MS-Net: Multi-Source Spatio-Temporal Network for Traffic Flow Prediction

Shen Fang, Véronique Prinet, *Member, IEEE*, Jianlong Chang, Michael Werman, *Member, IEEE*, Chunxia Zhang, Shiming Xiang, *Member, IEEE*, and Chunhong Pan, *Member, IEEE*

***Abstract*— Predicting urban traffic flow is a challenging task, due to the complicated spatio-temporal dependencies on traffic networks. Urban traffic flow usually has both short-term neighboring and long-term periodic temporal dependencies. It is also noticed that the spatial correlations over different traffic nodes are both local and non-local. What's more, the traffic flow is affected by various external factors. To capture the non-local spatial correlations, we propose a Dilated Attentional Graph Convolution (DAGC). The DAGC utilizes a dilated graph convolution kernel to expand the nodes' receptive field and exploit multi-order neighborhood. Technically, the lower-order neighborhood corresponds to local spatial dependencies, while the higher-order neighborhood corresponds to non-local spatial dependencies between nodes. Based on DAGC, a Multi-Source Spatio-Temporal Network (MS-Net) is designed, which suffices to integrate long-range historical traffic data as well as multi-modal external information. MS-Net consists of four components: a spatial feature extraction module, a temporal feature fusion module, an external factors embedding module, and a multi-source data fusion module. Extensive experiments on three real traffic datasets demonstrates that the proposed model performs well on both the public transportation networks, road networks, and can handle large-scale traffic networks in particular the Beijing bus network which has more than 4,000 traffic nodes.**

***Index Terms*— Graph convolution, deep attention mechanism, traffic network, traffic flow prediction, artificial intelligence, deep learning.**

I. Introduction

DUE to urbanization and growth of public travel demand, urban transportation system is under tremendous pressure. Traffic flow prediction is viewed as a critical component of Intelligent Transportation Systems (ITS) [1], providing significant value in alleviating traffic congestion [2], [3], reducing traffic accident risk [4]–[6], improving traffic control efficiency and enhancing the dispatch of resources in a timely manner [7]–[9]. It could also be adopted to recommend more convenient routes for car drivers, providing information to help residents avoid crowded roads when scheduling trips [10].

Early work regarded traffic forecasting was based on time series analysis [11]. Technically, most methods adopted classical time series models [2], [12], or shallow machine learning algorithms [13], [14]. However, these conventional approaches were restricted to few traffic nodes [12], or traffic networks with a simple topology, such as several sequential nodes on urban freeway [14]. Moreover, prediction accuracy of the models was not satisfactory due to the scarcity of traffic data available for training the models [15].

Recently, inspired by the impressive performance over the past decade of deep learning (DL) in various tasks [16]–[18], and an increasing amount of available traffic data, obtained from vehicles or roads now equipped with sensors [19], DL based traffic prediction models have attracted much attention [20]. Compared with traditional methods, DL based prediction models exhibit higher model capacity in learning the complicated spatio-temporal features on traffic networks [21]. As a result, traffic flow prediction is not restricted to a few traffic nodes, but can be applied to the entire network [7]. Unfortunately, most of the existing DL models do not fully handle the unique spatio-temporal patterns on traffic networks and still fall short in practice.

First, urban traffic flow usually possesses both short-term neighboring (one or two hours) and long-term periodic dependencies (daily, weekly, or monthly) [15]. Therefore, the feature fusion strategy needs to be carefully designed in order to fully exploit these two temporal patterns. **Second**, it has been found that both local and non-local spatial correlations exist on traffic networks [22], illustrated in Figure 1. However, most current works mainly only focus on the local spatial dependencies while neglecting the non-local components. **Third**, various external factors from different sources can affect traffic flow, which should be comprehensively considered to improve the prediction, such as weather, holidays, and the distribution of Points of Interests (PoIs). For instance, traffic patterns on holidays are different from those on weekdays [23]. Extreme weather, such as heavy rain or fog, impacts the traffic flow in a short term [24], [25]. Spatial distribution of PoIs, which reflects the urban functions of different regions, can also have an impact [26]. For instance, traffic flow in residential and office areas usually show different patterns [27].

Manuscript received March 29, 2020; revised January 5, 2021; accepted March 11, 2021. This work was supported in part by the Major Project for New Generation of AI under Grant 2018AAA0100400 and in part by the National Natural Science Foundation of China under Grant 91646207, Grant 62072039, Grant 62076242, and Grant 61976208. The Associate Editor for this article was E. I. Vlahogianni. *(Corresponding author: Shiming Xiang.)*

Shen Fang and Shiming Xiang are with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 101408, China (e-mail: shen.fang@nlpr.ia.ac.cn; smxiang@nlpr.ia.ac.cn).

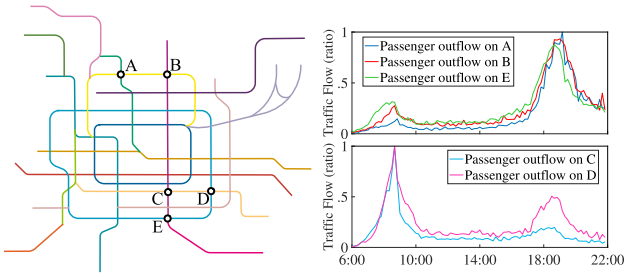
Véronique Prinet, Jianlong Chang, and Chunhong Pan are with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: vprinet@gmail.com; jianlong.chang@nlpr.ia.ac.cn; chpan@nlpr.ia.ac.cn).

Michael Werman is with The Rachel and Selim Benin School of Computer Science and Engineering, The Hebrew University of Jerusalem, Jerusalem 91904, Israel (e-mail: michael.werman@mail.huji.ac.il).

Chunxia Zhang is with the School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China (e-mail: cxzhang@bit.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TITS.2021.3067024, provided by the authors.

Digital Object Identifier 10.1109/TITS.2021.3067024



(a) Beijing subway network (b) Traffic flow on five nodes

Fig. 1. Passenger flow of five selected nodes on Beijing subway network. The traffic flow of A, B, and E could be highly correlated while E is far from A and B. Nodes C, D, and E are adjacent but the traffic flow of E and that of C, D could express different patterns.

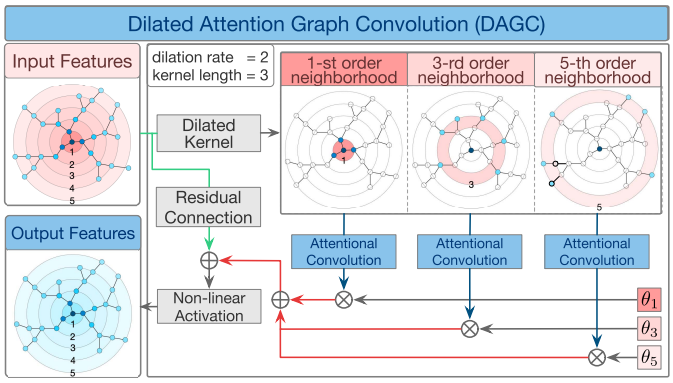


Fig. 2. Structure of DAGC. DAGC first expands the receptive field through a dilated kernel to consider multi-order neighborhoods. The lower-order neighborhood and the higher-order neighborhood correspond to the local and non-local spatial dependencies, respectively. Convolution results of different neighborhoods are aggregated by learnable parameters θ_1, θ_3, and θ_5.

In order to capture local and non-local spatial correlations, we propose a Dilated Attentional Graph Convolution operator (**DAGC**). As illustrated in Figure 2, our DAGC makes use of a dilated graph convolution kernel to expand the nodes' receptive field and exploit multi-order neighborhood. The lower-order neighborhood corresponds to local spatial dependencies, while the higher-order neighborhood corresponds to non-local spatial dependencies between nodes. Multiple layers of DAGC can be stacked. Stacking layers not only enhances model capacity but also expands the receptive field, taking into account the non-local correlations between traffic nodes that are far apart.

By taking advantage of the DAGC, a Multi-Source Spatio-Temporal Network (**MS-Net**) is proposed, which is illustrated in Figure 3. The MS-Net consists of four main components: the Spatial Feature Extraction Module (**SFM**), the Temporal Feature Fusion Module (**TFM**), the External Factors Embedding Module (**EFM**), and the Multi-Source Data Fusion Module (**MFM**). The SFM extracts local and non-local spatial features by stacking multi-layers of DAGC. In TFM, the spatio-temporal features are obtained by a DAGC operation with the concatenated spatial features from different historic days. The EFM embeds various external data, of discrete and continuous nature. Finally, the external data features and traffic data features are integrated into the MFM on spatial and temporal dimensions to further improve predictions.

The main contributions of the MS-Net are as follows:

- We develop a novel dilated attentional graph convolution. It enables one to capture local and non-local spatial features, accounting for correlations between far apart nodes.
- Traffic data from different historical days are integrated, which endows MS-Net with the capability of learning from both short-term neighboring and long-term periodic temporal dependencies.
- Various external factors from different sources, including both discrete and continuous nature, as well as data variously distributed on temporal and spatial dimensions, can be integrated into our overall model.
- Extensive experiments prove that our model is competitive w.r.t. state-of-the-art approaches on both the public transportation network and the road network.

