{de}^{C,d}$ are the decoders. The model is trained by minimizing mean squared error between predicted demand and true demand:

$$\begin{aligned}
\mathcal{L}(\theta) = &\gamma^{B,p} \left\| Y_{t+1}^{B,p} - \hat{Y}_{t+1}^{B,p} \right\|^2 + \gamma^{B,d} \left\| Y_{t+1}^{B,d} - \hat{Y}_{t+1}^{B,d} \right\|^2 + \\
&\gamma^{C,p} \left\| Y_{t+1}^{C,p} - \hat{Y}_{t+1}^{C,p} \right\|^2 + \gamma^{C,d} \left\| Y_{t+1}^{C,d} - \hat{Y}_{t+1}^{C,d} \right\|^2,
\end{aligned} \tag{18}$$

where $\mathcal{L}$ is the loss function, and θ is all the trainable parameters. We use Adam (Kinga and Adam [6]) for optimization. And the experiments are implemented by Keras. However, the correlations between different transport are complex and it is hard to balance the $\gamma^{B,p}, \gamma^{B,d}, \gamma^{C,p}, \gamma^{C,d}$ when the model is being trained. We predict the 4 kinds of demands separately in practice.

4 EXPERIMENTS

To validate the efficiency of our method, in this section, we evaluate our method via detailed experiments on real-world datasets. We first describe the datasets used for the experiments and then list the baselines. Finally, we present experiment results in detail and have further discussions. Codes and datasets will be released.

4.1 Datasets

We conduct experiments on two real-world datasets collected from NYC OpenData. The two datasets contain order records of taxi and bike in NYC.

- **NYC Citi Bike**: NYC Bike Sharing System generates the Citi Bike orders and put them on Citi Bike official website. 3 million and 700 thousand transaction records are available from April 1st 2016 to June 30th 2016(91 days). This data set contains the following information: bike pick-up station, bike drop-off station, bike pick-up time, bike drop-off time, trip duration.
- **NYC Taxi**: NYC Taxi consists of about 35 million taxicab trip records in New York from April 1st 2016 to June 30th 2016. On average, there are about 380 thousand trip records generated every day. The records consist of the following information: pick-up time, drop-off time, pick-up longitude, pick-up latitude, drop-off longitude, drop-off latitude, trip distance, etc.
- **External Factors**: The meteorological data is collected from a weather monitoring website [1]. There are 28 attributes in the original data and we finally select 9 attributes. Temperature, humidity, weather condition are included. There are 27 holidays in 91 days.

4.2 Baselines

We compare our model with the following methods. For each method, we tune the key hyper-parameters and make sure that they have the best performance.

4.2.1 Classical machine learning methods. We choose some typical and classical machine learning methods as our baseline.

- **HA**: We predict the transportation demand by averaging historical demand values for a region in the same time interval. For example, the pick-up demands of bike in region(i, j) at 1:00pm-1:30pm on Friday are predicted by the average of historical demands at 1:00pm - 1:30pm of the same region.
- **ARIMA**: Auto-Regressive Integrated Moving Average is a well-known model for understanding and predicting time series.
- **LR**: Linear Regression is a classical method to model the relationship between variables.
- **SVR**: Support Vector Regression (SVR) is a regression version of Support-Vector Machine.
- **XGBoost**: XGBoost is a widely used method based on gradient boosting tree.

4.2.2 Neural-network-based methods. Classical deep learning methods and state-of-the-art methods are listed in this section.

- **Multiple layer perceptron (MLP)**: The MLP we employ contains four fully connected layers. The numbers of hidden units of layers are 128, 128, 64 and 64.
- **ConvLSTM[16]**: The ConvLSTM was proposed in 2015, and it enables capturing both temporal and spatial relationship simultaneously by combining CNN and LSTM.
- **DeepST[23]**: A deep neural network based DNN, which is used to deal with sptaio-temporal data. It has 4 parts, which focuses on different temporal dependencies and external factors.
- **ST-ResNet**: ST-ResNet is a state-of-the-art approach for transportation demand prediction. It captures spatial correlation with Deep Residual Network. Furthermore, the temporal correlation is divided into trend, period, and closeness information which are captured separately.

