Deep Multi-View Spatial-Temporal Network for Taxi Demand Prediction

Huaxiu Yao*, **Fei Wu**
Pennsylvania State University
{huaxiuyao, fxw133}@ist.psu.edu

Jintao Ke*
Hong Kong University of Science and Technology
jke@connect.ust.hk

Xianfeng Tang
Pennsylvania State University
xianfeng@ist.psu.edu

Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye
Didi Chuxing
{jiayitian, lusiyu, gongpinghua, yejieping}@didichuxing.com

Zhenhui Li
Pennsylvania State University
jessieli@ist.psu.edu

Abstract

Taxi demand prediction is an important building block to enabling intelligent transportation systems in a smart city. An accurate prediction model can help the city pre-allocate resources to meet travel demand and to reduce empty taxis on streets which waste energy and worsen the traffic congestion. With the increasing popularity of taxi requesting services such as Uber and Didi Chuxing (in China), we are able to collect large-scale taxi demand data continuously. How to utilize such big data to improve the demand prediction is an interesting and critical real-world problem. Traditional demand prediction methods mostly rely on time series forecasting techniques, which fail to model the complex non-linear spatial and temporal relations. Recent advances in deep learning have shown superior performance on traditionally challenging tasks such as image classification by learning the complex features and correlations from large-scale data. This breakthrough has inspired researchers to explore deep learning techniques on traffic prediction problems. However, existing methods on traffic prediction have only considered spatial relation (e.g., using CNN) or temporal relation (e.g., using LSTM) independently. We propose a Deep Multi-View Spatial-Temporal Network (DMVST-Net) framework to model both spatial and temporal relations. Specifically, our proposed model consists of three views: temporal view (modeling correlations between future demand values with near time points via LSTM), spatial view (modeling local spatial correlation via local CNN), and semantic view (modeling correlations among regions sharing similar temporal patterns). Experiments on large-scale real taxi demand data demonstrate effectiveness of our approach over state-of-the-art methods.

Introduction

Traffic is the pulse of a city that impacts the daily life of millions of people. One of the most fundamental questions for future smart cities is how to build an efficient transportation system. To address this question, a critical component is an accurate demand prediction model. The better we can predict demand on travel, the better we can pre-allocate resources to meet the demand and avoid unnecessary energy consumption. Currently, with the increasing popularity of taxi requesting services such as Uber and Didi Chuxing, we are able to collect massive demand data at an unprecedented scale. The question of how to utilize big data to better predict traffic demand has drawn increasing attention in AI research communities.

In this paper, we study the taxi demand prediction problem; that problem being how to predict the number of taxi requests for a region in a future timestamp by using historical taxi requesting data. In literature, there has been a long line of studies in traffic data prediction, including traffic volume, taxi pick-ups, and traffic in/out flow volume. To predict traffic, time series prediction methods have frequently been used. Representatively, autoregressive integrated moving average (ARIMA) and its variants have been widely applied for traffic prediction (Li et al. 2012; Moreira-Matias et al. 2013; Shekhar and Williams 2008). Based on the time series prediction method, recent studies further consider spatial relations (Deng et al. 2016; Tong et al. 2017) and external context data (e.g., venue, weather, and events) (Pan, Demiryurek, and Shahabi 2012; Wu, Wang, and Li 2016). While these studies show that prediction can be improved by considering various additional factors, they still fail to capture the complex nonlinear spatial-temporal correlations.

Recent advances in deep learning have enabled researchers to model the complex nonlinear relationships and have shown promising results in computer vision and natural language processing fields (LeCun, Bengio, and Hinton 2015). This success has inspired several attempts to use deep learning techniques on traffic prediction problems. Recent studies (Zhang, Zheng, and Qi 2017; Zhang et al. 2016) propose to treat the traffic in a city as an image and the traffic volume for a time period as pixel values. Given a set of historical traffic images, the model predicts the traffic image for the next timestamp. Convolutional neural network (CNN) is applied to model the complex spatial correlation. Yu et al. (2017) proposes to use Long Short Term Memory networks (LSTM) to predict loop sensor readings. They show the proposed LSTM model is capable of modeling complex sequential interactions. These pioneering attempts show superior performance compared with previous methods based on traditional time series prediction methods. However, none of them consider spatial relation and tempo-

*The paper was done when these authors were interns in Didi Chuxing.

ral sequential relation simultaneously.

In this paper, we harness the power of CNN and LSTM in a joint model that captures the complex nonlinear relations of both space and time. However, we cannot simply apply CNN and LSTM on demand prediction problem. If treating the demand over an entire city as an image and applying CNN on this image, we fail to achieve the best result. We realize including regions with weak correlations to predict a target region actually hurts the performance. To address this issue, we propose a novel local CNN method which only considers spatially nearby regions. This local CNN method is motivated by the First Law of Geography: "near things are more related than distant things," (Tobler 1970) and it is also supported by observations from real data that demand patterns are more correlated for spatially close regions.

