MDTP: A Multi-source Deep Traffic Prediction Framework over Spatio-Temporal Trajectory Data

Ziquan Fang[†], Lu Pan[†], Lu Chen[†], Yuntao Du[†], Yunjun Gao[†,♯]
[†]College of Computer Science, Zhejiang University, Hangzhou, China
[♯]Alibaba–Zhejiang University Joint Institute of Frontier Technologies, Hangzhou, China
{zqfang, play, luchen, ytdu, gaoyj}@zju.edu.cn

ABSTRACT

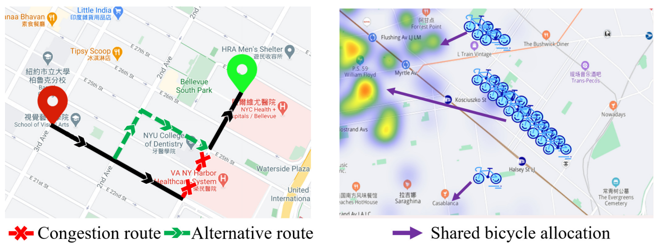
Traffic prediction has drawn increasing attention for its ubiquitous real-life applications in traffic management, urban computing, public safety, and so on. Recently, the availability of massive trajectory data and the success of deep learning motivate a plethora of deep traffic prediction studies. However, the existing neural-network-based approaches tend to ignore the correlations between multiple types of moving objects located in the same spatio-temporal traffic area, which is suboptimal for traffic prediction analytics.

In this paper, we propose a multi-source deep traffic prediction framework over spatio-temporal trajectory data, termed as **MDTP**. The framework includes two phases: spatio-temporal feature modeling and multi-source bridging. We present an enhanced graph convolutional network (GCN) model combined with long short-term memory network (LSTM) to capture the spatial dependencies and temporal dynamics of traffic in the feature modeling phase. In the multi-source bridging phase, we propose two methods, Sum and Concat, to connect the learned features from different trajectory data sources. Extensive experiments on two real-life datasets show that MDTP i) has superior efficiency, compared with classical time-series methods, machine learning methods, and state-of-the-art neural-network-based approaches; ii) offers a significant performance improvement over the single-source traffic prediction approach; and iii) performs traffic predictions in seconds even on tens of millions of trajectory data. we develop **MDTP$^+$**, a user-friendly interactive system to demonstrate traffic prediction analysis.

PVLDB Reference Format:
Ziquan Fang, Lu Pan, Lu Chen, Yuntao Du, Yunjun Gao. MDTP: A Multi-source Deep Traffic Prediction Framework over Spatio-Temporal Trajectory Data. PVLDB, 14(8): 1289 - 1297, 2021.
doi:10.14778/3457390.3457394

1 INTRODUCTION

Traffic prediction that forecasts future traffic status (e.g., traffic volume of a road network) based on historical traffic data, serves a wide range of real-life applications in traffic management [2], urban computing [36], public safety [31], and so on. Here we give two examples. As the first example shown in Figure 1(a), the accurate volume prediction of a road network can support real-time road



(a) Road Navigation (b) Bike Allocation

Figure 1: Traffic Prediction for Real-life Applications

navigation services to help drivers choose a more convenient travel path when they are driving. Except for that, good traffic prediction is also essential for transportation scheduling. As another example depicted in Figure 1(b), a precise region volume prediction can assist bike allocation by distributing more shared-bicycles in crowded urban regions to ease traffic congestion in advance.

With the increasing availability of massive spatio-temporal trajectory data and the urgent demand to build intelligent transportation system (ITS), many studies have investigated the problem of traffic prediction in terms of volume prediction [19], speed prediction [11], flow prediction [33], and so on. In this paper, we mostly focus on the volume prediction, since it is the key and foundation to solve a series of traffic problems [24]. Figure 2 illustrates a volume prediction example. There are 4 disjointed nodes where each node denotes a region in the map, while the directed edges represent the transition flows among these observed regions. Based on that, we can calculate the traffic volume including *in* and *out* volume of each region during each discretized time interval. For example, the *in* volume of region r_1 during T_1 is 2 and the corresponding *out* volume is 4, since there are 2 vehicles (i.e., one bike and one taxi) flowing in r_1 and 4 vehicles (i.e., three bikes and one taxi) flowing out r_1 during time interval T_1. The overall traffic volume during T_1 is as below: region r_1 (*in*: 2, *out*: 4), region r_2 (*in*: 3, *out*: 3), region r_3 (*in*: 3, *out*: 1), and region r_4 (*in*: 3, *out*: 3). Thus, given historical traffic status (i.e., traffic volume of each region from T_1 to T_n), we aim to predict the traffic volume of all observed regions for the future time T_{n+1}. To address this, many classical methods have been proposed, which can be classified into time-series based and traditional machine learning based ones. However, those approaches either do not well capture the complex nonlinear spatio-temporal dependencies in traffic networks or rely heavily on additional manual feature extraction processing [26]. The more details are discussed in Section 2.1. As a result, they are unsuitable and inefficient for large-scale, dynamic, and complex traffic prediction tasks.

Proceedings of the VLDB Endowment, Vol. 14, No. 8 ISSN 2150-8097.
doi:10.14778/3457390.3457394

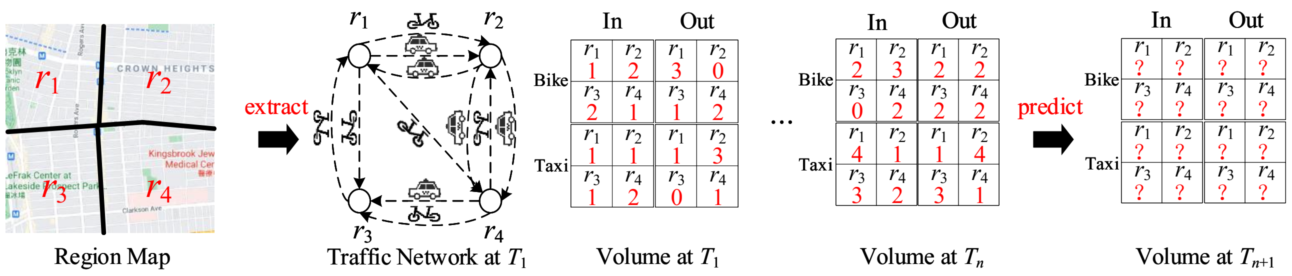
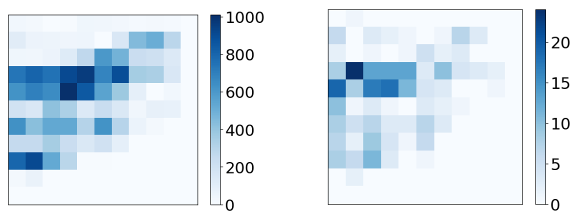


Figure 2: Illustration of Traffic Volume Prediction

More recently, inspired by the success of deep learning technologies in natural language processing [5], computer vision [18], and speech recognition [16], many researchers start to explore the end-to-end learning abilities of various deep learning models for data-driven traffic prediction studies, which aim to model the complex spatial and temporal dependencies of traffic networks by neural-network models. Three challenges exist when applying deep learning models to traffic prediction, as stated below.