4.3 Evaluation Metric

We evaluate our algorithm by Rooted Mean Square Error (RMSE) and Pearson Correlation Coefficient (PCC) as follows.

$$RMSE = \sqrt{\frac{1}{\xi}\sum_{i=1}^{\xi}(Y_{t+1}^{i} - \hat{Y}_{t+1}^{i})^2},$$

$$PCC = \frac{1}{\xi}\sum_{i=1}^{\xi}\frac{\sum_{l=1}^{\epsilon}(\hat{Y}_{t+1}^{i,l} - \bar{\hat{Y}}_{t+1}^{i})(Y_{t+1}^{i,l} - \bar{Y}_{t+1}^{i})}{\sqrt{\sum_{l=1}^{\epsilon}(\hat{Y}_{t+1}^{i,l} - \bar{\hat{Y}}_{t+1}^{i})^2}\sqrt{\sum_{l=1}^{\epsilon}(Y_{t+1}^{i,l} - \bar{Y}_{t+1}^{i})^2}},$$

where the ξ represents the total number of samples. Y_{t+1}^{i} and $\hat{Y}_{t+1}^{i}$ are the ground truth and prediction value. In the evaluation of PCC, we transform every 2D map into 1D and $\epsilon = I * J$. $\bar{\hat{Y}}_{t+1}^{i}$ and $\bar{Y}_{t+1}^{i}$ are the average values of $\hat{Y}_{t+1}^{i}$ and Y_{t+1}^{i}.

4.4 Experimental Setting

The region we discuss is a rectangle, 14.4km×8.4km, which covers a part of Brooklyn, West New York, and Manhattan Island. We divide the region into 20×20, and per unit grid is a rectangle, not a square, which can protect the spatial correlation and help us conduct further extraction. The size of each grid is $0.72km \times 0.42km$. The time interval is set as half an hour. The time intervals for the data used to train heterogeneous LSTM are 0:00am to 0:30am, 0:30am to 1:00am, 1:00am to 1:30am, etc., which are in good order. The data in the last 2 weeks are used for testing, and the dataset for bike pick-up demand contains about 3300 demand graphs for training and about 670 graphs for testing. The dataset we use to train the deep convolutional AutoEncoder is different from the dataset to train heterogeneous LSTM. According to the previous work [7], the small size of the dataset will hurt the learning of high-level features for the autoencoder. We address the problem by scaling up the dataset. The dataset used to train the deep convolutional AutoEncoder is 8 times larger than the dataset used to train heterogeneous LSTM. When we prepared the data for the AutoEncoder, we cut with small shifts on the original data of heterogeneous LSTM many times. If we have the 10 minutes shift, the demand graphs in time intervals 0:10am to 0:40am, 0:40am to 1:10am, etc., will be added into the dataset which is used to train the convolutional AutoEncoder. Repeating the work and having different time shifts, we get a much

[1] https://www.wunderground.com/

bigger dataset than the dataset which is used to train heterogeneous LSTM. The method we use to generate the dataset for the deep convolutional AutoEncoder has two good points. On the one side, two datasets have the same original data, which ensure the accuracy and robustness of the model. On the other side, according to the state-of-the-art work [17], traffic data is not strictly periodic. There are some temporal shifting between different days and weeks. For example, the peak hour on weekdays may vary from 4:30*pm* to 6:00*pm*, which makes it difficult to catch the peak hour with a single time cut. In addition, the shifts in our dataset can be diverse, which help to capture the invariance of the spatial demand bases.

The demands for all locations were normalized to [0,1] by Max-Min normalization. In the convolutional AutoEncoder, the depth of CNN is 2. The kernel size of the filter is 3×3, and the numbers of filters are setbike to 16 and 14. The stride is set to 1. Because we predict the demands separately, the output of LSTM has the same dimension as the input of the decoder, and the dimension of it is set to 196, which can be reshaped to 14×14.

4.5 Results

4.5.1 Comparison with different Baselines. Table 2 shows the comparison results with baselines. We conduct experiments on NYC Citi Bike pick-up demands, NYC Citi Bike drop-off demands, NYC Taxi pick-up demands, and NYC Taxi drop-off demands. Each baseline was run 10 times and we compare the average results. All methods show a smaller RMSE in NYC Citi bike compared with NYC Taxi. This is because the NYC taxi has a larger demand than the NYC Citi bike. The model we propose achieves the best performance and has the lowest RMSE on all comparison experiments.