While local CNN method filters weakly correlated remote regions, this fails to consider the case that two locations could be spatially distant but are similar in their demand patterns (i.e., on the semantic space). For example, residential areas may have high demands in the morning when people transit to work, and commercial areas may be have high demands on weekends. We propose to use a graph of regions to capture this latent semantic, where the edge represents similarity of demand patterns for a pair of regions. Later, regions are encoded into vectors via a graph embedding method and such vectors are used as context features in the model. In the end, a fully connected neural network component is used for prediction.

Our method is validated via large-scale real-world taxi demand data from Didi Chuxing. The dataset contains taxi demand requests through Didi service in the city of Guangzhou in China over a two-month span, with about 300,000 requests per day on average. We conducted extensive experiments to compare with state-of-the-art methods and have demonstrated the superior performance of our proposed method.

In summary, our contributions are summarized as follow:

- We proposed a unified multi-view model that jointly considers the spatial, temporal, and semantic relations.
- We proposed a local CNN model that captures local characteristics of regions in relation to their neighbors.
- We constructed a region graph based on the similarity of demand patterns in order to model the correlated but spatially distant regions. The latent semantics of regions are learnt through graph embedding.
- We conducted extensive experiments on a large-scale taxi request dataset from Didi Chuxing. The results show that our method consistently outperforms the competing baselines.

Related Work

Problems of traffic prediction could include predicting any traffic related data, such as traffic volume (collected from GPS or loop sensors), taxi pick-ups or drop-offs, traffic flow, and taxi demand (our problem). The problem formulation process for these different types of traffic data is the same. Essentially, the aim is to predict a traffic-related value for a location at a timestamp. In this section, we will discuss the related work on traffic prediction problems.

The traditional approach is to use time series prediction method. Representatively, autoregressive integrated moving average (ARIMA) and its variants have been widely used in traffic prediction problem (Shekhar and Williams 2008; Li et al. 2012; Moreira-Matias et al. 2013).

Recent studies further explore the utilities of external context data, such as venue types, weather conditions, and event information (Pan, Demiryurek, and Shahabi 2012; Wu, Wang, and Li 2016; Wang et al. 2017; Tong et al. 2017). In addition, various techniques have also been introduced to model spatial interactions. For example, Deng et al. (2016) used matrix factorization on road networks to capture a correlation among road connected regions for predicting traffic volume. Several studies (Tong et al. 2017; Idé and Sugiyama 2011; Zheng and Ni 2013) also propose to smooth the prediction differences for nearby locations and time points via regularization for close space and time dependency. These studies assume traffic in nearby locations should be similar. However, all of these methods are based on the time series prediction methods and fail to model the complex nonlinear relations of the space and time.

Recently, the success of deep learning in the fields of computer vision and natural language processing (LeCun, Bengio, and Hinton 2015; Krizhevsky, Sutskever, and Hinton 2012) motivates researchers to apply deep learning techniques on traffic prediction problems. For instance, Wang et al. (2017) designed a neural network framework using context data from multiple sources and predict the gap between taxi supply and demand. The method uses extensive features, but does not model the spatial and temporal interactions.

A line of studies applied CNN to capture spatial correlation by treating the entire city's traffic as images. For example, Ma et al. (2017) utilized CNN on images of traffic speed for the speed prediction problem. Zhang et al. (2016) and Zhang, Zheng, and Qi (2017) proposed to use residual CNN on the images of traffic flow. These methods simply use CNN on the whole city and will use all the regions for prediction. We observe that utilizing irrelevant regions (e.g., remote regions) for prediction of the target region might actually hurts the performance. In addition, while these methods do use traffic images of historical timestamps for prediction, but they do not explicitly model the temporal sequential dependency.

Another line of studies uses LSTM for modeling sequential dependency. Yu et al. (2017) proposed to apply Long-short-term memory (LSTM) network and autoencoder to capture the sequential dependency for predicting the traffic under extreme conditions, particularly for peak-hour and post-accident scenarios. However, they do not consider the spatial relation.

In summary, the biggest difference of our proposed method compared with literature is that we consider *both* spatial relation and temporal sequential relation in a joint deep learning model.

Preliminaries

In this section, we first fix some notations and define the taxi demand problem. We follow previous studies (Zhang, Zheng, and Qi 2017; Wang et al. 2017) and define the set of non-overlapping locations $L = \{l_1, l_2, ..., l_i, ..., l_N\}$ as rectangle partitions of a city, and the set of time intervals as $\mathcal{I} = \{I_0, I_1, ..., I_t, ..., I_T\}$. 30 minutes is set as the length of the time interval. Alternatively, more sophisticated ways of partitioning can also be used, such as partition space by road network (Deng et al. 2016) or hexagonal partitioning. However, this is not the focus of this paper, and our methodology can still be applied. Given the set of locations L and time intervals T, we further define the following.

Taxi request: A taxi request o is defined as a tuple $(o.t, o.l, o.u)$, where $o.t$ is the timestamp, $o.l$ represents the location, and $o.u$ is user identification number. The requester identifications are used for filtering duplicated and spammer requests.

Demand: The demand is defined as the number of taxi requests at one location per time point, i.e., $y_t^i = |\{o : o.t \in I_t \wedge o.l \in l_i\}|$, where $|\cdot|$ denotes the cardinality of the set. For simplicity, we use the index of time intervals t representing I_t, and the index of locations i representing l_i for rest of the paper.