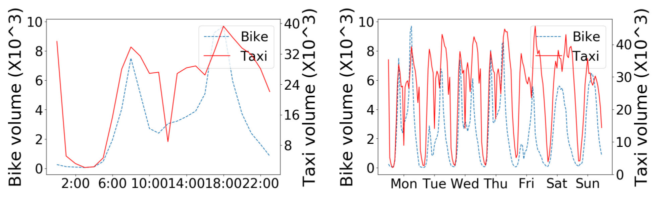
Challenge I: Multi-source data influence. Different types of moving objects may have spatio-temporal correlations when they move in the same/similar spatio-temporal traffic areas. To verify such an assumption, we give visualization insights into the evaluated public trajectory datasets (i.e., the NYC-Taxi and NYC-Bike whose dataset descriptions are detailed in Section 5) in terms of their spatial and temporal correlations, respectively. (i) To discover the spatial correlation, we first partition the whole city into 11*11 disjointed spatial regions and then count the number of taxis and bikes in each region. Finally, we plot the results in the form of heatmaps. Figure 3 shows the taxi distribution and bike distribution maps of New York City. It is obvious that their spatial distributions are similar, which implies their potential correlations. (ii) To find the temporal correlation, we choose a day/week, and split the whole time span into equal time slots, where each time slot represents four hours of a day or one day of a week. Next, we calculate the number of taxis and bikes in each time slot. As depicted in Figure 7, taxis and bikes have similar moving trends in a day/week, which also indicates their temporal dependence. Nonetheless, existing deep traffic prediction methods only consider single type of trajectory data (e.g., taxi or bike), and they can only forecast one type of traffic volume at each time. In other words, they predict taxi volume with historical taxi data or estimate bike volume with historical bike data. Motivated by these, we propose the multi-source deep traffic prediction framework, which can not only extract the hidden complex spatio-temporal correlations between different kinds of moving objects to improve traffic prediction performance but also supply a multi-prediction mechanism to forecast various traffic volumes simultaneously.

Challenge II: Dynamic spatial dependencies. Although existing deep traffic prediction studies use convolutional neural networks (CNNs) and graph convolutional networks (GCNs) to capture spatial dependencies, two limitations exist. On the one hand, the CNNs-based architectures capture the neighbor spatial traffic dependence step by step, but fail to directly capture long-term spatial dependence [8]. On the other hand, although the most popular GCNs-based approaches can learn more hidden features of traffic networks



(a) Taxi Distribution of NYC (b) Bike Distribution of NYC

Figure 3: Spatial Distribution Map of Taxi and Bike



(a) Day (b) Week

Figure 4: Temporal Correlation between Taxi and Bike

compared with CNNs, they are inefficient for capturing dynamic spatial traffic dependence. The dynamic spatial dependence means that the relationship between two static locations may vary at different timestamps. For example, the spatial connections between residential and commercial areas at morning and evening peak hours have more relevant than those at other times. GCNs-based traffic prediction models consist of two core concepts, *node* and *edge*, where a node denotes a road intersection [6] or a region [35] while an edge represents the relationship between nodes. Existing GCNs-based methods tend to assign edges with statistics attributes (such as road distance [30] and additional context [21]), which leads model to learn static spatial dependence of traffic network. To solve this challenge, we denote the edges of GCN model with dynamic traffic flows. Hence, our deep traffic prediction model can explicitly model dynamic spatial correlations of traffic network.

Challenge III: Demonstration of deep traffic prediction. Existing deep traffic prediction methods mainly focus on improving the efficiency and effectiveness of various traffic prediction tasks. However, little work puts attention on the visualization and demonstration of traffic prediction analysis, making it hard for developers and users to view and apply it to real-world application scenarios. To tackle this challenge, we further develop a user-friendly interactive system based on our proposed deep traffic prediction framework.

To address the above challenges, we present a multi-source deep traffic prediction framework over spatio-temporal trajectory data, i.e., **MDTP**, which achieve an effective and multi-prediction analysis. Specifically, our framework MDTP mainly contains two phases: (i) spatio-temporal feature modeling that captures dynamic spatio-temporal traffic dependencies, and (ii) multi-source bridging that discovers co-relations between multi-source trajectory data for improving traffic prediction performance. To sum up, this paper makes the following key contributions:

- *Multi-source deep traffic prediction.* We, for the first time, propose a deep traffic prediction framework MDTP over multi-source trajectory data, which utilizes the spatio-temporal correlations between different types of moving objects. Once the MDTP is trained, it can be efficiently adapted to simultaneously predict different types of traffic volumes.
- *Dynamic traffic network modeling.* We present an effective dynamic traffic network modeling approach to capture both spatial and temporal dependencies. Specifically, we introduce an enhanced GCN model with the traffic flow to learn the spatial dependency of the dynamic traffic network. Then, we use the LSTM model to capture the dynamic and long-term temporal relationships in the time-series of traffic status.
- *Visualization demonstration.* We develop a new cross-platform system called MDTP$^+$ based on the MDTP framework, which is a useful tool to help apply our framework in real-life applications. Besides, MDTP$^+$ provides a user-friendly interactive interface for traffic prediction analytics.
- *Extensive experiments.* We conduct extensive experimental evaluations on two large-scale real trajectory data sets. The results demonstrate that our proposed MDTP can achieve the best prediction performance, compared with 6 popular baselines. Besides, MDTP also offers superior scalability and efficiency as it makes predictions in seconds even on tens of millions of trajectory data.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 presents the problem definition. Section 4 details our framework MDTP. The experimental results are reported in Section 5, and the demonstration of our framework MDTP is presented in Section 6. Finally, Section 7 concludes the paper and offers potential research directions.

2 RELATED WORK

In this section, we review the related traffic prediction work including traditional methods and deep learning models.

2.1 Traditional Prediction Methods

In the earlier traffic prediction studies, a wide range of methods [25] have been proposed, including time-series and traditional machine learning methods. For the time-series methods, researchers treat the traffic prediction as a time-series prediction problem. Hence, the classic Autoregressive Integrated Moving Average (ARIMA) models combined with Kalman Filtering [9, 15] have been widely used for traffic prediction. However, those time-series methods are based on linear models which assume that the traffic is stationary. Thus, they fail to model the complex nonlinear relationships in the time dimension. Also, they ignore the traffic network's spatial dependence. To capture spatio-temporal correlations from traffic networks, machine learning models are employed to make a flexible data-driven traffic prediction. Traditional machine learning methods include the support vector regression [10], k-nearest neighbor [29], Bayesian Network [37], and probabilistic models [1]. Nevertheless, these methods achieve a high prediction accuracy for simple and small-size traffic data, and thus, they are inefficient to mine the deep and implicit spatio-temporal correlations in massive traffic data.

2.2 Deep Learning Prediction Models

Inspired by the success of deep learning in many application scenarios, the deep-learning-based methods are becoming the most popular in traffic prediction [20]. Many researchers apply deep learning models to traffic prediction, such as CNNs-based, RNNs-based, and hybrid models-based. CNNs-based approaches [12, 13] model the city traffic as an image that consists of disjointed equal grids, where each grid models its spatial traffic features. They use convolutional neural networks (CNNs) to capture the spatial dependency of traffic network. The RNNs-based methods [4, 14] utilize recurrent neural networks (RNNs) to extract the temporal correlations of traffic status. However, CNNs-based and RNNs-based methods model the traffic's spatial and temporal dependency separately. To this end, hybrid-based methods are developed. Wu et al. [23] propose a CNN-RNN model, to jointly capture the spatio-temporal features of traffic network. Since the traffic is also affected by external factors (e.g., weather and activities), Yao et al. [27] integrate CNN, LSTM, and semantic embedding to improve prediction performance. Recently, Zhang et al. [33] offer a new CNN-based multi-task deep learning framework to simultaneously forecast the node flow and edge flow. Yao et al. [26] combines CNN and multiple LSTMs to capture the long-term period dependence of dynamic traffic networks. Nonetheless, the above approaches can only capture neighbor spatial dependence step by step, but fail to directly capture long-term spatial dependence in a big traffic network.