A preliminary version of this work appears in [22]. Here we carry forward the idea of learning multi-resolution temporal dependencies and global spatial correlations on traffic networks and substantially improve the previous model. First, our multi-period temporal feature fusion module replaces the temporal convolution. It reduces running time while preserving the capability of the model to simultaneously learn both short-term dynamics and long-term periodic dependencies. Second, in terms of spatial feature extraction, the original global correlated spatial module does not consider the topological information of the traffic network very well. As a result, it is replaced with the proposed DAGC. Furthermore, several external factors (distribution of PoIs, weather information, and holiday notification) are also taken into consideration to further improve performance.

The remaining parts of this paper are organized as follows. In Section II, related works of traffic flow prediction are reviewed. Section III gives mathematical notations and the problem statement of traffic flow prediction. Then, the novel graph convolution DAGC is formally presented in Section IV, and the specific structure of the whole prediction model MS-Net is introduced in Section V and Section VI. The last two sections, Section VII and Section VIII present the experimental results and the conclusion of the paper.

II. Related Work

A. Traffic Flow Prediction

Traditional methods adopt time series models or shallow machine learning models to predict future road traffic flow. The autoregressive integrated moving average (ARIMA) model [28]–[30] and the Kalman filter [31] as well as their variants [32]–[35], are widely employed in traffic flow prediction task. However, due to the limited model capacity, these conventional methods handle each traffic observation node individually [11], or a few sequential nodes on one urban major road [14] where the topology of the traffic network and the spatial correlations are simple and intuitive.

Recently, as a large amount of traffic data has become available, deep learning (DL) based traffic prediction models have been developed. Compared with previous methods, DL models exhibit higher model capacity and better capability in learning the spatio-temporal features on traffic networks [15].

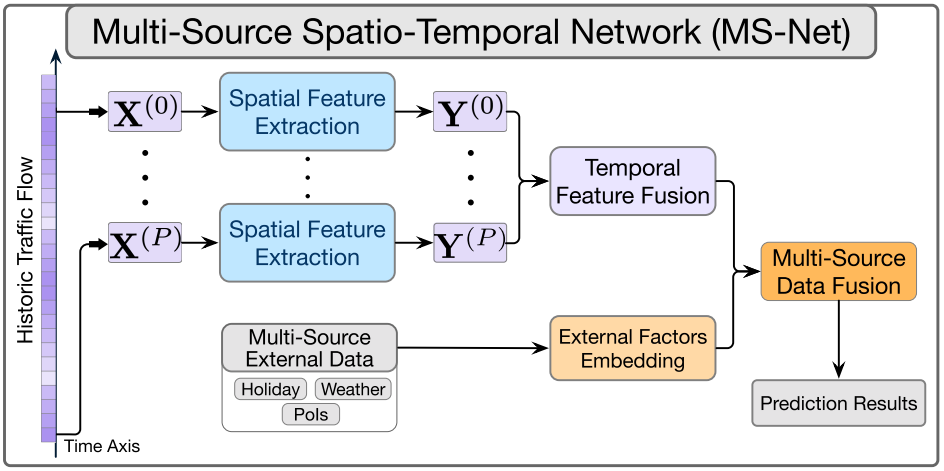


Fig. 3. Structure of MS-Net. MS-Net consists of four components: spatial feature extraction, temporal feature fusion, external factors embedding, and multi-source data fusion. The spatial feature extraction module is composed of stacked DAGCs. Features of different historical days are then concatenated and fed into the temporal feature fusion module. External factors are embedded and integrated with traffic spatio-temporal features to predict urban traffic flow.

The stacked auto-encoders (SAEs) [36] are adopted to predict traffic flow on multiple nodes. While the model in [36] can learn spatio-temporal features of traffic data, there is no explicit modeling of spatial correlations, and the chronological order of historical data is not maintained.

To take into account the temporal dependencies of traffic data, several approaches make use of a long short term memory network (LSTM) [37]–[40]. LSTM and SAEs are combined to predict urban traffic data under abnormal conditions (*e.g.*, extreme congestion or traffic accidents) [40]. Different from the LSTM based models, convolutional neural networks (CNNs) are first adopted to predict urban crowd flow in [15]. Specifically, the city map is divided into non-overlapping grids to form a pseudo image, where the pixel values are determined by the crowd flow through each grid at each time interval. Then CNNs equipped with residual connections [41] are employed to extract the spatial features. One of the main contributions of [15] is to convert traffic data into image-like data in order to utilize image based CNNs in the field of urban computing.

Inspired by [15], subsequent works have adopted similar data transformations [21], some of which combines LSTM and the attention mechanism [27], [42]–[44]. LSTMs are either embedded with the CNNs to construct ConvLSTM module [42], [43], or embedded with attention mechanism to take into account historical data at different times [27], [44]. While the above CNNs based models can take advantage of a large amount of traffic data, the prediction results of these methods are not of high granularity: The strategy of converting traffic data into images may distort the spatial and topological correlations between traffic nodes.

B. Deep Learning on Graph Structured Data

To achieve high-granularity traffic prediction results, traffic prediction methods based on deep graph convolution models have been recently favored [45]. Formally, graph convolution aims to perform similar convolutional operations like traditional CNNs where the model aggregates features of neighboring nodes [46]. However, due to the irregular structure of graph data, it is not intuitive to define a convolutional kernel with a fixed shape and size similar to the ordinary CNNs [47]. There are two dominant approaches to solve this problem: spectral approach [47]–[53] and spatial approach [54]–[59].

The spectral approach utilizes the Convolution Theorem to transform nodes' features into spectral domain, where multiplication on spectral domain is equivalent to the convolution on spatial domain. Formally, a graph can be represented by $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{v_i\}, (i = 1, \cdots, N)$ is the set of nodes, and $\mathcal{E} = \{e_{i,j}\}$ is the edges between nodes v_i and v_j. Let $x_i \in \mathbb{R}$ be the v_i's feature, $\boldsymbol{x} \in \mathbb{R}^N$ be the feature vector on all N nodes, and $\mathbf{A} \in \mathbb{R}^{N \times N}$ be the adjacent matrix of graph (*i.e.,* $A_{i,j} = 1$ if $e_{i,j} \in \mathcal{E}$ else $A_{i,j} = 0$), the spectral approach operates as follows:

$$\boldsymbol{y} = g_\theta(\mathbf{L})\boldsymbol{x} = g_\theta(\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{T}})\boldsymbol{x} = \mathbf{U}g_\theta(\mathbf{\Lambda})\mathbf{U}^{\mathrm{T}}\boldsymbol{x}, \tag{1}$$

where $\boldsymbol{y} \in \mathbb{R}^N$ is the convolution result, and $g_\theta(\Lambda) = \mathrm{diag}(\theta)$ is the filter of a diagonal matrix with learnable parameter θ. In Eq. (1), $\mathbf{U} \in \mathbb{R}^{N \times N}$ is the eigenvectors of the normalized Laplacian $\mathbf{L} = \mathbf{I}_N - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}} = \mathbf{U}\Lambda\mathbf{U}^{\mathrm{T}}$, where $\Lambda \in \mathbb{R}^{N \times N}$ are the corresponding eigenvalues, $\mathbf{I}_N$ is the $N \times N$ identity matrix, and $\mathbf{D}$ is the diagonal matrix with $D_{i,i} = \sum_j A_{i,j}$. Since graph laplacians are positive semi-definite, the eigenvectors constitute an orthonormal basis for the Fourier domain [47]. As a result, spectral graph convolution is theoretically guaranteed, but not applicable in practice due to the calculation complexity of graph Laplacian decomposition [49].

On the other hand, spatial graph convolution is directly developed on graph, and is mainly devoted to designing localized convolutional kernels which can be shared to different nodes in the graph. In general, spatial approach operates as follows:

$$y_i = f_\theta(x_i, \{x_j | \forall v_j \in \mathcal{N}(v_i)\}), \tag{2}$$

where $y_i \in \mathbb{R}$ is output feature on node v_i, and θ is the learnable convolutional kernel. In Eq. (2), $\mathcal{N}(v_i) = \{v_j | e_{i,j} \in \mathcal{E}\}$ is the neighborhood of node v_i, and $x_j \in \mathbb{R}$ is the features on v_j which is adjacent to node v_i.

In general, compared with spectral methods, spatial graph convolution is more efficient and flexible in constructing

convolutional kernels and choosing neighborhoods. It has become more and more popular in recent researches [46].

Recently, graph convolution has been applied to traffic flow prediction [25], [45], [60]–[66]. For instance, the DCNN [54] is embedded in the GRU to construct the Diffusion Convolutional Recurrent Neural Network (DCRNN) [60]. An integrated model based on multi priori graphs is proposed to predict urban bike flow [61]. A fully convolutional model [64], which combines ChebNet [49] and one-dimensional temporal convolution, is developed to capture the localized spatio-temporal patterns. Attention mechanism [63], high-order graph convolution [65], and data fusion of different modalities [25] have also been widely adopted to predict urban traffic flow. A meta-learning strategy is employed to extract spatial dependencies between adjacent traffic nodes [66]. However, most existing methods focus on local spatial correlations while ignoring the non-local spatial dependencies on traffic networks.

III. Preliminaries

A. Notations

Traffic Network. The traffic network can be represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of traffic nodes and $\mathcal{E}$ is the edges between these nodes. $e_{ij} \in \mathcal{E}$ indicates that there is a direct path between two traffic nodes v_i and v_j.