Demand prediction problem: The demand prediction problem aims to predict the demand at time interval $t+1$, given the data until time interval t. In addition to historical demand data, we can also incorporate context features such as temporal features, spatial features, meteorological features (refer to Data Description section for more details). We denote those context features for a location i and a time point t as a vector $\mathbf{e}_t^i \in \mathbb{R}^r$, where r is the number of features. Therefore, our final goal is to predict

$$y_{t+1}^i = \mathcal{F}(\mathcal{Y}_{t-h,...,t}^L, \mathcal{E}_{t-h,...,t}^L)$$

for $i \in L$, where $\mathcal{Y}_{t-h,...,t}^L$ are historical demands and $\mathcal{E}_{t-h,...,t}^L$ are context features for all locations L for time intervals from $t-h$ to t, where $t-h$ denotes the starting time interval. We define our prediction function $\mathcal{F}(\cdot)$ on all regions and previous time intervals up to $t-h$ to capture the complex spatial and temporal interaction among them.

Proposed DMVST-Net Framework

In this section, we provide details for our proposed Deep Multi-View Spatial-Temporal Network (DMVST-Net) framework, i.e., our prediction function $\mathcal{F}$. Figure 1 shows the architecture of our proposed method. Our proposed model has three views: spatial, temporal, and semantic view.

Spatial View: Local CNN

As we mentioned earlier, including regions with weak correlations to predict a target region actually hurts the performance. To address this issue, we propose a local CNN method which only considers spatially nearby regions. Our intuition is motivated by the First Law of Geography (Tobler 1970) - "near things are more related than distant things".

As shown in Figure 1(a), at each time interval t, we treat one location i with its surrounding neighborhood as one $S \times S$ image (e.g., 7×7 image in Figure 1(a)) having one channel of demand values (with i being at the center of the image), where the size S controls the spatial granularity. We use zero padding for location at boundaries of the city. As a result, we have an image as a tensor (having one channel) $\mathbf{Y}_t^i \in \mathbb{R}^{S \times S \times 1}$, for each location i and time interval t. The local CNN takes $\mathbf{Y}_t^i$ as input $\mathbf{Y}_t^{i,0}$ and feeds it into K convolutional layers. The transformation at each layer k is defined as follows:

$$\mathbf{Y}_t^{i,k} = f(\mathbf{Y}_t^{i,k-1} * \mathbf{W}_t^k + \mathbf{b}_t^k), \tag{1}$$

where $*$ denotes the convolutional operation and $f(\cdot)$ is an activation function. In this paper, we use the rectifier function as the activation, i.e., $f(z) = max(0, z)$. $\mathbf{W}_t^k$ and $\mathbf{b}_t^k$ are two sets of parameters in the k^{th} convolution layer. Note that the parameters $\mathbf{W}_t^{1,...,K}$ and $\mathbf{b}_t^{1,...,K}$ are shared across all regions $i \in L$ to make the computation tractable.

After K convolution layers, we use a flatten layer to transform the output $\mathbf{Y}_t^{i,K} \in \mathbb{R}^{S \times S \times \lambda}$ to a feature vector $\mathbf{s}_t^i \in \mathbb{R}^{S^2 \lambda}$ for region i and time interval t. At last, we use a fully connected layer to reduce the dimension of spatial representations $\mathbf{s}_t^i$, which is defined as:

$$\hat{\mathbf{s}}_t^i = f(W_t^{fc} \mathbf{s}_t^i + b_t^{fc}), \tag{2}$$

where W_t^{fc} and b_t^{fc} are two learnable parameter sets at time interval t. Finally, for each time interval t, we get the $\hat{\mathbf{s}}_t^i \in \mathbb{R}^d$ as the representation for region i.

Temporal View: LSTM

The temporal view models sequential relations in the demand time series. We propose to use Long Short-Term Memory (LSTM) network as our temporal view component. LSTM (Hochreiter and Schmidhuber 1997) is a type of neural network structure, which provides a good way to model sequential dependencies by recursively applying a transition function to the hidden state vector of the input. It is proposed to address the problem of classic Recurrent Neural Network (RNN) for its exploding or vanishing of gradient in the long sequence training (Hochreiter et al. 2001).

LSTM learns sequential correlations stably by maintaining a *memory cell* $\mathbf{c}_t$ in time interval t, which can be regarded as an accumulation of previous sequential information. In each time interval, LSTM takes an input $\mathbf{g}_t^i$, $\mathbf{h}_{t-1}$ and $\mathbf{c}_{t-1}$ in this work, and then all information is accumulated to the memory cell when the *input gate* $\mathbf{i}_t^i$ is activated. In addition, LSTM has a *forget gate* $\mathbf{f}_t^i$. If the forget gate is activated, the network can forget the previous memory cell $\mathbf{c}_{t-1}^i$. Also, the *output gate* $\mathbf{o}_t^i$ controls the output of the memory cell. In this study, the architecture of LSTM is formulated as follows:

$$\begin{aligned}
\mathbf{i}_t^i &= \sigma(\mathbf{W}_i \mathbf{g}_t^i + \mathbf{U}_i \mathbf{h}_{t-1}^i + \mathbf{b}_i), \\
\mathbf{f}_t^i &= \sigma(\mathbf{W}_f \mathbf{g}_t^i + \mathbf{U}_f \mathbf{h}_{t-1}^i + \mathbf{b}_f), \\
\mathbf{o}_t^i &= \sigma(\mathbf{W}_o \mathbf{g}_t^i + \mathbf{U}_o \mathbf{h}_{t-1}^i + \mathbf{b}_o), \\
\theta_t^i &= \tanh(\mathbf{W}_g \mathbf{g}_t^i + \mathbf{U}_g \mathbf{h}_{t-1}^i + \mathbf{b}_g), \\
\mathbf{c}_t^i &= \mathbf{f}_t^i \circ \mathbf{c}_{t-1}^i + \mathbf{i}_t^i \circ \theta_t^i, \\
\mathbf{h}_t^i &= \mathbf{o}_t^i \circ \tanh(\mathbf{c}_t^i).
\end{aligned} \tag{3}$$

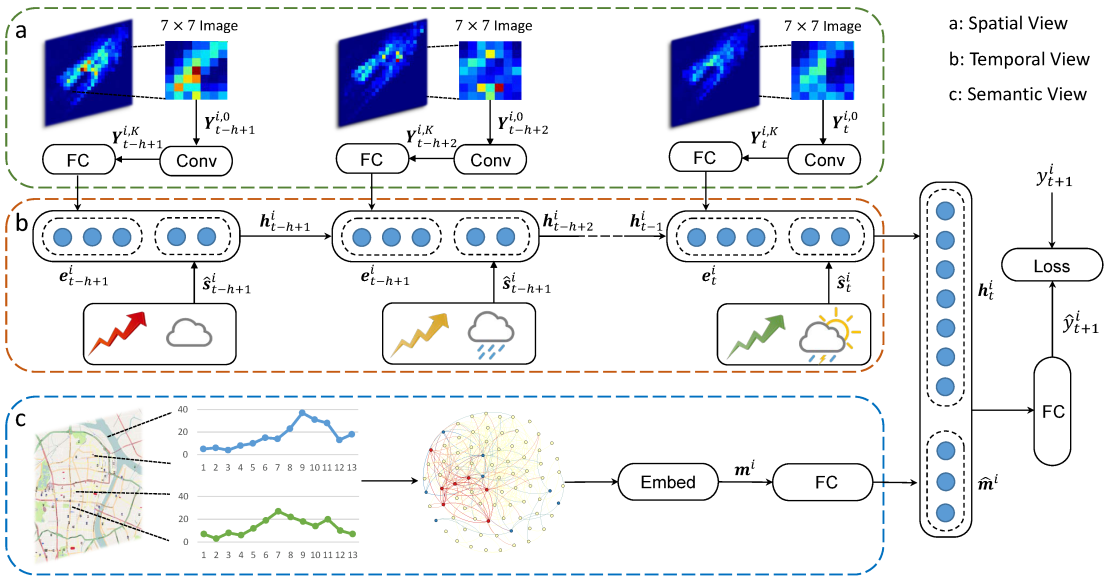


Figure 1: The Architecture of DMVST-Net. (a). The spatial component uses a local CNN to capture spatial dependency among nearby regions. The local CNN includes several convolutional layers. A fully connected layer is used at the end to get a low dimensional representation. (b). The temporal view employs a LSTM model, which takes the representations from the spatial view and concatenates them with context features at corresponding times. (c). The semantic view first constructs a weighted graph of regions (with weights representing functional similarity). Nodes are encoded into vectors. A fully connected layer is used at the end for jointly training. Finally, a fully connected neural network is used for prediction.

where $\circ$ denotes Hadamard product and $\tanh$ is hyperbolic tangent function. Both functions are element-wise. $\mathbf{W}_a, \mathbf{U}_a, \mathbf{b}_a$ $(a \in \{i, f, o, g\})$ are all learnable parameters. The number of time intervals in LSTM is h and the output of region i of LSTM after h time intervals is $\mathbf{h}_t^i$.

As Figure 1(b) shows, the temporal component takes representations from the spatial view and concatenates them with context features. More specifically, we define:

$$\mathbf{g}_t^i = \hat{\mathbf{s}}_t^i \oplus \mathbf{e}_t^i, \tag{4}$$

where $\oplus$ denotes the concatenation operator, therefore, $\mathbf{g}_t^i \in \mathbb{R}^{r+d}$.

Semantic View: Structural Embedding

Intuitively, locations sharing similar functionality may have similar demand patterns, e.g., residential areas may have a high number of demands in the morning when people transit to work, and commercial areas may expect to have high demands on weekends. Similar regions may not necessarily be close in space. Therefore, we construct a graph of locations representing functional (semantic) similarity among regions.

We define the semantic graph of location as $G = (V, E, D)$, where the set of locations L are nodes $V = L$, $E \in V \times V$ is the edge set, and D is a set of similarity on all the edges. We use Dynamic Time Warping (DTW) to measure the similarity ω_{ij} between node (location) i and node (location) j.

$$\omega_{ij} = \exp(-\alpha \mathrm{DTW}(i, j)), \tag{5}$$

where α is the parameter that controls the decay rate of the distance (in this paper, $\alpha = 1$), and $\mathrm{DTW}(i, j)$ is the dynamic time warping distance between the demand patterns of two locations. We use the average weekly demand time series as the demand patterns. The average is computed on the training data in the experiment. The graph is fully connected because every two regions can be reached.