More recently, the state-of-the-art graph convolutional network (GCNs) is used to exploit the graph structure information in city traffic [22, 28], e.g., road network and region network. Specifically, the GCNs build blocks to learn graph-structured data via pre-defined nodes and edges. Based on this, GCNs enable extracting a node's high-level features by aggregating feature information from its neighbor nodes through edges. Inspired by that, existing GCN-based traffic prediction methods typically model the city traffic as a traffic graph, where the node denotes traffic object (e.g., a sensor or a road intersection) and the edge represents relation between nodes. Hence, the node features (i.e., traffic volume) can be captured via GCN models. However, existing GCN-based methods mostly assign a static weight to the edge, such as road distance or static information, which are not able to deal with dynamic traffic network. To this end, we present an enhanced GCN model to capture the temporal and spatial dependence of dynamic city traffic, where the node denotes region volumes, and the edge represents region flows. Note that, all the above deep traffic prediction approaches model the traffic network based on a single type of trajectory data, and ignore the correlations between multiple types of moving objects in the same spatio-temporal conditions. In contrast, our proposed MDTP method handles the complex and dynamic traffic prediction by capturing the hidden correlations across multi-source trajectories.

3 PROBLEM STATEMENT

Since we model the city traffic using GCNs [28], a specific traffic graph structure $G = (V, E)$ should be carefully designed to capture spatio-temporal dependence of dynamic traffic. In this section, we first define a *node* or an *edge* in our graph model G, and then, we give the problem formulation of multi-source deep traffic prediction.

Definition 3.1. **(Node)** *To model the traffic with the graph topology, we partition the whole city space into an $I \times J$ grid map based on the longitude and latitude, denoted as $V = \{r_1, r_2, ..., r_{I \times J}\}$. Each region $r_i (1 \leq i \leq I \times J)$ represents a spatial node.*

There are totally $N = I \times J$ nodes (i.e., regions). As an example shown in Figure 2, there exists four regions in the map, including r_1, r_2, r_3, and r_4. We model each region as a node, which associates with two types of volumes, *in* volume and *out* volume.

Definition 3.2. **(In/Out Volume)** *Given a set P of trajectories and a discretized time interval T, for a node r_{ij} that locates at the i^{th} row and j^{th} column in the grid map, it associates with two types of volumes during time interval T, as defined respectively.*

$$x_T^{in,i,j} = \sum_{T_r \in P} \left|\left\{T_r(e).l \in r_{ij} \wedge T_r(s).l \notin r_{ij} \wedge T_r(e).t \in T\right\}\right| \quad (1)$$

$$x_T^{out,i,j} = \sum_{T_r \in P} \left|\left\{T_r(s).l \in r_{ij} \wedge T_r(e).l \notin r_{ij} \wedge T_r(s).t \in T\right\}\right| \quad (2)$$

where $x_T^{in,i,j}$ and $x_T^{out,i,j}$ denote the in volume and out volume of region r_{ij} in time interval T, respectively. T_r is a trajectory in P, and it consists of time-ordered GPS points in the form of (l, t), of which l is the geo-location and t denotes the timestamp. $T_r(s)$ and $T_r(e)$ represent the start and end GPS point of T_r, respectively. In addition, $|\cdot|$ denotes the cardinality of a set.

For example, in Figure 2, for each node/region at time interval T_1, we can calculate its *in* volumes of taxi and bike as well as *out* volumes of taxi and bike according to Definition 3.2. For four nodes (i.e., r_1, r_2, r_3, and r_4) at T_1, the *in* and *out* volumes of the bike are (1, 2, 2, 1) and (3, 0, 1, 2) respectively; and the *in* and *out* volumes of taxi are (1, 1, 1, 2) and (1, 3, 0, 1) respectively.

Definition 3.3. **(Edge)** *Given a trajectory set P and a time interval T, an edge $e_T^{(r_s,r_e)}$ connects nodes r_s and r_e during time interval T, which denotes the directed volume flow from r_s to r_e at T.*

$$e_T^{(r_s,r_e)} = \sum_{T_r \in P} |\{T_r(s) \in r_s \wedge T_r(e) \in r_e \wedge T_r(s).t \in T \wedge T_r(e).t \in T\}| \quad (3)$$

where r_s and r_e are start node and end node of $T_r \in P$, respectively.

Note that, Equation 3 defines the edge weight. Different from the previous studies that simply employ the spatial distance between nodes as the edge weights, we utilize the direct volume flows between nodes as the edge weights. Hence, the edge is dynamic since the volume flow from r_s to r_e varies across different time intervals. As an example depicted in Figure 2, for the bike, $e_{T_1}^{(r_1,r_2)}$ is 1, and $e_{T_1}^{(r_2,r_1)}$ is 0; for the taxi, $e_{T_1}^{(r_1,r_2)}$ is 1, and $e_{T_1}^{(r_2,r_1)}$ is 1.

Based on the node and edge, a city traffic can be represented by $G = (V, E)$, where V denotes the set of nodes and E represents the set of directed edges between all node pairs. Note that, G is a dynamic graph since the *in*/*out* volume of each node and the volume flow (i.e., edge) between nodes vary across different times.

A particular graph $G_T = (V, E)$ captures the topological state of the spatio-temporal traffic during a time interval T. The input (feature) matrix derived from G_T is denoted as $X_T = [\mathcal{X}_T^1, ..., \mathcal{X}_T^N] \in \mathcal{R}^{N \times F_I \times M}$, where $\mathcal{X}_T^i (1 \leq i \leq N)$ represents the F_I features (i.e., *in* and *out* volumes) of node r_i considering M multi-source traffic flows at time interval T. For example, $M = 2$ means that two sources of flows (i.e., bike and taxi flows) are used for traffic prediction. The adjacency matrix $A_T^{flow} \in \mathcal{R}^{N \times N \times M}$ derived from G_T denotes the edge information between nodes during time interval T, where $A_{ij} \in A_T^{flow}$ represents the directed edge from node r_i to node r_j.

Definition 3.4. **(Multi-source Deep Traffic Prediction)**. *Given a set $\mathcal{G} = \{G_{T\text{-}S+1}, G_{T\text{-}S+2}, ..., G_T\}$ of historical traffic graphs over past S time intervals from current time interval T, its corresponding feature matrix set is $\mathcal{X} = \{X_{T\text{-}S+1}, X_{T\text{-}S+2}, ..., X_T\}$ and the adjacency matrix set is $\mathcal{A} = \{A_{T\text{-}S+1}, A_{T\text{-}S+2}, ..., A_T\}$.* Note that $\mathcal{G}$, $\mathcal{X}$, and $\mathcal{A} \in \mathcal{R}^{S \times M}$, where M is the number of traffic flow sources. Multi-source deep traffic prediction *is to learn a neural-network function f to forecast the traffic status X_{T+1} during the next time T+1.*

$$X_{T+1} = \boldsymbol{f}(\mathcal{G}) \quad or \quad X_{T+1} = \boldsymbol{f}(\mathcal{X}, \mathcal{A}) \quad (4)$$

4 OUR METHOD MDTP

In this section, we first give an overview of our framework MDTP, and then, we present the detailed methods.