Neighborhood of Nodes. The neighborhood of node v_i, $\mathcal{N}(v_i)$, contains all the nodes directly connected to v_i. If $v_j \in \mathcal{N}(v_i)$, there is an edge $e_{ij} \in \mathcal{E}$, and vice versa.

Multi-order Neighborhood of Nodes. The multi-order neighborhood of node v_i refers to a set of nodes, which are not directly connected to v_i, but can be reached through a series of intermediate nodes. Formally, the l-th order neighborhood of node v_i, $\mathcal{N}(v_i, l)$, is recursively defined as follows:

$$\mathcal{N}(v_i, l) = \bigcup_{v_j \in \mathcal{N}(v_i, l-1)} \mathcal{N}(v_j) \setminus \mathcal{N}(v_i, l-2),$$

$\mathcal{N}(v_i, 0) = \{v_i\}$ and $\mathcal{N}(v_i, 1) = \mathcal{N}(v_i)$.

Historical Traffic Flow. Historical traffic flow is divided into two categories, *i.e.*, immediate past traffic flow (short term) and periodic (daily) traffic flow. Suppose there are N traffic nodes and M types of traffic data (typically, inflow and outflow), the m-th type of traffic flow observed on node v_i at time t can be noted as $x_{i,t,m}$. Assuming that there are r_d traffic records at regular interval each day, then the vector $\{x_{i,t-r_d \cdot p-\tau,m} | \tau = 1, 2, \cdots, T\} \in \mathbb{R}^T$ represents the traffic flow captured during τ time steps preceding t at the p-th day in past. The traffic flow on all nodes at the p-th historical day constitutes a tensor $\mathcal{X}^{(p)} \in \mathbb{R}^{N \times T \times M}$. If $p = 0$, $\mathcal{X}^{(0)}$ stands for the short-term neighboring traffic flow; if $p > 0$, $\mathcal{X}^{(p)}$ represents the long-term periodic traffic flow on the p-th day in history.

External factors. We use three external factors: two global of weather[1] and holiday notifications[2]; one local factor, Points of Interests[3] (PoIs) around each traffic node. Weather information includes temperature, wind speed, visibility, precipitation, and weather condition. Holiday notifications indicate whether the recording is acquired during a working day or a holiday. The number of different categories of PoIs around each traffic node is additional features. Mathematically, the weather information and holiday notifications at time t are represented by vectors $\boldsymbol{e}_t^w \in \mathbb{R}^{D^w}$ and $\boldsymbol{e}_t^h \in \mathbb{R}^{D^h}$, respectively, where D^w and D^h stand for the feature dimensions. The PoIs on node v_i is $\boldsymbol{e}_i^p \in \mathbb{R}^{D^p}$. Considering all traffic nodes, the data of PoIs is represented by a matrix $\mathbf{E}^p \in \mathbb{R}^{N \times D^p}$.

[1] https://www.wunderground.com
[2] https://pypi.org/project/chinesecalendar
[3] https://lbs.amap.com

B. Problem Statement

Our aim is to learn a function $\mathcal{F}_\Theta(\cdot)$, which takes as input the traffic network $\mathcal{G}$, the observed historical traffic flow $\mathcal{X}^{(p)}$, $p = 0, 1, \cdots, P-1$, the external factors $\mathbf{E} = \{\boldsymbol{e}_t^w, \boldsymbol{e}_t^h, \mathbf{E}^p\}$ at future time t, and outputs the traffic flow on all nodes at the next time step t. $\mathcal{F}_\Theta(\cdot)$ is defined by its learnable parameter Θ. It can be written as follows:

$$\hat{\mathbf{X}} = \mathcal{F}_\Theta(\{\mathcal{X}^{(p)}\}_{p=0}^{P-1}, \mathbf{E}, \mathcal{G}), \tag{3}$$

where $\hat{\mathbf{X}} \in \mathbb{R}^{N \times M}$ stands for the predicted flow values, of dimension M, estimated at each of the N nodes. For the multi-step prediction, the weather information and holiday notifications are adjusted to the corresponding time interval, and the function $\mathcal{F}_\Theta(\cdot)$ is trained to predict the traffic flow at k-time step ahead.

IV. Dilated Attentional Graph Convolution

The main principle of our Dilated Attentional Graph Convolution (**DAGC**) is to equip the graph convolution operator with a flexible receptive field on spatial domain, so that graph convolution can account for both local and non-local spatial dependencies between nodes.

To effectively aggregate spatial information on the graph, DAGC expands the receptive field through a dilated convolution graph kernel. Specifically, message passing is performed in two stages. Nodes are first passed through graph convolution operators of various receptive field size independently. This operation is coupled with an attention mechanism. The outputs of graph convolution at each order of neighborhood are then aggregated with a weighted summation. The detailed structure of our graph convolution operator is shown in Figure 2.

A. Attentional Graph Convolution

Given $\boldsymbol{x}_i \in \mathbb{R}^C$, a feature of dimension C on node v_i, the attentional graph convolution is defined as follows:

$$\boldsymbol{y}_i = \sum_{v_j \in \mathcal{N}(v_i, l)} f_\psi(\boldsymbol{x}_i, \boldsymbol{x}_j) \cdot \boldsymbol{x}_j \mathbf{W}_a + g(\boldsymbol{x}_i), \tag{4}$$

where $\boldsymbol{y}_i \in \mathbb{R}^D$ is the output features on node v_i, D is the number of output channels, and $\mathcal{N}(v_i, l)$ is the l-th order neighbors around node v_i. $f_\psi(\cdot, \cdot)$ is a learnable correlation function between two nodes. $\mathbf{W}_a \in \mathbb{R}^{C \times D}$ is a learnable projection matrix, and "$+g(\boldsymbol{x}_i)$" denotes the residual connection on the central node v_i. If C is equal to D, then $g(\boldsymbol{x}_i)$ is an identity function, *i.e.*, $g(\boldsymbol{x}_i) = \boldsymbol{x}_i$, otherwise $g(\boldsymbol{x}_i) = \boldsymbol{x}_i \mathbf{W}_g$, where $\mathbf{W}_g \in \mathbb{R}^{C \times D}$ is a learnable matrix. Through the attention mechanism, the value of the convolutional kernel is not fixed

but determined by the correlations between nodes. Therefore, adjacent nodes with stronger spatial correlations are assigned larger weights, which facilitates the information propagation in the graph $\mathcal{G}$.

Generally speaking, the neighborhood size of different nodes is inconsistent, *i.e.*, nodes of the graph do not necessarily have the same degree. If this imbalance is not accounted for, a few nodes with large connectivity might receive more and more messages during the diffusion process. Eventually, the features on these nodes will dominate the features on other nodes having a smaller degree, and thus occupy a dominant position in expressing the patterns of the whole graph. A simple solution is to normalize the attention function when summing over nodes' features in Equation (4):

$$f_\psi(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\exp(\tilde{f}_\psi(\boldsymbol{x}_i, \boldsymbol{x}_j))}{\sum_{v_k \in \mathcal{N}(v_i, l)} \exp(\tilde{f}_\psi(\boldsymbol{x}_i, \boldsymbol{x}_k))}, \tag{5}$$

so that $\sum_{v_j \in \mathcal{N}(v_i, l)} f_\psi(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1$. Here, $\exp(\cdot)$ is an exponential function to ensure the non-negativity. Concretely, we define $\tilde{f}_\psi(\boldsymbol{x}_i, \boldsymbol{x}_j) = h_\psi(\boldsymbol{x}_i)^{\mathrm{T}} \cdot h_\psi(\boldsymbol{x}_j)$, where $h_\psi(\cdot) : \mathbb{R}^C \to \mathbb{R}^F$ is a multilayer perceptron defined by parameters ψ, transforming the node features into a latent space.

Note that the model size so far is determined by the parameters ψ, $\mathbf{W}_a$, and $\mathbf{W}_g$, which are independent of the number of nodes in the graph. This is in contrast with spectral graph convolution approaches, whose parametrization depends on the graph structure itself [47].

B. Higher-Order Attentional Graph Convolution

The attentional graph convolution operator proposed in Equation (4) is defined on one order of neighborhood. We can extend it to a higher-order operator:

$$\boldsymbol{y}'_i = \sum_{l=1}^{L} \theta_l \left(\sum_{v_j \in \mathcal{N}(v_i, l \cdot d)} f_\psi(\boldsymbol{x}_i, \boldsymbol{x}_j) \cdot \boldsymbol{x}_j \mathbf{W}_a \right) + g(\boldsymbol{x}_i), \tag{6}$$

$$\boldsymbol{y}_i = \sigma\left(\boldsymbol{y}'_i\right), \tag{7}$$

where $\mathcal{N}(v_i, l \cdot d)$ represents the $(l \cdot d)$-th order neighborhood of node v_i, d is the dilation rate, $\{\theta_l \in \mathbb{R} | l = 1, 2, \cdots, L\}$ are learnable weighting factors of the l-th hop, and $\sigma(\cdot)$ is the non-linear activation function.

In Equation (6), each order of neighborhood is processed separately and assigned with a learnable parameter θ_l, so that both the static topological information (θ_l for the l-th hop) and the dynamic feature dependencies (adaptive weight factors determined by $f_\psi(\boldsymbol{x}_i, \boldsymbol{x}_j)$) are modeled in the same framework. What's more, the size of the receptive field on spatial domain is given by L and d, two hyper-parameters that can be adjusted.

Hence, compared with most existing methods that only notice the localized features [54]–[58], our DAGC can not only capture the features on adjacent nodes, but also learn spatial dependencies between distant nodes.