In order to encode each node into a low dimensional vector and maintain the structural information, we apply a graph embedding method on the graph. For each node i (location), the embedding method outputs the embedded feature vector $\mathbf{m}^i$. In addition, in order to co-train the embedded $\mathbf{m}^i$ with our whole network architecture, we feed the feature vector $\mathbf{m}^i$ to a fully connected layer, which is defined as:

$$\hat{\mathbf{m}}^i = f(W_{fe}\mathbf{m}^i + b_{fe}), \tag{6}$$

W_{fe} and b_{fe} are both learnable parameters. In this paper, we use LINE for generating embeddings (Tang et al. 2015).

Prediction Component

Recall that our goal is to predict the demand at $t + 1$ given the data till t. We join three views together by concatenating $\hat{\mathbf{m}}^i$ with the output $\mathbf{h}_t^i$ of LSTM:

$$\mathbf{q}_t^i = \mathbf{h}_t^i \oplus \hat{\mathbf{m}}^i. \tag{7}$$

Note that the output of LSTM $\mathbf{h}_t^i$ contains both effects of temporal and spatial view. Then we feed $\mathbf{q}_t^i$ to the fully connected network to get the final prediction value $\hat{y}_{t+1}^i$ for each region. We define our final prediction function as:

$$\hat{y}_{t+1}^i = \sigma(W_{ff}\mathbf{q}_t^i + b_{ff}), \tag{8}$$

where W_{ff} and b_{ff} are learnable parameters. $\sigma(x)$ is a Sigmoid function defined as $\sigma(x) = 1/(1+e^{-x})$. The output of

Algorithm 1: Training Pipeline of DMVST-Net

Input: Historical observations: $\mathcal{Y}^L_{1,...,t}$; Context features: $\mathcal{E}^L_{t-h,...,t}$; Region structure graph $G = (V, E, D)$; Length of the time period h;
Output: Learned DMVST-Net model

```
Initialization;
for ∀i ∈ L do
    Use LINE on G and get the embedding result m^i;
    for ∀t ∈ [h, T] do
        S_spa = [Y^i_{t-h+1}, Y^i_{t-h+2}, ..., Y^i_t];
        S_cox = [e^i_{t-h+1}, e^i_{t-h+2}, ..., e^i_t];
        Append < {S_spa, S_cox, m^i}, y^i_{t+1} > to Ω_bt ;
    end
end
Initialize all learnable parameters θ in DMVST-Net;
repeat
    Randomly select a batch of instance Ω_bt from Ω;
    Optimize θ by minimizing the loss function
      Eq. (9) with Ω_bt
until stopping criteria is met;
```

our model is in $[0, 1]$, as the demand values are normalized. We later denormalize the prediction to get the actual demand values.

Loss function

In this section, we provide details about the loss function used for jointly training our proposed model. The loss function we used is defined as:

$$\mathcal{L}(\theta) = \sum_{i=1}^{N}((y^i_{t+1} - \hat{y}^i_{t+1})^2 + \gamma(\frac{y^i_{t+1} - \hat{y}^i_{t+1}}{y^i_{t+1}})^2), \quad (9)$$

where θ are all learnable parameters in the DMVST-Net and γ is a hyper parameter. The loss function consists of two parts: mean square loss and square of mean absolute percentage loss. In practice, mean square error is more relevant to predictions of large values. To avoid the training being dominated by large value samples, we in addition minimize the mean absolute percentage loss. Note that, in the experiment, all compared regression methods use the same loss function as defined in Eq. (9) for fair comparison. The training pipeline is outlined in Algorithm 1. We use Adam (Kingma and Ba 2014) for optimization. We use Tensorflow and Keras (Chollet and others 2015) to implement our proposed model.

Experiment

Dataset Description

In this paper, we use a large-scale online taxi request dataset collected from Didi Chuxing, which is one of the largest online car-hailing companies in China. The dataset contains taxi requests from $02/01/2017$ to $03/26/2017$ for the city of Guangzhou. There are 20×20 regions in our data. The size of each region is $0.7km \times 0.7km$. There are about $300,000$ requests each day on average. The context features used in our experiment are the similar types of features used in (Tong et al. 2017). These features include temporal features (e.g., the average demand value in the last four time intervals), spatial features (e.g., longitude and latitude of the region center), meteorological features (e.g., weather condition), event features (e.g., holiday).

In the experiment, the data from $02/01/2017$ to $03/19/2017$ is used for training (47 days), and the data from $03/20/2017$ to $03/26/2017$ (7 days) is used for testing. We use half an hour as the length of the time interval. When testing the prediction result, we use the previous 8 time intervals (i.e., 4 hours) to predict the taxi demand in the next time interval. In our experiment, we filter the samples with demand values less than 10. This is a common practice used in industry. Because in the real-world applications, people do not care about such low-demand scenarios.