4.1 Overview

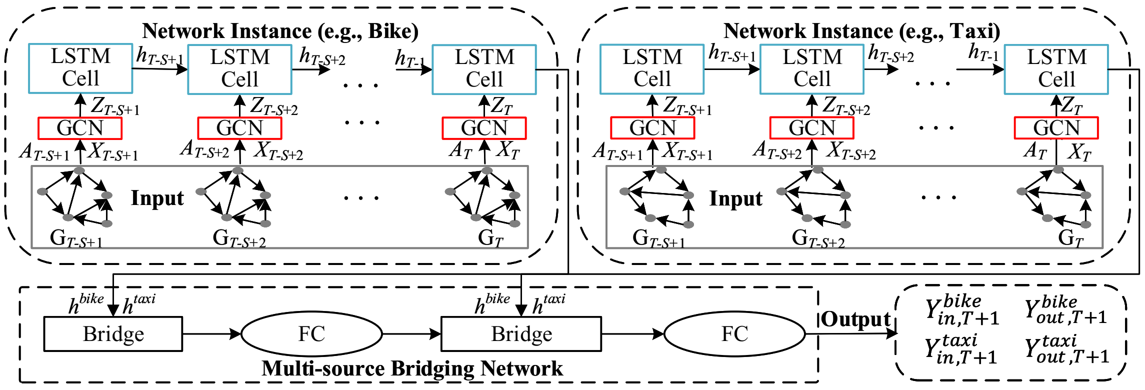
Figure 5 shows MDTP overview, which consists of spatio-temporal (ST) feature modeling and multi-source bridging. In the first phase, two networks (one for bike while one for taxi) are constructed to capture the dynamic spatial and temporal features of two types of moving objects over each time interval in $\mathcal{T} = \{T - S + 1, T - S + 2, ..., T\}$. This is because, to train a neural-network for prediction, a set of traffic graphs during the past S time intervals are required as defined in Definition 3.4. In the second phase, ST features learned from bike and taxi are merged to capture the correlation between them. Next, we detail the two processing phases, respectively.

4.2 Spatio-temporal Feature Modeling

In ST feature modeling phase, the traffic graphs during the past S time intervals are first sequentially fed into our GCN model to capture the spatial dependence and then fed into the LSTM network to learn the long-term temporal relationships of traffic network.

Spatial dependency. As discussed in Section 3, a traffic graph $G_T = (V, E_T)$ captures the topological state of the dynamic traffic during a time interval T. For each G_T, we can get an input feature matrix X_T to denote the features of all nodes, and a counterpart weight matrix A_T that is also called adjacency matrix to represent the edges between nodes. In MDTP, A is a dynamic flow adjacency matrix, termed as A^{flow}, whose entry A_{ij} denotes the directed volume flow from node r_i to node r_j in Definition 3.3. Therefore, we define the new enhanced graph convolution layer as below,

$$Z = A^{flow} X W \quad (5)$$



Network Instance (e.g., Bike)
Network Instance (e.g., Taxi)
LSTM Cell
GCN
Input
Bridge
FC
Output
Multi-source Bridging Network

Figure 5: An Overview of Our Framework MDTP

Temporal dependency. We directly utilize the LSTM network to capture the evolving temporal relationships of the dynamic traffic, because LSTM can learn the long-term dependencies of sequential traffic data [23]. Specifically, after extracting the spatial dependency of traffic graph G using the above GCN layer, we can get a sequence of learned traffic status $\{Z_{T-S+1}, Z_{T-S+2}, ..., Z_T\}$. Then, we feed each $Z_t(T - S + 1 \leq t \leq T)$ into the LSTM layer in order. In other words, Z_t is output of the GCN layer and the input for the LSTM layer. By doing these, the LSTM model could model the long temporal dependency of the dynamic traffic network.

The standard LSTM can be described as an encapsulated cell with several multiplicative gate units. For a certain time interval T, the LSTM cell takes as inputs current input vector x_T and the state vector of the last time step h_{T-1}, and outputs the state vector at current time step h_T. Specifically, $h_T = o_T \odot \tanh(s_T)$, where $o_T = \sigma\left(M_x^o x_T + M_h^o h_{T-1} + b^o\right)$ and $s_T = f_T \odot s_{T-1} + i_T \odot \tilde{s_T}$. Besides, $f_T = \sigma\left(M_x^f x_T + M_h^f h_{T-1} + b^f\right)$, $i_T = \sigma\left(M_x^i x_T + M_h^i h_{T-1} + b^i\right)$, and $\tilde{s_T} = \sigma\left(M_x^s x_T + M_h^s h_{T-1} + b^s\right)$. Here, i_T, f_T, o_T, and s_T represent input gate, forget gate, output gate, and memory cell, respectively. $\{M_x, M_h, b\}$ are the parameters of the corresponding unit. $\sigma(.)$ is sigmoid activation function, and $\odot$ denotes element-wise multiplication. In terms MDTP, the x_T is the learned traffic representation Z_T that corresponds to the original traffic graph G_T. Eventually, we treat the last hidden state h_{T+1} as the prediction result for the subsequent processing.

4.3 Multi-source Bridging

Considering the correlations between multiple types of moving objects, the ST-features captured from multi-sources (e.g., taxi and bike) in the first feature modeling phase should be merged, to better represent the traffic network. As shown in Figure 5, two latent representations based on the bike network and the taxi network, i.e., h_{taxi} and h_{bike}, are connected in the multi-source bridging network. Specifically, the bridging pipeline first feeds two latent representations into the bridge component (i.e., Bridge) and develops a fully connected network layer (i.e., FC). Then, it feeds the latent representations and the output of the previous FC layer together into the bridge component, and finally develops another FC layer for final prediction results. Such a bridging pipeline enables simultaneously capturing ST-features from multiple moving objects and the relationship between them. Hence, MDTP can achieve more accurate traffic prediction results, as verified in experiments. Next, we detail two connect methods [33] used in the bridge component.

Sum Connect. The sum connect method directly sum up two latent representations (i.e., h_{taxi} and h_{bike}), because they have the same dimensionality.

$$\mathcal{H}(d,:) = h_{bike}(d,:) + h_{taxi}(d,:), d = 0, 1, ..., D \tag{6}$$

where D denotes the dimensionality of two latent representations h_{taxi} and h_{bike}, and $\mathcal{H} \in \mathcal{R}^{D\times N}$.

Concat Connect. The concatenation of the two latent representations h_{taxi} and h_{bike} is defined as below.

$$\mathcal{H}(d,:) = h_{bike}(d,:), d = 0, 1, ..., D_1 \tag{7}$$

$$\mathcal{H}(n+d,:) = h_{taxi}(d,:), d = 0, 1, ..., D_2 \tag{8}$$

where D_1 and D_2 denote the dimensionality of h_{taxi} and h_{bike}, respectively, and $\mathcal{H} \in \mathcal{R}^{(D_1+D_2)\times N}$.