V. Spatio-Temporal Feature Extraction

Equipped with the proposed DAGC, our forecasting model MS-Net can learn the spatio-temporal features on traffic flow data, which include two main modules, spatial feature extraction module and temporal feature fusion module.

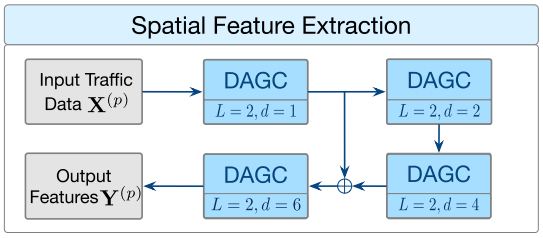


Fig. 4. Spatial feature extraction module.

A. Spatial Feature Extraction

The spatial feature extraction module is designed to extract the local and non-local spatial correlations of traffic data in different historical days. It is composed of multiple stacked layers of DAGC, as shown in Figure 4. Each layer of DAGC has a corresponding dilation rate d and a maximal hop number L. The number of hops L of different layers remains the same, but the dilation rate d gradually increases, so that the module can gradually transit from capturing local spatial dependencies to extracting non-local spatial correlations. In addition, the output feature of the first layer of DAGC is added as a skip connection to the input features of the last layer. It has the effect to not only integrate the local and non-local spatial relationships together, but also alleviate the potential problem of gradient vanishing in multi-layer networks.

The traffic flow at the p-th historical day is denoted by a tensor $\mathcal{X}^{(p)} \in \mathbb{R}^{N \times T \times M}$ and further transformed into a matrix $\mathbf{X}^{(p)} \in \mathbb{R}^{N \times TM}$, where T is the length of the immediate past traffic data vector. There are three reasons for this conversion process. (*i*) Although the chronological order is ignored (temporal information is transformed into the feature dimensions), all historical data is retained. Meanwhile, ignoring the chronological order can accelerate the running time of the model and reduce memory consumption (the graph convolution does not need to be repeated for T times), and consequently make the model applicable to large-scale graph structure. (*ii*) Traffic data recorded at different historical days are selected and processed separately, so the temporal dependencies can still be extracted on a larger scale. (*iii*) Finally, the value of T is usually not large. Formally, the spatial feature extraction result on the p-th historical day is formulated as follows:

$$\mathbf{Y}^{(p)} = \mathcal{H}^{(p)}(\mathbf{X}^{(p)}, \mathcal{G}), \quad p = 0, 1, \cdots, P-1 \tag{8}$$

where $\mathbf{Y}^{(p)} \in \mathbb{R}^{N \times F}$ is the output features, F is the feature dimension, and $\mathcal{H}^{(p)}(\cdot)$ is the spatial feature extraction module corresponding to the traffic data on the p-th day in history. The detailed structure of $\mathcal{H}^{(p)}$ is shown in Figure 4.

B. Temporal Feature Fusion

The spatial features of the historical context can then be integrated altogether to capture the temporal dynamic of the traffic flow. It is pointed out in [25] that temporal features might be fused with a weighted summation:

$$\mathbf{Y} = w^{(0)} \cdot \mathbf{Y}^{(0)} + \cdots + w^{(P-1)} \cdot \mathbf{Y}^{(P-1)}, \tag{9}$$

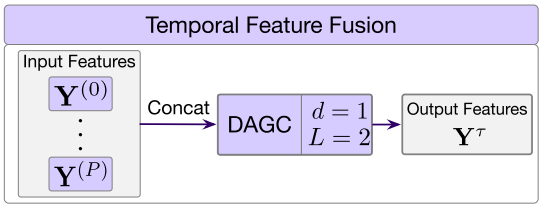


Fig. 5. Temporal feature fusion module.

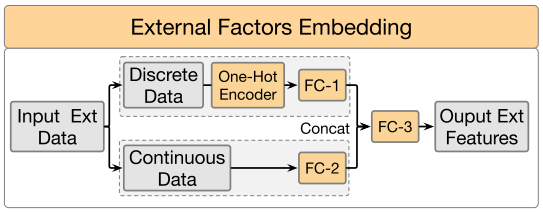


Fig. 6. External factors embedding module.

where $w^{(p)} \in \mathbb{R}$ is a learnable weight parameter. This enables to assign different weights to observations acquired at different days, but applies the same weight to all traffic nodes. It might not be the best strategy. For instance, the traffic flow on some nodes have a relatively stable one-day periodicity, while the periodicity of traffic flow on other nodes is less obvious; it does not seem appropriate to assign the same weight factor to these two kinds of nodes. On the other hand, directly assigning a fusion weight to each traffic node produces too many learnable parameters, which could potentially result in over-fitting.

Instead, we adopt a feature concatenating approach. As illustrated in Figure 5, our temporal fusion module first concatenates the output features of the spatial module along the channel dimension: $\mathbf{Y}^c = [\mathbf{Y}^{(0)}, \mathbf{Y}^{(1)}, \cdots, \mathbf{Y}^{(P-1)}] \in \mathbb{R}^{N \times PF}$. Then a DAGC operator is employed to transform the node features:

$$\mathbf{Y}^{\tau} = \mathcal{H}^{\tau}(\mathbf{Y}^c, \mathcal{G}), \tag{10}$$

where $\mathbf{Y}^{\tau} \in \mathbb{R}^{N \times Q}$ is the Q dimensional output of a one-layer DAGC applied to the N nodes, $\mathcal{H}^{\tau}(\cdot)$. Hence, the spatio-temporal features $\mathbf{Y}^{\tau}$ are processed according to their inherent characteristics, instead of sharing the same summation weight. In the next section, we consider integrating traffic spatio-temporal features with external data.

VI. Data Fusion and Forecasting Model

A. External Factors Embedding

We employ three kinds of external data, weather information $\boldsymbol{e}_t^w \in \mathbb{R}^{D^w}$ and holiday notifications $\boldsymbol{e}_t^h \in \mathbb{R}^{D^h}$ at the prediction time t, as well as the Points of Interests (PoIs) $\mathbf{E}^p \in \mathbb{R}^{N \times D^p}$ around all the traffic nodes.

1) Weather Information: The weather information $\boldsymbol{e}_t^w$ includes both continuous data (*e.g.*, temperature, wind speed) and discrete data such as weather condition (cloudy or rainy). These two types of data cannot be seen as the same, nor can they be directly fed into an embedding module as a whole. Therefore, as shown in Figure 6, external factors are separated into two categories based on their attributes. Discrete data is encoded as a one-hot vector and then embedded with a fully connected (FC) layer, whereas continuous data can be directly fed into a FC layer. Finally, the embedded features of these two types of data are fused through another FC network:

$$\hat{\boldsymbol{e}}_t^w = ext^w(\boldsymbol{e}_t^w), \tag{11}$$

where $\hat{\boldsymbol{e}}_t^w \in \mathbb{R}^{F^w}$ is the embedding result, F^w is the number of output channels, and $ext^w(\cdot)$ denotes the external factors embedding module for processing weather information.

2) Holiday Notifications: The holiday notification $\boldsymbol{e}_t^h$ indicates whether the prediction time is on holiday or workday. It is a discrete categorical value. It is again embedded using a FC block:

$$\hat{\boldsymbol{e}}_t^h = ext^h(\boldsymbol{e}_t^h), \tag{12}$$

where $\hat{\boldsymbol{e}}_t^h \in \mathbb{R}^{F^h}$ is the embedding result, F^h is the number of channel, and $ext^h(\cdot)$ denotes the embedding module for processing holiday notification data.

3) PoIs Distribution: Assuming there are $D^p - 1$ PoIs categories, and the number of d^p-th PoIs around node v_i is denoted by $\tilde{e}_{i,d^p}$. Generally, the number of PoIs around different nodes is usually quite different, so the features are normalized before being fed into the embedding module:

$$e_{i,d^p} = \frac{\tilde{e}_{i,d^p}}{\sum_{j=1}^{N} \tilde{e}_{j,d^p}}, \quad d^p = 1, 2, \cdots, D^p - 1. \tag{13}$$

The vector of $\{e_{i,d^p} | d^p = 1, 2, \cdots, D^p - 1\} \in \mathbb{R}^{D^p - 1}$ constitutes the first $D^p - 1$ elements of the feature $\boldsymbol{e}_i^p$. The last element is reserved for the number of all kinds of PoIs around the node —it has been shown experimentally to have significant correlations with the urban traffic flow [26]. Normalization is again applied to obtain the feature $e_{i,D^p} \in \mathbb{R}$ of the last dimension. Hence, PoIs' feature vector is denoted by $\boldsymbol{e}_i^p = [e_{i,d^p}, e_{i,D^p}] \in \mathbb{R}^{D^p}$ at each node v_i, where $[,]$ is a concatenation. Considering all traffic nodes, the PoIs' features is represented by a matrix $\mathbf{E}^p \in \mathbb{R}^{N \times D^p}$. Now the data embedding can write:

$$\hat{\mathbf{E}}^p = ext^p(\mathbf{E}^p), \tag{14}$$

where $\hat{\mathbf{E}}^p \in \mathbb{R}^{N \times F^p}$ is the embedding result of dimension F^p, and $ext^p(\cdot)$ represents the FC embedding function. Note that the FC layers processing different external factors do not share the same parameters.