Table 2: Comparison of CoST-Net with Baselines in *RMSE*.

Method	NYC Citi Bike		NYC Taxi	
	Pick-up	Drop-off	Pick-up	Drop-off
HA	5.4801	5.4672	22.6326	21.5601
ARIMA	6.4427	4.9956	10.2461	9.1361
LR	8.1028	8.1767	34.6678	33.6617
SAR	3.7536	3.5412	15.4175	13.5590
XGBoost	2.7166	2.5306	9.8231	7.9314
MLP	2.9364	2.6273	13.1985	11.4133
ConvLSTM	2.8726	2.7443	10.7542	10.3004
DeepST	2.7358	2.5506	11.2670	10.1583
ST-ResNet	2.7776	2.6219	10.5112	9.1331
CoST-Net	**2.4819**	**2.2522**	**8.3853**	**7.4696**

Classical machine learning methods show a relatively poor performance than deep learning methods. LR and HA have the largest RMSE values on two datasets. ARIMA only extracts temporal dependencies. The worse performance may indicate the limitations of employing temporal features only. It seems that XGBoost has an advantage in this predicting problem for having a better performance than some deep learning methods.

In deep learning methods, MLP has a relatively poor performance, because MLP does not employ CNN to extract spatial correlation. CNN has shown a strong ability to handle this problem. DeepST, ST-ResNet, are state-of-the-art methods. The outcome of ST-ResNet is not stable, which makes it not as good as DeepST. When we predict NYC Citi Bike drop-off demands by ST-ResNet, there are some results whose RMSE are higher than 3, which will be further discussed in Section 4.5.2. ConvLSTM is a novel structure which combines the LSTM and CNN and captures the spatial and temporal correlation simultaneously, and it is suitable for spatio-temporal prediction. Our methods achieve 13.6%, 17.93%, 22.03%, and 27.48% lower RMSE than ConvLSTM in four kinds of experiments. We think the comparative methods do not extract deep correlation between demand graphs.

Table 3: Comparison of CoST-Net with Baselines in *PCC*.

Method	NYC Citi Bike		NYC Taxi	
	Pick-up	Drop-off	Pick-up	Drop-off
HA	0.6659	0.6684	0.8391	0.8208
ARIMA	0.1058	0.1050	0.0073	0.0122
LR	**0.8047**	0.7917	0.7917	0.7742
SAR	0.5331	0.5459	0.4119	0.4003
XGBoost	0.7985	0.7930	0.9549	0.9453
MLP	0.6871	0.5139	0.9522	0.9079
ConvLSTM	0.7397	0.7248	0.9412	0.9253
DeepST	0.7593	0.7560	0.9539	0.9430
ST-ResNet	0.7941	0.7901	0.9537	0.9494
CoST-Net	0.8040	**0.8027**	**0.9611**	**0.9535**

The results of comparing with baselines in PCC are shown in Table 3. HA, ARIMA, SAR have relatively poor performances. ARIMA is not suitable for this prediction problem because its results are close to 0.1, which indicates that there is little correlation between the forecast results of ARIMA and the ground truth. LR outperforms our method with a tiny advantage in bike pick-up demands prediction, which can not hide the superiority of our method. CoST-Net gets the highest PCC in the other three experiments.

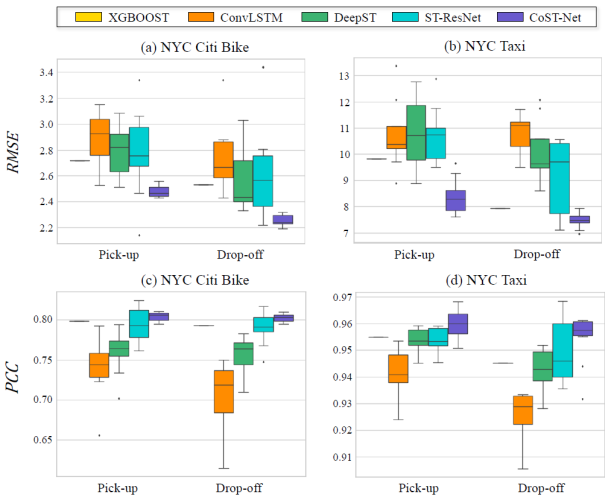

Figure 3: Comparison of CoST-Net with XGBoost, ConvLSTM, DeepST, and ST-ResNet.