Model Scalability. In this paper, we utilize bike and taxi flows as the illustration, as they are widely used in road networks and previous studies [8, 19, 21, 26, 32, 33]. However, it is straightforward to integrate other traffic flows (e.g., walk flow), as MDTP is flexible and scalable to support multiple trajectory flows. In the first phase of MDTP, there are multiple networks where each network enables capturing the spatio-temporal features from each type of traffic flow, e.g., bike flow, taxi flow, or other transportation flows. In the second phase of MDTP, two bridging methods are designed to merge the spatio-temporal correlations among multiple types of moving objects (e.g., bikes and taxis in the paper) in order to improve the prediction performance. Note that the two bridging methods can support multiple traffic flows, as they are matrix operations without any limitation on the number of flow categories.

4.4 Overall Loss Function

Finally, we present the loss function of MDTP, which considers *in* and *out* volumes of each node for different types of moving objects.

$$\mathcal{L} = \sum_{i=1}^{N}(\left|y_{i,in}^{type} - \hat{y}_{i,in}^{type}\right| + \left|y_{i,out}^{type} - \hat{y}_{i,out}^{type}\right|) \tag{9}$$

where $\hat{y}_{i,in}$ denotes the predicted *in* volume of node n_i, and $\hat{y}_{i,out}$ represents the predicted *out* volume of node n_i. Here, *type* stands for different moving objects (e.g., taxi or bike), and y denotes the actual value of prediction (i.e., ground-truth). The detailed training process of our method is depicted in Algorithm 1.

Algorithm 1: Multi-source Deep Traffic Prediction

```
  Input: historical $S$ discretized temporal traffic status
         $\mathcal{G} = \{G_{T-S+1}, G_{T-S+2}, ..., G_T\}$, maximum number of
         training iterations $MaxIter$
  Output: future traffic volume $X_{T+1}$ at $T + 1$
1 initialization: historical $(\mathcal{X}, \mathcal{A}) \leftarrow \mathcal{G}, and f_\theta = (\theta_G; \theta_L; \theta_B)$
2 foreach $iter \in \{0, 1, ..., MaxIter\}$ do
3     $\mathcal{Z}^b$ = GCN $(\theta_G^b; \mathcal{X}^b; \mathcal{A}^b)$, $h^b$ = LSTM $(\theta_L; \mathcal{Z}^b)$
4     $\mathcal{Z}^t$ = GCN $(\theta_G^t; \mathcal{X}^t; \mathcal{A}^t)$, $h^t$ = LSTM $(\theta_L^t; \mathcal{Z}^t)$
5     $X_{T+1}^{b,t}$ = Bridge $(\theta_B; h^b; h^t)$
6     update $f_\theta$ based on Equation 9
7 return $X_{T+1}^{b,t}$
```

5 EXPERIMENTS

In this section, we first present experimental settings and then evaluate MDTP, using two real trajectory data sets.

5.1 Experimental Settings

Datasets. We verify MDTP using two real datasets from New York (NYC) and Chicago (CHI). Each dataset contains bike and taxi flows, i.e., NYC-Taxi[1], NYC-Bike[2], CHI-Taxi[3], and CHI-Bike[4].

- **NYC-Taxi.** This dataset contains 53,901,033 trip records of NYC in 2016 (from 01/01/2016 to 06/30/2016).
- **NYC-Bike.** This dataset contains 5,675,719 trip records of NYC in 2016 (from 01/01/2016 to 06/30/2016).
- **CHI-Taxi.** This dataset contains 956,972 trip records of CHI in 2016 (from 03/01/2016 to 04/30/2016).
- **CHI-Bike.** This dataset contains 336,075 trip records of CHI in 2016 (from 03/01/2016 to 04/30/2016).

We partition NYC into 3 sub-datasets for temporal range evaluation, i.e., the sub-dataset of 1^{st} and 2^{nd} months contains 17,411,420 taxi trips and 1,070,345 bike trips, the sub-dataset of 3^{nd} and 4^{th} months contains 18,760,664 taxi trips and 1,933,028 bike trips, and the sub-dataset of 5^{th} and 6^{th} months contains 17,728,949 taxi trips and 2,672,346 bike trips. Note that, the last sub-dataset of 5^{th} and 6^{th} months is the default dataset. Each trip includes: pick-up time/location, drop-off time/location. By the way, it is not easy to get satisfied datasets for multi-source traffic prediction, as they need to contain multiple transportation flows, which are required to have overlaps in both spatial and temporal dimensions.

Baselines. We compare MDTP with existing widely used time series methods (i.e., HA and ARIMA), popular machine learning methods (i.e., XGBoost), and the state-of-the-art neural-network-based approaches (i.e., ST-ResNet and STDN). Specifically, the compared baselines are: (i) **HA [9]**, a classic method that predicts in volume and out volume of a region using the average value of historical traffic data in the time periods; (ii) **ARIMA [15]**, a classical time-series model that combines AR (auto-regression) and MA (moving average) for spatio-temporal data prediction analysis; (iii) **XGBoost [3]**, a scalable machine learning algorithm for tree boosting in prediction studies; (iv) **ConvL [17]**, a neural-network-based method that combines convolution network and LSTM model to capture both spatial and temporal features for traffic prediction; (v) **ST [32]**, a CNN-based prediction model that offers the state-of-the-art results for crowd volume prediction; and (vi) **STDN [26]**, a Spatio-Temporal Dynamic Network that uses dynamic similarity between locations and shifted attention mechanism. Note that, STDN has the state-of-the-art result on traffic prediction, whose corresponding implementation codes are accessible online[5]. For the left baselines, the prediction results on NYC dataset are directly obtained in their original papers.

Preprocessing. For the spatial granularity, we split both the NYC and CHI into 11×21 regions (i.e., nodes) considering the spatial distribution of traffic flows and the city boundary, following the study [26]. For the temporal aspect, we set the time interval as one hour to split the time spans of datasets, which is commonly used in the literature [34]. Specifically, we predict the traffic volume for each hour, based on its traffic volumes in the past 24 hours. However, it is worth mentioning that, MDTP enables flexible spatial and temporal partitioning strategies determined by specific applications. Then, we calculate the *in* and *out* volumes of each region and the volume flow between every pair of regions, and use the min-max normalization to normalize the volumes and flows. Thus, denormalization is performed on the final prediction results, and we use the denormalized value for prediction performance evaluation. During the model testing, we filter the data with volume values less than 10, in order to maintain the same settings as the baselines. It is also a common practice in academic and industrial research. Finally, for each data set, we use 70% as the training set, 10% as the validation set, and 20% as the testing set.

Hyperparameters. In experiments, we set the hyperparameters of MDTP based on the performance of the validation data. In the spatial network, GCN parameters are initialized by default, and the output feature size is 128. In the temporal network, we set the window size as 24 to sample data, which is also the length of LSTM. The layer number of LSTM is 2, and the hidden size of LSTM is 256. We directly adopt Adam optimizer [7] for training. The batch size is set to 32, the gradient threshold is set to 5, and the dropout is set to 0.5. The initial learning rate is 0.001, which is reduced by 10 times every 50 epochs. In addition, we have implemented the early termination mechanism. Specifically, the training is terminated when the default validation set's loss performance does not decline for 20 consecutive epochs. We implement MDTP in Python and Pytorch. All experiments are conducted on a server with Intel Silver 4210R, 2.40GHz CPU, 16-GB RAM, and a GTX-1080 11G GPU.