B. Multi-Source Data Fusion

Because some of the external factors are local (PoIs) while others are global (weather, holidays), we adopt a two-stage fusion procedure to combine external factors with the traffic spatio-temporal features, which is shown in Figure 7.

In the first stage, the PoIs features $\hat{\mathbf{E}}^p$ are concatenated with the traffic flow features $\mathbf{Y}^{\tau}$ defined in Equation (10). The concatenated feature is then fed into a DAGC module:

$$\mathbf{Y} = \mathcal{H}([\mathbf{Y}^{\tau}, \hat{\mathbf{E}}^p], \mathcal{G}), \tag{15}$$

where $\mathbf{Y} \in \mathbb{R}^{N \times \tilde{Q}}$ is the spatio-temporal output of dimension $\tilde{Q}$, and $\mathcal{H}(\cdot)$ is a one layer of DAGC operator.

In the second stage, the weather features $\hat{\boldsymbol{e}}_t^w$ and holiday notifications $\hat{\boldsymbol{e}}_t^h$ are first expanded into matrices $\hat{\mathbf{E}}_t^w \in \mathbb{R}^{N \times F^w}$,

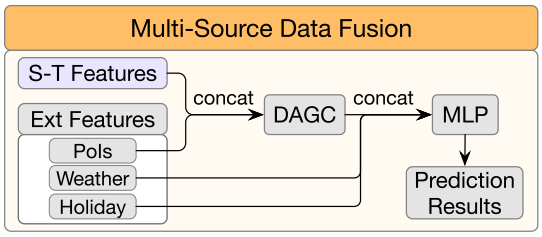


Fig. 7. Multi-source data fusion module.

$\hat{\mathbf{E}}_t^h \in \mathbb{R}^{N \times F^h}$, respectively, by copying the features $\hat{\boldsymbol{e}}_t^w$, $\hat{\boldsymbol{e}}_t^h$ to each row. The two features are then concatenated with the spatio-temporal feature $\mathbf{Y}$ and sent to a multi-layer perceptron $MLP(\cdot)$ to produce the prediction results:

$$\hat{\mathbf{X}} = MLP([\mathbf{Y}, \hat{\mathbf{E}}_t^w, \hat{\mathbf{E}}_t^h]), \tag{16}$$

where $\hat{\mathbf{X}} \in \mathbb{R}^{N \times M}$ is the prediction results, and M is the data dimension defined in Section III-A.

C. Model Summarization

The loss function of Mean Square Error is adopted to train the model:

$$\mathcal{L}(\Theta) = \left\| \tilde{\mathbf{X}} - \hat{\mathbf{X}} \right\|_F^2, \tag{17}$$

where $\tilde{\mathbf{X}} \in \mathbb{R}^{N \times M}$ is the ground-truth at the prediction time t, and $\|\cdot\|_F$ is the Frobenius norm. We now summarize the key principles of our model as follows:

- Spatial features are generated for each of the traffic data vectors of the past and current days (historical events) independently. To this end, we employ a multi-layer DAGC. It enables to extract both local and non-local features on spatial domain.
- Temporal features are obtained by processing altogether the spatial features computed at past and current days with a DAGC operation. This fusion step results in spatio-temporal features.
- Various External factors (both global or local, continuous or discrete) are embedded into feature vectors, then integrated with the traffic spatio-temporal features using again a DAGC operation.
- Using the multi-source spatio-temporal features as input, a multi-layer perception generates predictions at time t.
- The model is trained end-to-end to forecast traffic flow at the next time step.

VII. Experiments

A. Dataset Description

We evaluate the performance of our model on three real-world traffic datasets: Beijing Subway Dataset, Beijing Bus Dataset, and Beijing Taxi trajectories Dataset. The original records of the first two datasets are the transaction records of Beijing Municipal Transportation Card, and that of the third dataset is the taxi GPS trajectories information in Beijing. The meta information of these datasets is given in Table I.

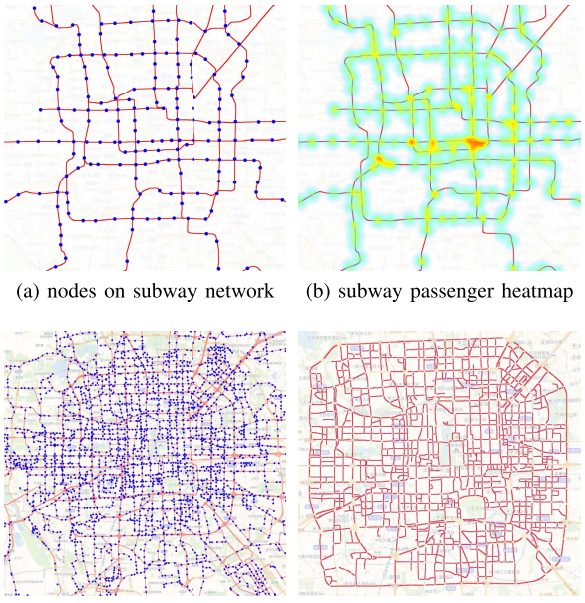
Subway Transaction Dataset: The transaction records of Subway Dataset contains the entering and exiting stations as well as the corresponding entering and exiting timestamps of each travel record. From these records, we can infer the number of passengers entering and exiting each station at each time interval of 10 minutes. There were 18 subway lines and 278 stations until 2016. The nodes are the 278 subway stations, and edges are the subway lines between two adjacent nodes. Since most of the subway lines are closed at night, only the transaction records between 6:00 and 22:00 are taken into consideration. Prediction value is the number of passengers entering and exiting each node at further times. Figures 8(a) and 8(b) illustrate the traffic network (with partial nodes) and the heat map of the passenger outflow at one time interval.

Bus Transaction Dataset: The original records of the Beijing Bus Transaction Dataset are the same as those in Subway Dataset. There are totally more than 800 bus lines and 4,500 bus stops, where 4,219 stops have valid transaction records. Therefore, nodes are the 4,219 bus stops and edges are the bus lines between two adjacent nodes. Prediction value is the passenger flow entering and exiting each node at future time interval of 1 hour. Figure 8(c) shows several bus stops within the fifth ring road.

Taxi Trajectories Dataset: Original content of the taxi dataset is the GPS trajectories of all taxis collected with about 30,000 taxis in Beijing. The taxi traffic data does

TABLE I
Dataset Meta Information

Properties	Datasets		
	Subway	Bus	Taxi
# traffic nodes	278	4219	300
time interval	10 mins	1 hour	20 mins
time span	2016/6/1 - 2016/6/29		2015/11/28 - 2016/1/26
# train days	15 days		32 days
# valid & test days	7 days		14 days
daily range	06:00-22:00		



(a) nodes on subway network (b) subway passenger heatmap

(c) bus network (partial nodes) (d) road segments on taxi dataset

Fig. 8. Dataset Description. Figure 8(a) and Figure 8(b) illustrate the graph structure of the Beijing subway network and the passenger flow heat map on one time interval, respectively. Figure 8(c) shows the bus stops within the fifth ring road of Beijing, while Figure 8(d) is the road segments of taxi network.

TABLE II
WEATHER CATEGORIES

ID	Categories	ID	Categories
0	cloudy	4	thunder
1	partly cloudy	5	fog
2	mostly cloudy	6	snow / hail
3	light rain	7	sunny

TABLE III
POIS CATEGORIES

ID	PoIs categories	ID	PoIs categories
0	food & beverage service	7	automobile service
1	shopping center	8	education
2	hotel	9	medical treatment
3	public transportation service	10	tourism service
4	entertainment	11	enterprises & institutions
5	residence	12	finance & insurance
6	living service	13	government agency

not reflect overall road features. We only employ it as a closed (platform) dataset to evaluate the proposed model. Actually, there are about 20 million people working or living in Beijing, constituting a great taxi travel demand [67]. On Taxi Dataset, according to the work in [68], nodes are the 300 main road segments (red lines in Figure 8(d)) within the fourth ring road of Beijing, and edges are the topological connections between two adjacent road segments. Prediction value is the number of taxis on each road segments at each time interval of 20 minutes.

External Factors: (1) Weather information. Weather information data consists of five record types: temperature, wind speed, visibility, precipitation, and weather condition. Weather condition is determined by the time span of three traffic datasets, with up to eight categories, which is reported in Table II. Note that weather information comes from forecasts. (2) Holiday notifications. The holiday notification data indicates whether the prediction time is on holiday or workday and thus is binary variable. (3) PoIs' distribution. With reference to [26], urban PoIs facilities are classified into 14 categories, as shown in Table III. The categories from ID 0 to ID 7 contain most of the daily travel demand, and the remaining depict the detailed social travel purposes of education, medical treatment, government management, and commercial activities. Through the above categories, main functions of different regions are described in detail, which benefit more accurate predictions.

B. Experimental Settings

We predict the traffic flow at the next time step. For the Subway and Bus Dataset, the transaction data of the first 15 days is used as the training set, and that of the last 14 days is equally divided into two parts, which are adopted as the validation and testing datasets, respectively. In the Taxi Dataset, the training dataset is the trajectories of the first 32 days, and the validation and testing datasets are equally divided from the data at last 28 days.

1) Baselines: We compare our model with eight algorithms, including both shallow models and state-of-the-art deep learning models.