4.5.2 Variance comparison. We draw box-plot with the experiment results of XGBoost, ConvLSTM, DeepST, ST-ResNet, and CoST-Net. Figure 3 shows the comparison in NYC Citi Bike and NYC Taxi. The box-plot was invented in 1977 and it displays the maximum, minimum, median, upper and lower quartiles of a dataset. Our method has the best performance in all methods and the smallest variance in the deep learning methods. For example, in bike pick-up demand prediction, ST-ResNet achieved the lowest RMSE in a single experiment. But most experiments of our model have RMSE lower than 2.5, which exceeds 75% results of ST-ResNet. What's more, ST-ResNet has an outlier with the RMSE higher than 3.3. The best result of XGBoost is poorer than the best of ST-ResNet, but it has a better performance than the average of ST-ResNet.

Table 4: Comparison with Variants of CoST-Net.

Method	NYC Citi Bike		NYC Taxi	
	Pick-up	Drop-off	Pick-up	Drop-off
A Channel	2.5709	2.4254	8.9663	8.3146
Two Channels	2.5654	2.3875	8.8858	7.9817
Three Channels	2.5199	2.3687	8.6943	7.8892
Four Channels	**2.4819**	**2.2522**	**8.3853**	**7.4696**

4.5.3 Comparison with variants of CoST-Net. Table 4 shows the comparison with the variants of CoST-Net. For example, if we attend to predict bike pick-up demand, the experiment with a channel means that we only employ bike pick-up demand data to predict it with our model. Predicting with two channels represents that conducting experiment uses bike pick-up demand and bike drop-off demand data. We add the taxi pick-up data into the prediction with three channels. Our methods contain four channels of input. In Table 4, the more channels we have employed, the better results we get. This is because we can extract more information with multiple inputs. In addition, the model with an input channel outperforms all the baselines except XGBoost, although it is the worst in the variants of our model.

4.5.4 Performance at Different Time Points. We illustrate the superiority of our method under different circumstances. As it has shown in Figure 4, we evaluate our methods with some baselines at 4 specific time points, which are selected as 0:00, 5:00, 10:00, 16:00 in weekday. We have counted the total volume of four kinds of demands in each time interval and choose the special time points. Because of space limitation, we haven't shown the demands' activities figure in this paper. XGBoost has a good performance in Table 2, but it has the highest RMSE in predicting NYC Citi Bike demands at 0:00*am*. All methods perform worse in predicting NYC Citi Bike demands at 16:00 compared to predicting the demands at 5:00. This is because there are relatively stable demands of bike from 0:00 to 5:00. Even so, our method achieves the lowest RMSE.

5 RELATED WORK

There are some previous works on the prediction of traffic flow. Most of them attend to predict the traffic volume for a set of locations at a specific timestamp. The early researches mainly focused on using data mining techniques and empirical statistical methods. Autoregressive Integrated Moving Average (ARIMA) and its

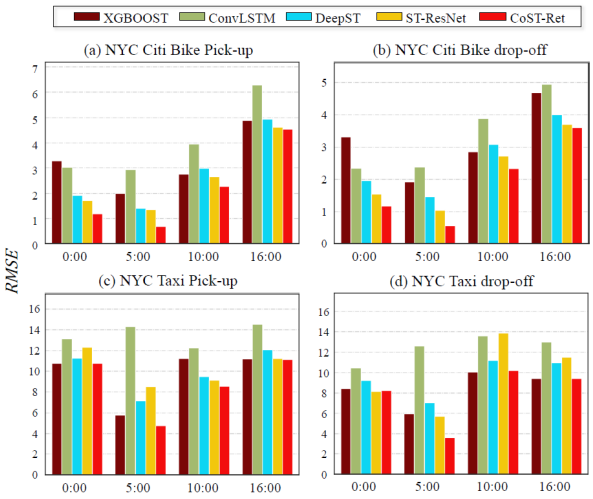

Figure 4: Comparison of CoST-Net with XGBoost, ConvLSTM, DeepST, and ST-ResNet at different time points.