Evaluation metrics. In the experiments, we use three popular metrics to evaluate the prediction performance following previous traffic prediction studies [20]: (a) (MAE) $MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$, (b) (RMSE) $RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$, and (c) (MAPE) $MAPE = \frac{100\%}{N}\sum_1^N\left|\frac{y_i - \hat{y}_i}{y_i}\right|$. Here, y_i is the ground truth (i.e., actual *in*/*out* volume) of each region n_i, $\hat{y}_i$ is the corresponding predicted value (i.e., predicted *in*/*out* volume) of each region n_i, and N is the region numbers. The more *MAE*, *RMSE*, and *MAPE* close to 0, the higher prediction accuracy means.

[1]https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
[2]https://s3.amazonaws.com/tripdata/index.html
[3]https://data.cityofchicago.org/Transportation/Taxi-Trips-2016/bk5j-9eu2
[4]https://data.cityofchicago.org/Transportation/Divvy-Trips/fg6s-gzvg

[5]https://github.com/tangxianfeng/STDN

Table 1: Performance Comparison on NYC

NYC	Methods	*Out* Volume			*In* Volume		
		MAE	*RMSE*	*MAPE*	*MAE*	*RMSE*	*MAPE*
Taxi	HA	33.56	53.63	25.82%	34.53	62.89	24.10%
	ARIMA	27.78	44.56	24.74%	28.36	51.50	23.84%
	XGBoost	19.87	31.83	21.56%	22.57	41.09	21.32%
	ConvL	21.41	34.34	22.84%	24.50	44.75	23.24%
	ST	20.06	32.02	23.54%	22.38	40.89	24.04%
	STDN	19.10	29.71	19.37%	19.87	36.07	18.63%
	MDTP	17.75	29.32	18.19%	18.71	35.69	17.92%
Bike	HA	10.37	14.55	28.75%	12.32	17.62	27.29%
	ARIMA	9.58	13.44	27.23%	11.95	17.07	26.01%
	XGBoost	7.95	11.16	24.30%	8.65	12.37	22.73%
	ConvL	8.65	12.14	25.93%	9.67	13.86	23.40%
	ST	8.14	11.44	25.89%	9.30	13.30	23.17%
	STDN	7.30	10.28	22.30%	8.56	12.14	20.86%
	MDTP	5.95	8.74	19.01%	6.88	9.95	17.83%

Table 2: Performance Comparison on CHI

CHI	Methods	*Out* Volume			*In* Volume		
		MAE	*RMSE*	*MAPE*	*MAE*	*RMSE*	*MAPE*
Taxi	STDN	8.89	14.71	22.27%	7.35	11.03	21.84%
	MDTP	8.89	14.67	20.67%	6.91	11.03	19.87%
Bike	STDN	5.64	9.17	22.77%	6.27	10.38	24.13%
	MDTP	4.98	7.72	19.44%	5.09	7.99	19.51%

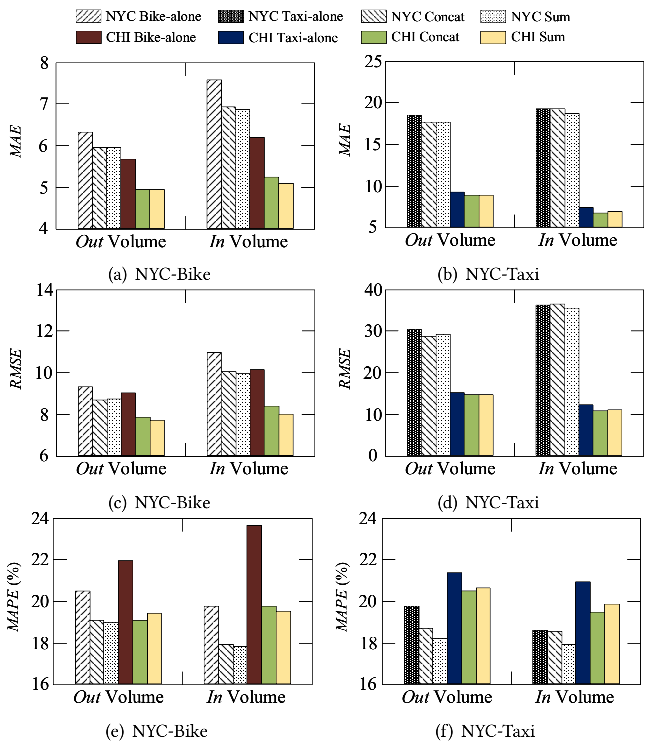


Figure 6: Effect of Multi-source Data

5.2 Prediction Performance Evaluations

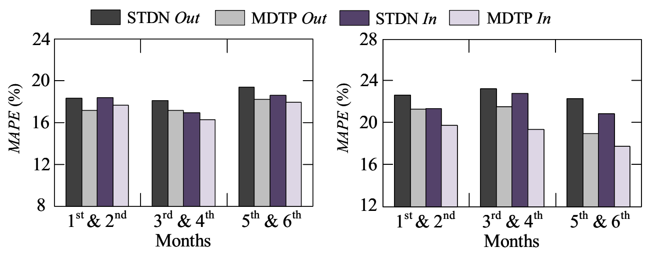
In this subsection, we evaluate the prediction performance of MDTP compared with baselines on NYC and CHI dataset in terms of *MAE*, *RMSE*, and *MAPE*. Table 1 shows the corresponding results on NYC. The first observation is that the traditional time-series methods (i.e., HA and ARIMA) perform the worst since they simply treat historical traffic data as time-series data and ignore the spatial dependency of traffic network. The second observation is that the machine-learning-based XGBoost performs better than HA and ARIMA, as XGBoost considers spatial correlations of traffic. Furthermore, the neural-network-based methods (i.e., ST-ResNet, STDN, and our MDTP) perform the best. The reason is that deep-neural-networks are able to capture the complex nonlinear temporal dependencies and dynamic spatial relationships. Last but not the least, MDTP significantly outperforms all competing baselines by achieving the lowest *MAE*, *RMSE*, and *MAPE* on both NYC-Taxi and NYC-Bike datasets. This is because, MDTP further considers the correlations between multiple types of moving objects (i.e., bike and taxi), and it can learn more spatio-temporal features of the complex traffic network to improve the prediction performance. Table 2 shows the corresponding results on CHI dataset. Here, only STDN is used for comparision, as STDN achieves the best performance among baselines. As observed, MDTP also performs better than STDN. This is because STDN can only utilize a single traffic flow for traffic prediction at a time and ignore the dependence among multiple kinds of moving objects. In contrast, MDTP considers the correlations between bike and taxi flows to improve the prediction accuracy.

5.3 Model Performance Evaluations

We then give model evaluations in terms of multi-source effect, temporal range effect, model scalability, and model efficiency.