- *HA:* Historical Average. It computes the average value of all historical data as the prediction result.
- *GRU:* Gated Recurrent Unit. GRU model is a commonly used time series processing model and can also be used for urban traffic prediction. In the experiments, for a fair comparison of model parameters, all traffic nodes share the same GRU unit.
- *GAT*[4] *[58]:* Graph Attention Network. GAT adopts the attentional mechanism to construct the convolutional kernel on graph structure.
- *ChebNet*[5] *[49]:* Graph Convolution with Chebyshev polynomial kernel. It is spectral approach that employs Chebyshev polynomials as an orthogonal basis.
- *DCRNN*[6] *[60]:* Diffusion Convolution Recurrent Neural Network. It embeds the diffusion convolution [54] into the RNN unit to jointly handle temporal dependencies and spatial graph structure.
- *STGCN*[7] *[64]:* Spatio-Temporal Graph Convolution Network. It adopts the ChebNet [49] to extract spatial features and captures temporal patterns with one dimensional convolution on temporal axis.
- *STGCNAction*[8] *[69]:* Spatio-Temporal Graph Convolution Network for skeleton-based human action recognition, where the last layer is modified for regression.
- *GSTNet*[9] *[22]*: Global Spatial-Temporal Network is our previous work to directly consider non-local spatial correlations.

2) Implementation Details: **Data preprocessing.** We use historical data from the past two days, *i.e.*, $P = 3$ in Equation (3). The temporal length T at each day is set to $T = 6$. The type of traffic flow data M is set to $M = 2$ for the subway and bus datasets (inflow and outflow), and $M = 1$ for the taxi dataset. All input data are first normalized to [0, 1] with the min-max normalization method.

Experimental environment. We use the Adam optimizer [70] and set the learning rate to $\alpha = 1e^{-3}$. The size of one mini-batch is set to 32 or 64 for the Subway or Taxi Dataset. Since the number of nodes in the Bus Dataset is very large (more than 4,000), the size of mini-batch is set to 4. The PyTorch toolbox is adopted to implement our model and all the comparative methods. All the models are implemented on one TITAN XP GPU with 12 GB memory.

3) Model Hyper-Parameters: The architecture of spatial feature extraction module in our model is shown in Figure 4. The spatial feature extraction model has 4 layers, and the connection hops L of each layer is set to the same value $L = 2$. The dilation rate d increases linearly. The feature dimension of each layer is 12. The 36 output channels from historical three days are then fed into the temporal feature fusion module and the output dimension is also set to 12. As for the external factors embedding module, the input dimension of weather part is 12, and the output dimension is 4. The input dimension

[4]https://github.com/Diego999/pyGAT
[5]https://github.com/mdeff/cnn_graph
[6]https://github.com/chnsh/DCRNN_PyTorch
[7]https://github.com/VeritasYin/STGCN_IJCAI-18
[8]https://github.com/yysijie/st-gcn
[9]https://github.com/WoodSugar/GSTNet

TABLE IV
EXPERIMENTAL RESULTS OF THE THREE EVALUATION DATASETS

Models	Subway Dataset			Bus Dataset			Taxi Dataset		
	MAE	MAPE (%)	RMSE	MAE	MAPE (%)	RMSE	MAE	MAPE (%)	RMSE
HA	45.08	31.02	94.94	35.93	55.47	73.54	26.18	40.24	55.95
GAT	36.68 ± 2.58	28.97 ± 2.29	65.35 ± 6.31	26.40 ± 0.29	46.88 ± 3.00	52.73 ± 0.38	22.05 ± 1.01	35.27 ± 1.47	45.44 ± 1.66
GRU	23.33 ± 0.20	20.29 ± 0.67	41.92 ± 0.33	24.07 ± 0.22	40.73 ± 1.59	53.46 ± 0.25	20.24 ± 0.19	32.77 ± 1.73	40.04 ± 0.15
ChebNet	22.91 ± 0.59	19.38 ± 0.39	40.02 ± 0.98	27.06 ± 1.10	42.89 ± 2.67	56.01 ± 2.73	19.81 ± 0.07	31.97 ± 0.38	38.39 ± 0.58
DCRNN	22.49 ± 0.22	19.50 ± 1.08	38.63 ± 0.47	27.06 ± 0.12	43.95 ± 0.47	55.23 ± 0.09	20.46 ± 0.34	31.58 ± 1.38	42.03 ± 0.18
STGCN	21.69 ± 0.62	19.13 ± 1.74	36.49 ± 0.48	23.42 ± 0.31	39.01 ± 1.57	48.80 ± 2.64	19.34 ± 0.24	31.34 ± 1.16	37.30 ± 0.26
STGCNAction	21.65 ± 0.27	18.97 ± 1.32	37.06 ± 0.39	21.05 ± 0.72	36.12 ± 1.01	40.93 ± 1.90	19.78 ± 0.13	31.45 ± 1.90	39.41 ± 0.14
GSTNet	21.33 ± 0.13	18.63 ± 0.72	36.08 ± 0.22	N / A	N / A	N / A	19.17 ± 0.32	30.77 ± 1.35	37.01 ± 0.35
MS-Net	**19.44 ± 0.14**	**16.97 ± 0.30**	**32.19 ± 0.17**	**19.15 ± 0.28**	**33.12 ± 1.19**	**36.42 ± 0.39**	**18.60 ± 0.06**	**29.37 ± 0.46**	**35.62 ± 0.12**

of holiday notification part is 2, which is the same as the output dimension. The number of input channels of PoIs embedding part is 15 (14 categories plus 1 for the sum of the number of PoIs), and the number of output channels is 15. The hyper-parameters of the proposed model and all comparison methods are searched on the validation dataset.

4) Evaluation Metric: We use the three most-widely adopted metrics [25], [60]–[65], MAE (Mean Absolute Error), MAPE (Mean Absolute Percentage Error), and RMSE (Root Mean Square Error) to evaluate the prediction accuracy of different models. MAE reflects the overall prediction accuracy, MAPE is susceptible to traffic nodes with small flow, while RMSE is more sensitive to nodes with larger traffic flow:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|\hat{x}_i - x_i\right|, \tag{18}$$

$$\text{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{\hat{x}_i - x_i}{x_i + \epsilon}\right|, \tag{19}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{x}_i - x_i)^2}, \tag{20}$$

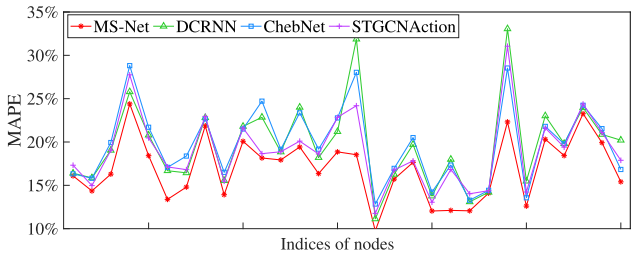
where n is the number of samples, $\epsilon = 5$ ensures the validity of division, $\hat{x}_i$ is prediction result, and x_i is ground truth.

C. Experimental Results

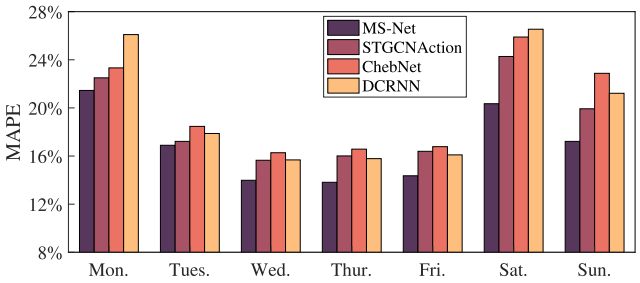
1) Model Performance: Table IV reports the next time-step prediction accuracy of the different models on all datasets. All the uncertainties are computed by re-training the models randomly and modified by a Student's t-distribution with a confidence probability of $P_r = 0.9$. Our model achieves the best prediction accuracy on all the metrics and all the datasets.

HA model provides the worst results due to the incapacity of extracting the spatio-temporal dependencies. The prediction accuracy of the models considering only the spatial correlations (GAT, ChebNet) are usually inferior to that of the methods which combine both the spatial and temporal modules (see the results of STGCN and STGCNAction). The DCRNN model performs better than the GRU model on the Subway Dataset, but are worse on the other two datasets, indicating the generalization of DCRNN is not satisfactory.

Our previous work GSTNet achieves the second best results, but the computational complexity of non-local spatial correlations are $O(N^2)$, which makes it not applicable on Bus Dataset with more than 4,000 traffic nodes. Besides, the proposed MS-Net is better than the original GAT model, although both of them employ the attention mechanism. This illustrates that the non-local spatial correlations extracted by the dilated convolution kernel can enhance the model performance.



(a) Prediction error on randomly selected 30 nodes

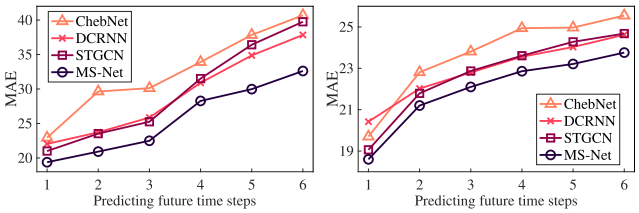


(b) Prediction error during one week

Fig. 9. Detailed prediction results of Subway Dataset.