Evaluation Metric

We use Mean Average Percentage Error (MAPE) and Rooted Mean Square Error (RMSE) to evaluate our algorithm, which are defined as follows:

$$MAPE = \frac{1}{\xi}\sum_{i=1}^{\xi}\frac{|\hat{y}^i_{t+1} - y^i_{t+1}|}{y^i_{t+1}}, \quad (10)$$

$$RMSE = \sqrt{\frac{1}{\xi}\sum_{i=1}^{\xi}(\hat{y}^i_{t+1} - y^i_{t+1})^2}, \quad (11)$$

where $\hat{y}^i_{t+1}$ and y^i_{t+1} mean the prediction value and real value of region i for time interval $t + 1$, and where ξ is total number of samples.

Methods for Comparison

We compared our model with the following methods, and tuned the parameters for all methods. We then reported the best performance.

- **Historical average (HA)**: Historical average predicts the demand using average values of previous demands at the location given in the same relative time interval (i.e., the same time of the day).
- **Autoregressive integrated moving average (ARIMA)**: ARIMA is a well-known model for forecasting time series which combines moving average and autoregressive components for modeling time series.
- **Linear regression (LR)**: We compare our method with different versions of linear regression methods: ordinary least squares regression (OLSR), Ridge Regression (i.e., with ℓ_2-norm regularization), and Lasso (i.e., with ℓ_1-norm regularization).
- **Multiple layer perceptron (MLP)**: We compare our method with a neural network of four fully connected layers. The number of hidden units are 128, 128, 64, and 64 respectively.
- **XGBoost** (Chen and Guestrin 2016): XGBoost is a powerful boosting tree based method and is widely used in data mining applications.

- **ST-ResNet** (Zhang, Zheng, and Qi 2017): ST-ResNet is a deep learning based approach for traffic prediction. The method constructs a city's traffic density map at different times as images. CNN is used to extract features from historical images.

We used the same context features for all regression methods above. For fair comparisons, all methods (except ARIMA and HA) use the same loss function as our method defined in Eq. (9).

We also studied the effect of different view components proposed in our method.

- **Temporal view**: For this variant, we used only LSTM with inputs as context features. Note that, if we do not use any context features but only use the demand value of last timestamp as input, LSTM does not perform well. It is necessary to use context features to enable LSTM to model the complex sequential interactions for these features.
- **Temporal view + Semantic view**: This method captures both temporal dependency and semantic information.
- **Temporal view + Spatial (Neighbors) view**: In this variant, we used the demand values of nearby regions at time interval t as $\hat{\mathbf{s}}_t^i$ and combined them with context features as the input of LSTM. We wanted to demonstrate that simply using neighboring regions as features cannot model the complex spatial relations as our proposed local CNN method.
- **Temporal view + Spatial (LCNN) view**: This variant considers both temporal and local spatial views. The spatial view uses the proposed local CNN for considering neighboring relation. Note that when our local CNN uses a local window that is large enough to cover the whole city, it is the same as the global CNN method. We studied the performance of different parameters and show that if the size is too large, the performance is worse, which indicates the importance of locality.
- **DMVST-Net**: Our proposed model, which combines spatial, temporal and semantic views.

Preprocessing and Parameters

We normalized the demand values for all locations to $[0, 1]$ by using Max-Min normalization on the training set. We used one-hot encoding to transform discrete features (e.g., holidays and weather conditions) and used Max-Min normalization to scale the continuous features (e.g., the average of demand value in last four time intervals). As our method outputs a value in $[0, 1]$, we applied the inverse of the Max-Min transformation obtained on training set to recover the demand value.

All these experiments were run on a cluster with four NVIDIA P100 GPUs. The size of each neighborhood considered was set as 9×9 (i.e., S = 9), which corresponds to $6km \times 6km$ rectangles. For spatial view, we set $K = 3$ (number of layers), $\tau = 3 \times 3$ (size of filter), $\lambda = 64$ (number of filters used), and $d = 64$ (dimension of the output). For the temporal component, we set the sequence length $h = 8$ (i.e., 4 hours) for LSTM. The output dimension of graph embedding is set as 32. The output dimension for the semantic view is set to 6. We used Sigmoid function as the activation function for the fully connected layer in the final prediction component. Activation functions in other fully connected layers are ReLU. Batch normalization is used in the local CNN component. The batch size in our experiment was set to 64. The first 90% of the training samples were selected for training each model and the remaining 10% were in the validation set for parameter tuning. We also used early-stop in all the experiments. The early-stop round and the max epoch were set to 10 and 100 in the experiment, respectively.

Table 1: Comparison with Different Baselines

Method	MAPE	RMSE
Historical average	0.2513	12.167
ARIMA	0.2215	11.932
Ordinary least square regression	0.2063	10.234
Ridge regression	0.2061	10.224
Lasso	0.2091	10.327
Multiple layer perceptron	0.1840	10.609
XGBoost	0.1953	10.012
ST-ResNet	0.1971	10.298
DMVST-Net	**0.1616**	**9.642**

Performance Comparison

Comparison with state-of-the-art methods. Table 1 shows the performance of the proposed method as compared to all other competing methods. DMVST-Net achieves the lowest MAPE (0.1616) and the lowest RMSE (9.642) among all the methods, which is 12.17% (MAPE) and 3.70% (RMSE) relative improvement over the best performance among baseline methods. More specifically, we can see that HA and ARIMA perform poorly (i.e., have a MAPE of 0.2513 and 0.2215, respectively), as they rely purely on historical demand values for prediction. Regression methods (OLSR, LASSO, Ridge, MLP and XGBoost) further consider context features and therefore achieve better performance. Note that the regression methods use the same loss function as our method defined in Eq. (9). However, the regression methods do not model the temporal and spatial dependency. Consequently, our proposed method significantly outperforms those methods.