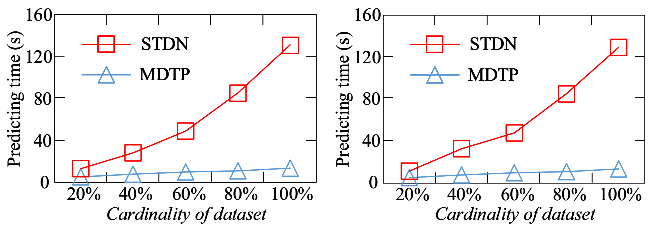
Multi-source Effect. We first evaluate multi-source effect on the prediction performance on NYC and CHI datasets, in order to validate that our multi-source based MDTP method can improve the prediction performance over single-source based approaches. Figure 6 plots the results, where Concat and Sum are the two multi-source bridging mechanisms of MDTP that considering both bike and taxi flows for traffic prediction, while Bike-alone and Taxi-alone mean that MDTP only considers bike or taxi flow at the same time. As expected, the MDTP methods (i.e., Sum and Concat) significantly improve the prediction performance on both NYC and CHI datasets compared with single-source based prediction approaches (i.e., Bike-alone and Taxi-alone), which further demonstrates the effectiveness of MDTP. Another observation is that Sum performs slightly better than Contact on NYC while Contact performs slightly better than Sum on CHI in terms of *MAPE* metric. This is because the volume difference between bike and taxi flows is big in NYC but small in CHI. In the case, Contact is based on matrix multiplication operation, which is more suitable for traffic flow matrices without big differences. In contrast, Sum is based on matrix add operation, which is more general for different situations.

Temporal Range Effect. We proceed to study the model performance under different temporal ranges. Specifically, we divide the NYC dataset into 3 sub-datasets with the details provided in Section 5.1, and verify the prediction performance of MDTP compared with STDN using the sub-datasets. Figure 7 illustrates the corresponding results. Here, only *MAPE* is employed due to the space limitation and similar performance shown by *MAE* and *RMSE*. As observed, MDTP shows stable and superior prediction performance under different temporal ranges, which demonstrates the proposed MDTP is not sensitive to time-varying.



(a) Taxi (b) Bike

Figure 7: Effect of Temporal Range



(a) Taxi (b) Bike

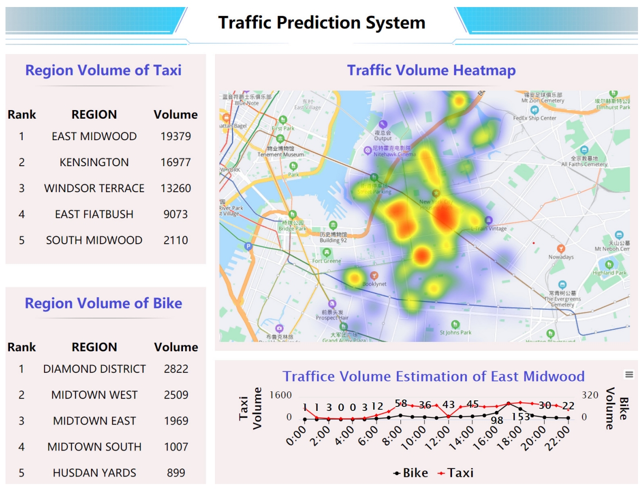
Figure 8: Scalability Performance vs. *cardinality*

Table 3: Efficiency Evaluation

Datasets	**Methods**	**Training (s/epoch)**		**Predicting (s)**	
		Bike	**Taxi**	**Bike**	**Taxi**
NYC	STDN	591	368	27.50	27.97
	MDTP	188.49		9.21	
CHI	STDN	495	430	33.20	24.90
	MDTP	184.24		10.00	

Model Scalability. In the sequel, we explore the scalability performance of MDTP compared with STDN, using the NYC-Taxi and NYC-Bike datasets that contain trips in 5 months. Figure 8 depicts the experimental results, where 20% denotes a one-month dataset, 40% is a two-months dataset (i.e., our default dataset), 60% represents a three-months dataset, 80% is a four-months dataset, and 100% denotes a five-months dataset. The first observation is that the prediction time increases with the growth of cardinality. This is because, 20% of each dataset is used for testing, and the total prediction time ascends as the testing data size grows. Second, MDTP has significant prediction performance improvement over STDN. Moreover, the prediction performance of MDTP is more stable with the growth of cardinality, while STDN shows rapid increase in the predicting time. The reason is that STDN relies on multiple serial LSTMs which cause performance degradation especially for larger datasets. In contrast, MDTP utilizes dynamic flows to capture the traffic dependence across different times, and thus, only one LSTM is enough. Hence, MDTP achieves traffic prediction within 10 seconds. However, STDN takes beyond 2 minutes for per traffic prediction when the cardinality equals to 100%, which is not acceptable in real-life scenarios. Overall, our MDTP model shows good potential scalability to support large-scale traffic prediction.

Model Efficiency. Finally, we investigate the model efficiency considering both training and predicting phases. Table 3 depicts the corresponding results. The first observation is that MDTP outperforms STDN in both training and predicting phases. In the training phase, MDTP runs up to three times faster than STDN on NYC, and two times faster than STDN on CHI. In the predicting (i.e., testing) phase, we measure the total running time of each method on the validation data. As observed, MDTP achieves up to 300% performance improvement on NYC, and at least 200% performance improvement on CHI, compared with STDN. In addition, MDTP enables predicting bike volume and taxi volume at the same time, which further boosts the prediction efficiency.



Traffic Prediction System

Region Volume of Taxi

Rank	REGION	Volume
1	EAST MIDWOOD	19379
2	KENSINGTON	16977
3	WINDSOR TERRACE	13260
4	EAST FIATBUSH	9073
5	SOUTH MIDWOOD	2110

Region Volume of Bike

Rank	REGION	Volume
1	DIAMOND DISTRICT	2822
2	MIDTOWN WEST	2509
3	MIDTOWN EAST	1969
4	MIDTOWN SOUTH	1007
5	HUSDAN YARDS	899

Traffic Volume Heatmap

Traffice Volume Estimation of East Midwood

Figure 9: The Illustration of MDTP$^+$ System

6 DEMONSTRATION

We also develop a cross-platform web-based system MDTP$^+$ by using taxi and bike datasets in NYC, which can be used for traffic monitoring. The interface is shown in Figure 9, where users can interact with MDTP$^+$ in the following scenarios. (i) Traffic Volume Visualization. We use a heatmap to illustrate the predicted volume of each region, as depicted in the upper right corner of Figure 9. The color variation visualizes the congestion or smoothness of traffic conditions of regions. (ii) Traffic Volume Statistics. When a certain region is chosen, a traffic volume trend-line for the last 24 hours is plotted in the lower right corner of Figure 9. In addition, users can easily obtain the exact volume by hovering the mouse over a certain time interval. (iii) Crowded Region Ranking. As shown in the left part of Figure 9, MDTP$^+$ also offers period traffic statistics to show the number of taxis/bikes in different periods (e.g., different hours in one day) and regions (e.g., East Midwood in NYC) for traffic analysis. By default, we demonstrate the top five predicted busy regions at the next time interval to attract users' attention.

7 CONCLUSIONS

In this paper, we propose a novel multi-source deep traffic prediction framework MDTP to capture the correlations between multiple types of moving objects in the dynamic spatio-temporal traffic network. Extensive experiments are conducted using two real data sets (e.g., NYC-Taxi/Bike and CHI-Taxi/Bike), which confirm that MDTP achieves better prediction accuracy and efficiency compared with the state-of-the-art methods. In the future, it is of interest to integrate more than two types of traffic flows in MDTP.