variants are also very popular in classical methods [12]. Liu et al. [9] proposed a spatial demand prediction model by comparing the correlation with history demand and weather. Liu et al. [10] made an improvement when it came to predicting the bicycle drop-off demands by estimating the trip duration by a 2-peak Gaussian function. Some recent studies began to model spatial information and smooth the differences between nearby regions and timestamps by regularization [13]. Ye et al. [19] proposed two efficient methods for multi-user mobile sequential recommendation problem. However, classical prediction methods mostly aim at specific regions and it is hard for them to model spatio-temporal correlation simultaneously.

Deep learning methods provide a novel way to capture nonlinear relations. Lv et al. [11], using a SAE(stacking autoencoders) model, first introduced deep learning methods into transportation demand prediction problem. At the same time, CNN and LSTM were developing rapidly and they have been widely applied in computer vision and natural language processing. Zhang et al. [22] proposed a method which treats the whole city's demands at a timestamp as an image, then they applied Residual Network to capture the spatial correlation between different regions. Cui et al. [1] and Wei et al. [14] both applied LSTM to capture sequential dependency. However, those methods either use hand designed features to model temporal sequential dependency or do not consider the spatial relation, and none of them capture both correlations simultaneously.

Recently, Yao et al. [18] proposed a multi-view spatial-temporal network which predicts the taxi demands from three different views: spatial view with local CNN, temporal view with LSTM, and semantic view with structural embedding. Jin et al. [5] improved previous ST-ResNet by adding an LSTM structure. Convolutional LSTM (ConvLSTM) is a novel deep neural network which was proposed in 2015 [16]. Several works used it to capture temporal and spatial dependency simultaneously [15]. Zonoozi et al. [25] employed a convolutional GRU to predict crowd density. There are some prediction works with spatio-temporal data in other scenarios. Yu et al. [21] used the combination of CNN and LSTM to predict the

road traffic speed, and similar methods were applied in air quality prediction [20]. However, they did not consider the correlation between different transport and high-level representations of demand graphs. Zhou et al. [24] gives us a new view to handle the problem. They clustered all demand graphs into K clusters and select K representative demand graphs. The novel method predicts the demand graph at timestamp t+1 by combining the K demand graphs. The K representative demand graphs contain a large amount of information. Representation learning is a hot research topic in data mining and computer vision. In classical machine learning and statistical methods, PCA(Principal component analysis), ICA(Independent Component Analysis), clustering, all of them can be classified into this field. Hinton and Salakhutdinov [2] compared representation learning method in deep learning (autoencoder) with PCA. Le [7] built a huge network and employed a large amount of data to capture high-level representations, such as human face, human body. Liu et al. [8] decomposed the medical event sequences into graph bases so that all medical event sequences can be represented as the combination of graph bases. We propose a deep convolutional AutoEncoder to capture the spatial demand bases in the demand graphs. What's more, we extract the correlation between multiple transports and take external factors into account.

6 CONCLUSION

This paper provides a novel transportation demand prediction method, namely, Co-prediction based on Spatio-Temporal neural Network (CoST-Net). It consists of three modules: 1) a representation learning module, which encodes a spatial demand distribution into a combination of hidden spatial demand bases; 2) a heterogeneous learning module, which fuses the states of multiple transportation demands, and models the dynamics of these states; 3) an integrating module that incorporates the environmental factors, and presents the prediction of multiple transportation demands. Benefited from spatial decomposition representation and heterogeneous fusion of multiple transportation demands, CoST-Net is able to outperform the existing transportation demand prediction methods. To test the proposed method, experiments have been conducted on real-world taxi and sharing bike data, results demonstrate the effectiveness of CoST-Net in terms of both prediction accuracy and robustness. This research provides new insights to the study of transportation demand prediction from both micro and macro perspectives. In future, we will study how to enrich this research by taking bus and railway data into account.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their constructive comments on this research work. This work is supported by the National Natural Science Foundation of China under Grant No. 51822802, 51778033, U1811463, and 71531001, the Science and Technology Major Project of Beijing under Grant No. Z171100005117001.