Furthermore, our proposed method achieves 18.01% (MAPE) and 6.37% relative improvement over ST-ResNet. Compared with ST-ResNet, our proposed method further utilizes LSTM to model the temporal dependency, while at the same time considering context features. In addition, our use of local CNN and semantic view better captures the correlation among regions.

Comparison with variants of our proposed method. Table 2 shows the performance of DMVST-Net and its variants. First, we can see that both Temporal view + Spatial (Neighbor) view and Temporal view + Spatial (LCNN) view achieve a lower MAPE (a reduction of 0.63% and 6.10%, respectively). The result demonstrates the effectiveness of

Table 2: Comparison with Variants of DMVST-Net

Method	MAPE	RMSE
Temporal view	0.1721	9.812
Temporal + Semantic view	0.1708	9.789
Temporal + Spatial (Neighbor) view	0.1710	9.796
Temporal + Spatial (LCNN) view	0.1640	9.695
DMVST-Net	**0.1616**	**9.642**

considering neighboring spatial dependency. Furthermore, Temporal view + Spatial (LCNN) view outperforms Temporal view + Spatial (Neighbor) view significantly, as the local CNN can better capture the nonlinear relations. On the other hand, Temporal view + Semantic view has a lower MAPE of 0.1708 and an RMSE of 9.789 compared to Temporal view only, demonstrating the effectiveness of our semantic view. Lastly, the performance is best when all views are combined.

Performance on Different Days

Figure 2 shows the performance of different methods on different days of the week. Due to the space limitation, We only show MAPE here. We get the same conclusions of RMSE. We exclude the results of HA and ARIMA, as they perform poorly. We show Ridge regression results as they perform best among linear regression models. In the figure, it shows that our proposed method DMVST-Net outperforms other methods consistently in all seven days. The result demonstrates that our method is robust.

Moreover, we can see that predictions on weekends are generally worse than on weekdays. Since the average number of demand requests is similar (45.42 and 43.76 for weekdays and weekends, respectively), we believe the prediction task is harder for weekends as demand patterns are less regular. For example, we can expect that residential areas may have high demands in the morning hours on weekdays, as people need to transit to work. Such regular patterns are less likely to happen on weekends. To evaluate the robustness of our method, we look at the relative increase in prediction error on weekends as compared to weekdays, i.e., defined as $|\bar{wk} - \bar{wd}|/\bar{wd}$, where $\bar{wd}$ and $\bar{wk}$ are the average prediction error of weekdays and weekends, respectively. The results are shown in Table 3. For our proposed method, the relative increase in error is the smallest, at 4.04%.

At the same time, considering temporal view, only (LSTM) has a relative increase in error of 4.77%, while the increase is more than 10% for Ridge regression, MLP, and XGBoost. The more stable performance of LSTM can be attributed to its modeling of the temporal dependency. We see that ST-ResNet has a more consistent performance (relative increase in error of 4.41%), as the method further models the spatial dependency. Finally, our proposed method is more robust than ST-ResNet.

Influence of Sequence Length for LSTM

In this section, we study how the sequence length for LSTM affects the performance. Figure 3a shows the prediction error of MAPE with respect to the length. We can see that when the length is 4 hours, our method achieves the best performance. The decreasing trend in MAPE as the length increases shows the importance of considering the temporal dependency. Furthermore, as the length increases to more than 4 hours, the performance slightly degrades but mainly remains stable. One potential reason is that when considering longer temporal dependency, more parameters need to be learned. As a result, the training becomes harder.

Table 3: Relative Increase in Error (RIE) on Weekends to Weekdays

Method	RIDGE	MLP	XGBoost	ST-ResNet	Temporal	DMVST-Net
RIE	14.91%	10.71%	16.08%	4.41%	4.77%	**4.04%**

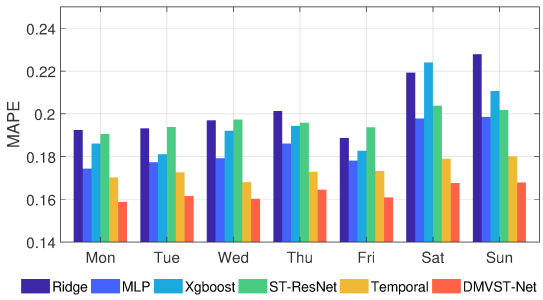


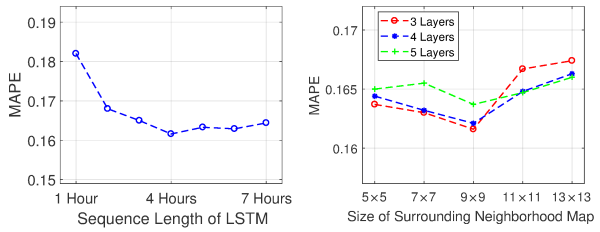
Figure 2: The Results of Different Days.