ACKNOWLEDGMENTS

This work was supported by the NSFC under Grant No. 62025206 and 61972338. Yunjun Gao is the corresponding author of the work.

REFERENCES

[1] Yasunori Akagi, Takuya Nishimura, Takeshi Kurashima, and Hiroyuki Toda. 2018. A Fast and Accurate Method for Estimating People Flow from Spatiotemporal Population Data. In *IJCAI*. 3293–3300.

[2] Suresh Chavhan and Pallapa Venkataram. 2020. Prediction based traffic management in a metropolitan area. *TTE* 7, 4 (2020), 447–466.

[3] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *SIGKDD*. 785–794.

[4] Zhiyong Cui, Ruimin Ke, Ziyuan Pu, and Yinhai Wang. 2020. Stacked Bidirectional and Unidirectional LSTM Recurrent Neural Network for Forecasting Network-wide Traffic State with Missing Values. *arXiv preprint arXiv:2005.11627* (2020).

[5] Weiwei Guo, Huiji Gao, Jun Shi, Bo Long, Liang Zhang, Bee-Chung Chen, and Deepak Agarwal. 2019. Deep Natural Language Processing for Search and Recommender Systems. In *SIGKDD*. 3199–3200.

[6] Jilin Hu, Chenjuan Guo, Bin Yang, and Christian S Jensen. 2019. Stochastic weight completion for road networks using graph convolutional networks. In *ICDE*. 1274–1285.

[7] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[8] Ziqian Lin, Jie Feng, Ziyang Lu, Yong Li, and Depeng Jin. 2019. Deepstn+: Context-aware spatial-temporal neural network for crowd flow prediction in metropolis. In *AAAI*. 1020–1027.

[9] Marco Lippi, Matteo Bertini, and Paolo Frasconi. 2013. Short-Term Traffic Flow Forecasting: An Experimental Comparison of Time-Series Analysis and Supervised Learning. *TIST* 14, 2 (2013), 871–882.

[10] Marco Lippi, Matteo Bertini, and Paolo Frasconi. 2013. Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning. *TITS* 14, 2 (2013), 871–882.

[11] Zheng Liu, Lei Chen, and Yongxin Tong. 2018. Realtime traffic speed estimation with sparse crowdsourced data. In *ICDE*. 329–340.

[12] Zhongjian Lv, Jiajie Xu, Kai Zheng, Hongzhi Yin, Pengpeng Zhao, and Xiaofang Zhou. 2018. A deep learning model for traffic prediction. In *IJCAI*. 3470–3476.

[13] Xiaolei Ma, Jiyu Zhang, Bowen Du, Chuan Ding, and Leilei Sun. 2018. Parallel architecture of convolutional bi-directional LSTM neural networks for network-wide metro ridership prediction. *TITS* 20, 6 (2018), 2278–2288.

[14] Yuexin Ma, Xinge Zhu, Sibo Zhang, Ruigang Yang, Wenping Wang, and Dinesh Manocha. 2019. Trafficpredict: Trajectory prediction for heterogeneous traffic-agents. In *AAAI*. 6120–6127.

[15] Luís Moreira-Matias, João Gama, Michel Ferreira, João Mendes-Moreira, and Luís Damas. 2013. Predicting Taxi-Passenger Demand Using Streaming Data. *TIST* 14, 3 (2013), 1393–1402.

[16] Frank Seide and Amit Agarwal. 2016. CNTK: Microsoft's open-source deep-learning toolkit. In *SIGKDD*. 2135–2135.

[17] Xingjian Shi, 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 *NIPS*. 802–810.

[18] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *CVPR*. 2818–2826.

[19] Xianfeng Tang, Boqing Gong, Yanwei Yu, Huaxiu Yao, Yandong Li, Haiyong Xie, and Xiaoyu Wang. 2019. Joint modeling of dense and incomplete trajectories for citywide traffic volume inference. In *WWW*. 1806–1817.

[20] David Alexander Tedjopurnomo, Zhifeng Bao, Baihua Zheng, Farhana Choudhury, and AK Qin. 2020. A Survey on Modern Deep Neural Network for Traffic Prediction: Trends, Methods and Challenges. *TKDE* (2020).

[21] Yuandong Wang, Hongzhi Yin, Hongxu Chen, Tianyu Wo, Jie Xu, and Kai Zheng. 2019. Origin-destination matrix prediction via graph convolution: a new perspective of passenger demand modeling. In *SIGKDD*. 1227–1235.

[22] Shiwen Wu, Yuanxing Zhang, Chengliang Gao, Kaigui Bian, and Bin Cui. 2020. GARG: Anonymous Recommendation of Point-of-Interest in Mobile Networks by Graph Convolution Network. *DSE* 5, 4 (2020), 433–447.

[23] Yuankai Wu, Huachun Tan, Lingqiao Qin, Bin Ran, and Zhuxi Jiang. 2018. A hybrid deep learning based traffic flow prediction method and its understanding. *Transportation Research Part C: Emerging Technologies* 90 (2018), 166–180.

[24] Jianhua Xiao, Zhu Xiao, Dong Wang, Jing Bai, Vincent Havyarimana, and Fanzi Zeng. 2019. Short-term traffic volume prediction by ensemble learning in concept drifting environments. *KBS* 164 (2019), 213–225.

[25] Peng Xie, Tianrui Li, Jia Liu, Shengdong Du, Xin Yang, and Junbo Zhang. 2020. Urban flow prediction from spatiotemporal data using machine learning: A survey. *Information Fusion* 59 (2020), 1–12.

[26] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. 2019. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In *AAAI*. 5668–5675.

[27] 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. *arXiv preprint arXiv:1802.08714* (2018).

[28] Jiexia Ye, Juanjuan Zhao, Kejiang Ye, and Chengzhong Xu. 2020. How to Build a Graph-Based Deep Learning Architecture in Traffic Domain: A Survey. *arXiv preprint arXiv:2005.11691* (2020).

[29] Bin Yu, Xiaolin Song, Feng Guan, Zhiming Yang, and Baozhen Yao. 2016. k-Nearest neighbor model for multiple-time-step prediction of short-term traffic condition. *Journal of Transportation Engineering* 142, 6 (2016), 04016018.

[30] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875* (2017).

[31] Zhuoning Yuan, Xun Zhou, and Tianbao Yang. 2018. Hetero-convlstm: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data. In *KDD*. 984–992.

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

[33] Junbo Zhang, Yu Zheng, Junkai Sun, and Dekang Qi. 2019. Flow prediction in spatio-temporal networks based on multitask deep learning. *TKDE* 32, 3 (2019), 468–478.

[34] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. 2019. T-gcn: A temporal graph convolutional network for traffic prediction. *TITS* 21, 9 (2019), 3848–3858.

[35] Yu Zheng. 2015. Trajectory data mining: an overview. *TIST* 6, 3 (2015), 1–41.

[36] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. 2014. Urban computing: concepts, methodologies, and applications. *TIST* 5, 3 (2014), 1–55.

[37] Zheng Zhu, Bo Peng, Chenfeng Xiong, and Lei Zhang. 2016. Short-term traffic flow prediction with linear conditional Gaussian Bayesian network. *Journal of Advanced Transportation* 50, 6 (2016), 1111–1123.