REFERENCES

[1] Zhiyong Cui, Ruimin Ke, and Yinhai Wang. 2016. Deep Stacked Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-wide Traffic Speed Prediction. In *6th International Workshop on Urban Computing (UrbComp 2017)*.

[2] Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *science* 313, 5786 (2006), 504–507.

[3] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, and others. 2001. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. (2001).

[4] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[5] Wenwei Jin, Youfang Lin, Zhihao Wu, and Huaiyu Wan. 2018. Spatio-Temporal Recurrent Convolutional Networks for Citywide Short-term Crowd Flows Prediction. In *Proceedings of the 2nd International Conference on Compute and Data Analysis*. ACM, 28–35.

[6] D Kinga and J Ba Adam. 2015. A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, Vol. 5.

[7] Quoc V Le. 2013. Building high-level features using large scale unsupervised learning. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 8595–8598.

[8] Chuanren Liu, Fei Wang, Jianying Hu, and Hui Xiong. 2015. Temporal phenotyping from longitudinal electronic health records: A graph based framework. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 705–714.

[9] Junming Liu, Qiao Li, Meng Qu, Weiwei Chen, Jingyuan Yang, Hui Xiong, Hao Zhong, and Yanjie Fu. 2015. Station site optimization in bike sharing systems. In *Data Mining (ICDM), 2015 IEEE International Conference on*. IEEE, 883–888.

[10] Junming Liu, Leilei Sun, Weiwei Chen, and Hui Xiong. 2016. Rebalancing bike sharing systems: A multi-source data smart optimization. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1005–1014.

[11] Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, Fei-Yue Wang, and others. 2015. Traffic flow prediction with big data: A deep learning approach. *IEEE Trans. Intelligent Transportation Systems* 16, 2 (2015), 865–873.

[12] Luis Moreira-Matias, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. 2013. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems* 14, 3 (2013), 1393–1402.

[13] Yongxin Tong, Yuqiang Chen, Zimu Zhou, Lei Chen, Jie Wang, Qiang Yang, Jieping Ye, and Weifeng Lv. 2017. The simpler the better: a unified approach to predicting original taxi demands based on large-scale online platforms. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1653–1662.

[14] Hua Wei, Yuandong Wang, Tianyu Wo, Yaxiao Liu, and Jie Xu. 2016. Zest: a hybrid model on predicting passenger demand for chauffeured car service. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 2203–2208.

[15] Hong Wei, Hao Zhou, Jangan Sankaranarayanan, Sudipta Sengupta, and Hanan Samet. 2018. Residual Convolutional LSTM for Tweet Count Prediction. In *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 1309–1316.

[16] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*. 802–810.

[17] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, Yanwei Yu, and Zhenhui Li. 2018. Modeling Spatial-Temporal Dynamics for Traffic Prediction. *arXiv preprint arXiv:1803.01254* (2018).

[18] Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Zhenhui Li. 2018. Deep multi-view spatial-temporal network for taxi demand prediction. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[19] Zeyang Ye, Lihao Zhang, Keli Xiao, Wenjun Zhou, Yong Ge, and Yuefan Deng. 2018. Multi-User Mobile Sequential Recommendation: An Efficient Parallel Computing Paradigm. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2624–2633.

[20] Xiuwen Yi, Junbo Zhang, Zhaoyuan Wang, Tianrui Li, and Yu Zheng. 2018. Deep Distributed Fusion Network for Air Quality Prediction. (2018).

[21] Haiyang Yu, Zhihai Wu, Shuqin Wang, Yunpeng Wang, and Xiaolei Ma. 2017. Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. *Sensors* 17, 7 (2017), 1501.

[22] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction.. In *AAAI*. 1655–1661.

[23] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, and Xiuwen Yi. 2016. DNN-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 92.

[24] Xian Zhou, Yanyan Shen, Yanmin Zhu, and Linpeng Huang. 2018. Predicting multi-step citywide passenger demands using attention-based neural networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 736–744.

[25] Ali Zonoozi, Jung-jae Kim, Xiao-Li Li, and Gao Cong. 2018. Periodic-CRN: A Convolutional Recurrent Model for Crowd Density Prediction with Recurring Periodic Patterns.. In *IJCAI*. 3732–3738.