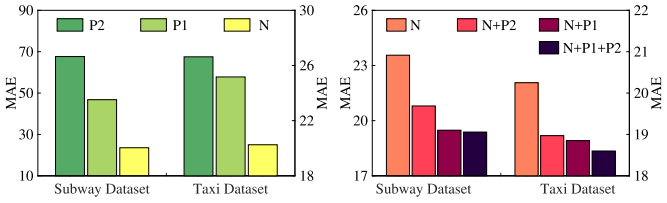
To further illustrate the effectiveness of our model, we report the average MAPE results of each traffic node among all testing samples on the Subway Dataset. Figure 9(a) shows the results. Only 30 randomly selected nodes are displayed here for clarity. Similar results can be shown for the other two datasets. The MAPE results are computed by averaging the MAPE of each station at all test samples. Our model outperforms consistently other comparable methods on these traffic nodes. Similarly, Figure 9(b) shows the average MAPE results on each day during one week. The MAPE results are calculated by averaging the MAPE of all stations at all time intervals on each day. In the figure, our model consistently achieves best results on all days.

It is noticed that the prediction accuracy on weekends is worse than that on weekdays. The possible reason is that the number of transaction records on weekends is less than that on the weekdays. In addition, the behavior of traveling on weekends is more complicated and less regular. For instance, traveling characteristics on weekdays are mainly dominated by the migration between working regions and residence. During holidays, residents may go shopping, go hiking, or work overtime. These two factors make it difficult to extract the intrinsic traffic patterns on weekends.



(a) results on subway dataset (b) results on taxi dataset

Fig. 10. Results on multi-step prediction.



(a) results on one type data (b) results on multi-types data

Fig. 11. Results of the employment with periperiodic traffic data.

TABLE V

TIME COMPARISON OF THE TWO DATASETS

Models	Subway Dataset (mins)		Bus Dataset (mins)	
	Training	Test	Training	Test
GAT	0.06	0.03	0.68	0.28
STGCNAction	0.08	0.04	0.32	0.14
MS-Net	0.08	0.04	4.32	2.29
ChebNet	0.14	0.07	0.88	0.44
GSTNet	0.36	0.18	N / A	N / A
DCRNN	0.37	0.15	8.08	4.36
STGCN	0.42	0.21	6.17	3.43
GRU	0.71	0.35	24.22	5.50

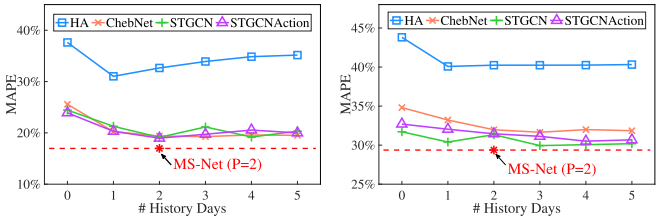
2) Multi-Step Prediction: We report MAE results on Subway and Taxi Datasets, for time steps from one to six —see Figure 10(a) and Figure 10(b), respectively. The input historical data remains unchanged, and the models are independently trained for each prediction step. Our model achieves the best prediction accuracy on these datasets for all prediction steps, demonstrating that it is also robust on long-term forecast. The ChebNet has the worst long-term predictions since the temporal dependencies are not considered. STGCN and DCRNN, both of which consider the spatio-temporal features, show similar performance, still inferior to our model.

3) Training Time: We evaluate and compare the running time of different models. Table V reports the training time and test time on the Subway and Bus Datasets, respectively. For a fair comparison, the training time is calculated on one epoch, while the test time is performed on all of the test samples. Besides, all models are evaluated under the same computing resources. Hence, GAT and STGCNAction have the best running time on both datasets, while our model presents different properties on these two datasets. Specifically, on the Subway Dataset, our model almost obtains the best running time; on the Bus Dataset, it achieves a compromise between the graph convolutional model (GAT) and the RNN based model (DCRNN). The GRU model gets the slowest running speed on Subway Dataset and on Bus Dataset (excluding GSTNet, since GSTNet is not applicable on Bus Dataset).

D. Ablation Study for Model Components

Three types of ablation studies are performed to test the following components: (1) use of historical traffic data; (2) temporal fusion method from historical data; (3) use of external factors. The experiments are performed on the Subway and the Taxi Datasets. Similar conclusions can be drawn for the Bus Dataset.

1) Adoption of Past Historical Data: Figure 11 reports the model performance resulting from different historical data settings. "N" represents the immediate past data (related to the parameter T), and "P1", "P2" stand for daily periodicity traffic flow for one day and two day, respectively. From Figure 11(a), it can be observed that when only one type of traffic data is employed, the experiment with immediate past data achieves the best prediction accuracy on both the Subway and Taxi Datasets, confirming (unsurprisingly) that the past neighboring information constitutes the most important information for an accurate prediction. Adding traffic data recorded during previous days can be employed as supplementary information to further improve the performance; it serves to smooth out the input signal —see Figure 11(b).



(a) results on subway dataset (b) results on taxi dataset

Fig. 12. Results on different historical days.

TABLE VI

AVERAGE RESULTS OF TEMPORAL FEATURE FUSION

Fusion Method	Subway Dataset		
	MAE	MAPE (%)	Time (mins)
Weighted Sum	20.85 ± 0.05	18.08 ± 0.31	0.07
Average	20.49 ± 0.12	17.89 ± 0.49	0.07
Proposed	**19.44 ± 0.14**	**16.97 ± 0.30**	0.08
	Taxi Dataset		
	MAE	MAPE (%)	Time (mins)
Weighted Sum	19.23 ± 0.17	30.93 ± 0.78	0.09
Average	19.24 ± 0.07	31.26 ± 0.35	0.09
Proposed	**18.60 ± 0.06**	**29.37 ± 0.46**	0.09

Figure 12 further explores the results of different historical days P on several comparison methods. It is observed that when $P > 2$, adding more history data could not significantly improve the performance, but bringing more computational overhead. Hence, the experimental setup of $P = 2$ is appropriate. Furthermore, all results of different methods on different history days do not exceed the results of MS-Net at $P = 2$ (red dot in the figure), which further demonstrates the effectiveness of our method.

2) Temporal Features Fusion Methods: To evaluate the effectiveness of the proposed temporal features fusion method, we compare it with two other methods. The first one is the weighted summation mentioned in Equation (9), and the second one is to directly average the temporal features at different historical days. The experimental results of Subway and Taxi Datasets are reported in Table VI. The first two

TABLE VII
AVERAGE RESULTS OF EXTERNAL FACTORS EMBEDDING

External Factors	Subway Dataset MAE	Subway Dataset MAPE (%)	Subway Dataset Time (mins)
w/o factors	20.71 ± 0.04	17.95 ± 0.15	0.06
with factors	**19.44 ± 0.14**	**16.97 ± 0.30**	0.08
	Taxi Dataset MAE	Taxi Dataset MAPE (%)	Taxi Dataset Time (mins)
w/o factors	19.25 ± 0.16	30.64 ± 0.45	0.08
with factors	**18.60 ± 0.06**	**29.37 ± 0.46**	0.09

columns of each dataset are MAE and MAPE results averaged on all nodes of the testing dataset, and the third column is the computing time of each epoch when training each model. All the uncertainties are computed by re-training the models randomly and modified by a Student's t-distribution with a confidence probability of $P_r = 0.9$.

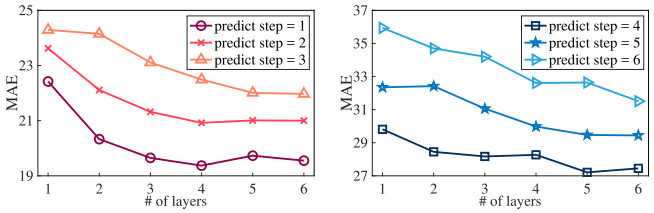
Indeed, our fusion method obtains best accuracy with statistical test on both datasets but not bring obvious computing burden, indicating that the proposed temporal fusion module is a simple but effective method. The weighted summation and the average method are not conclusive: The average method performs better on the Subway Dataset, while it is the opposite on the Taxi Dataset, suggesting that it is not a good choice to assign equal weight to all traffic nodes. However, our feature concatenating approach can consider the inherent temporal characteristics of each node, achieving a better generalization ability on different datasets.

3) Impact of External Factors: Table VII shows that with the addition of external factors, and the computing procedure is the same as in Table VI. It is observed that with considering external data, the model performance improves to a certain extent. This demonstrates the embedding module for external factors and the proposed data fusion strategy benefit more accurate predictions, without obviously increasing computational burden to the whole model.

E. Evaluation of Hyper-Parameters

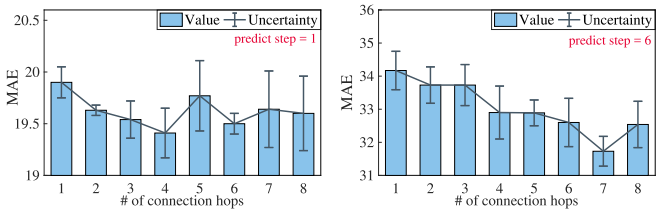
Different choices of hyper-parameters could affect the prediction accuracy. This subsection studies three hyper-parameters in the spatial feature extraction module: the number of module layers, the number of connection hops L and the dilation rate d defined in Equation (6). Models with different hyper-parameters are trained independently, and the results are reported on testing dataset. Without loss of generality, the experiments are handled on the Subway Dataset; similar conclusions can be generalized to the other two datasets.