Influence of Input Size for Local CNN

Our intuition was that applying CNN locally avoids learning relation among weakly related locations. We verified that intuition by varying the input size S for local CNN. As the input size S becomes larger, the model may fit for relations in a larger area. In Figure 3b, we show the performance of our method with respect to the size of the surrounding neighborhood map. We can see that when there are three convolutional layers and the size of map is 9×9, the method achieves the best performance. The prediction error increases as the size decreases to 5×5. This may be due to the fact that locally correlated neighboring locations are not fully covered. Furthermore, the prediction error increases significantly (more than 3.46%), as the size increases to 13×13 (where each area approximately covers more than 40% of the space in GuangZhou). The result suggests that locally significant correlations may be averaged as the size increases. We also increased the number of convolution layers to four and five layers, as the CNN needed to cover larger area. However, we observed similar trends of prediction error, as shown in Figure 3b. We can now see that the input size for local CNN when the method performs best remains consistent (i.e., the size of map is 9×9).

Conclusion and Discussion

The purpose of this paper is to inform of our proposal of a novel Deep Multi-View Spatial-Temporal Network (DMVST-Net) for predicting taxi demand. Our approach in-



(a) (b)

Figure 3: (a) MAPE with respect to sequence length for LSTM. (b) MAPE with respect to the input size for local CNN.

tegrates the spatial, temporal, and semantic views, which are modeled by local CNN, LSTM and semantic graph embedding, respectively. We evaluated our model on a large-scale taxi demand dataset. The experiment results show that our proposed method significantly outperforms several competing methods.

As deep learning methods are often difficult to interpret, it is important to understand what contributes to the improvement. This is particularly important for policy makers. For future work, we plan to further investigate the performance improvement of our approach for better interpretability. In addition, seeing as the semantic information is implicitly modeled in this paper, we plan to incorporate more explicit information (e.g., POI information) in our future work.

Acknowledgments

The work was supported in part by NSF awards #1544455, #1652525, #1618448, and #1639150. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

References

Chen, T., and Guestrin, C. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. ACM.

Chollet, F., et al. 2015. Keras. https://github.com/fchollet/keras.

Deng, D.; Shahabi, C.; Demiryurek, U.; Zhu, L.; Yu, R.; and Liu, Y. 2016. Latent space model for road networks to predict time-varying traffic. *Proceedings of the 22th ACM SIGKDD international conference on Knowledge discovery and data mining*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Hochreiter, S.; Bengio, Y.; Frasconi, P.; and Schmidhuber, J. 2001. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.

Idé, T., and Sugiyama, M. 2011. Trajectory regression on road networks. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 203–208. AAAI Press.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *Nature* 521(7553):436–444.

Li, X.; Pan, G.; Wu, Z.; Qi, G.; Li, S.; Zhang, D.; Zhang, W.; and Wang, Z. 2012. Prediction of urban human mobility using large-scale taxi traces and its applications. *Frontiers of Computer Science* 6(1):111–121.

Ma, X.; Dai, Z.; He, Z.; Ma, J.; Wang, Y.; and Wang, Y. 2017. Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. *Sensors* 17(4):818.

Moreira-Matias, L.; Gama, J.; Ferreira, M.; Mendes-Moreira, J.; and Damas, L. 2013. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems* 14(3):1393–1402.

Pan, B.; Demiryurek, U.; and Shahabi, C. 2012. Utilizing real-world transportation data for accurate traffic prediction. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, 595–604. IEEE.

Shekhar, S., and Williams, B. 2008. Adaptive seasonal time series models for forecasting short-term traffic flow. *Transportation Research Record: Journal of the Transportation Research Board* (2024):116–125.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, 1067–1077. International World Wide Web Conferences Steering Committee.

Tobler, W. R. 1970. A computer movie simulating urban growth in the detroit region. *Economic geography* 46(sup1):234–240.

Tong, Y.; Chen, Y.; Zhou, Z.; Chen, L.; Wang, J.; Yang, Q.; and Ye, J. 2017. The simpler the better: A unified approach to predicting original taxi demands on large-scale online platforms. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM.

Wang, D.; Cao, W.; Li, J.; and Ye, J. 2017. Deepsd: Supply-demand prediction for online car-hailing services using deep neural networks. In *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*, 243–254. IEEE.

Wu, F.; Wang, H.; and Li, Z. 2016. Interpreting traffic dynamics using ubiquitous urban data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 69. ACM.

Yu, R.; Li, Y.; Demiryurek, U.; Shahabi, C.; and Liu, Y. 2017. Deep learning: A generic approach for extreme condition traffic forecasting. In *Proceedings of SIAM International Conference on Data Mining*.

Zhang, J.; Zheng, Y.; Qi, D.; Li, R.; and Yi, X. 2016. Dnn-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 92. ACM.

Zhang, J.; Zheng, Y.; and Qi, D. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*.

Zheng, J., and Ni, L. M. 2013. Time-dependent trajectory regression on road networks via multi-task learning. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, 1048–1055. AAAI Press.