1) Number of Layers: Figure 13 illustrates the results of different module layers with different prediction steps. To make the graphs clearer, the results of the first three steps are shown in Figure 13(a), and that of the last three steps are shown in Figure 13(b). For all experiments, the hops $L = 2$. In the short-term prediction task, the 4-layer model performs best. The shallow network has weak feature extraction capabilities, while the overly deep network could cause overfitting. As the prediction step increases, deeper networks can achieve better performance. For instance, in Figure 13(b), the 6-layer network works best in the six-step prediction task.



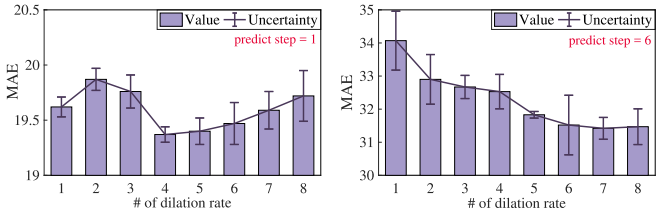
(a) results on the first three steps (b) results on the last three steps

Fig. 13. Results of different layers with different prediction steps.



(a) results on the one-step prediction (b) results on the six-step prediction

Fig. 14. Results of connection hops with two prediction steps.



(a) results on the one-step prediction (b) results on the six-step prediction

Fig. 15. Results of dilation rate with two prediction steps.

Deeper networks having higher capacity, have indeed stronger feature extraction capabilities and the ability to capture the non-local spatial correlations.

2) Hops L: We compare the impact of different hop values L on the task of short-term prediction (next step prediction) and long-term prediction (six step prediction), respectively. For simplicity, L is fixed to the same value for all layers. The results are shown in Figure 14. In this figure, the light blue histogram stands for the average prediction error, and the dark blue line indicates the standard deviation of the prediction results. All experiments of each L are trained six times with random seeds. The best four results are selected for calculating the average value and the standard deviation. A larger number of hops can enhance the performance of the model within a certain range. A long-term prediction task usually requires a larger number of hops to capture the spatial correlations with distant traffic nodes. For example, in the one-step prediction task, the setting of $L = 4$ performs best, while in the six-step prediction task, the setting of $L = 7$ achieves the lowest error.

3) Dilation Rate d: For simplicity, we fix the dilation rate to the same value for all layers, and make experiments for d taking values from 1 to 8. Figure 15 shows the MAE results for the next-step prediction task and long-term prediction task. In the figures, the MAE curves (MAE value as a function of d) show different characteristics in the two tasks. In the next-step prediction task, the prediction error first increases and then decreases, reaching the minimum value when the dilation rate is 4. As for the long-term prediction, a larger dilation rate can monotonically reduce the prediction error

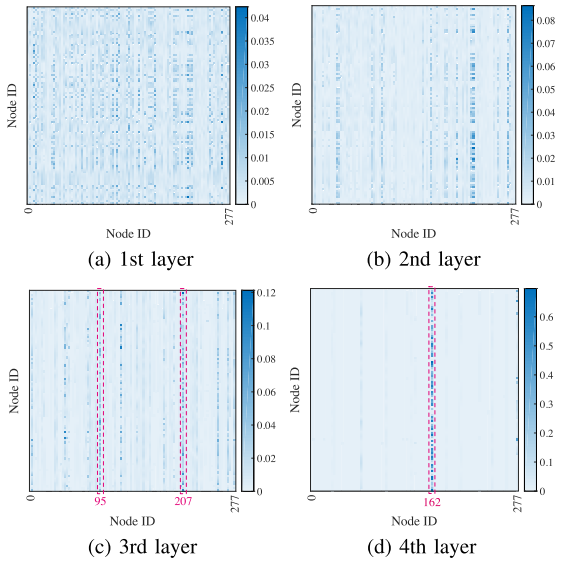


Fig. 16. Pair-wise score map at 08:00 on Jun. 23, 2016.

and the MAE is minimal when the dilation rate is equal to 7. The experimental results shown in Figure 15 suggest an interesting phenomenon, in the long-term prediction task, as temporal dependencies become weak, non-local spatial correlations carry more weights. As a result, a model with a larger receptive field on spatial domain can achieve better prediction accuracy. On the other hand, the traffic flow of a given node will affect the traffic flow of its adjacent nodes in the immediate future, as well as the traffic flow of distant nodes in the far future. Indeed, urban traffic flow reflects a dynamic spatial diffusion process.

F. Case Study

To further investigate the mechanism of our approach, we perform a case study over the Subway Dataset during the morning rush hours on Jun. 23, 2016. Figure 16 illustrates the normalized score of all pair-wise nodes' features at 08:00 in the spatial module that handles neighboring history data:

$$\text{score}_{i,j} = \frac{\exp(\boldsymbol{y}_i^{\mathrm{T}} \boldsymbol{y}_j)}{\sum_{k=1,k\neq i}^{N} \exp(\boldsymbol{y}_i^{\mathrm{T}} \boldsymbol{y}_k)}, \quad \forall v_i, v_j \in \mathcal{V}, \quad i \neq j, \tag{21}$$

where $\boldsymbol{y}_i$ is the node v_i' feature vector on a specific layer. From Figure 16(a) to Figure 16(d), the score map from the first layer to the fourth layer is respectively presented. It is observed that in the first layer, the intensity of the score map is not obvious, with the highest value being only about 0.04 (see the color bar), and the whole graph does not show clear features. However, as the number of layer increases, a few nodes are gradually getting more attention by other nodes (see each column in the score map), which not only reflects that the traffic network has come to represent certain characteristics, but also proves the existence of non-local spatial correlations (node pairs that get high scores may not be adjacent on graph).

To further explore the nodes' functions, three representative nodes (ID 95, ID 162, and ID 207) that gain key attentions are selected and highlighted with red dotted box in Figure 16(c) and Figure 16(d). The passenger flow of the three nodes during the morning rush hours is shown in Figure 17. The red lines are the Beijing Subway Network, and the blue points are the locations of each subway node (station). It is observed that although these nodes are far away and located at different directions around the city, they all express similar temporal patterns, that is, a large number of passengers are entering the subway system around 08:00, indicating the subway system is under the patterns of morning peaks on weekdays. Indeed, our model does also capture significant nodes that can represent the morning peaks features.

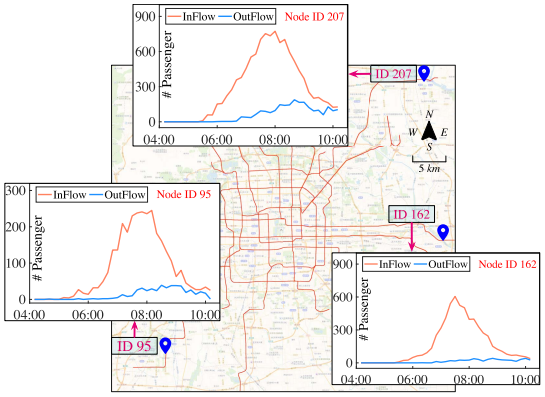


Fig. 17. Locations of three representative nodes and the traffic flow of the nodes during the morning rush hours on June 23, 2016.

VIII. Conclusion

In this paper, we address the problem of urban traffic flow prediction. To capture both the local and non-local spatial dependencies, we propose a Dilated Attentional Graph Convolution (DAGC). Based on DAGC, the Multi-Source Spatio-Temporal Network (MS-Net) is designed. Experiments on real traffic datasets prove that our model is successful when applied to both public transportation networks and road networks, and performs well in large-scale traffic networks.

There are two takeaways from our work. First, the proposed DAGC gears to the requirement of capturing non-local spatial dependencies. Second, multi-source data fusion is one of the key issues in computer science community. Heterogeneous data with different metrics usually cannot be directly measured in Euclidean space. Fortunately, this issue is addressed in the MS-Net. The design of two-stream external data embedding and two-phase feature learning can integrate various heterogeneous data, of both static (PoIs' distribution) and dynamic (weather, holiday) nature. As a result, the MS-Net constructs a general framework for multi-source data fusion.

In the future, we will extend our model to multi-step prediction task and further explore the issue of heterogeneous data fusion from different sources.

References

[1] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1624–1639, Dec. 2011.

[2] B. Ghosh, B. Basu, and M. O'Mahony, "Multivariate short-term traffic flow forecasting using time-series analysis," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 2, pp. 246–254, Jun. 2009.

[3] N. Ekedebe, C. Lu, and W. Yu, "Towards experimental evaluation of intelligent transportation system safety and traffic efficiency," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 3757–3762.

[4] Z. Ning, J. Huang, and X. Wang, "Vehicular fog computing: Enabling real-time traffic management for smart cities," *IEEE Wireless Commun.*, vol. 26, no. 1, pp. 87–93, Feb. 2019.

[5] D. Greene *et al.*, "An efficient computational architecture for a collision early-warning system for vehicles, pedestrians, and bicyclists," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 942–953, Dec. 2011.

[6] H. Malik and A. Rakotonirainy, "The need of intelligent driver training systems for road safety," in *Proc. 19th Int. Conf. Syst. Eng.*, Aug. 2008, pp. 183–188.

[7] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Short-term traffic forecasting: Where we are and where we're going," *Transp. Res. C, Emerg. Technol.*, vol. 43, pp. 3–19, Jun. 2014.

[8] L. Figueiredo, I. Jesus, J. T. Machado, J. R. Ferreira, and J. M. De Carvalho, "Towards the development of intelligent transportation systems," in *Proc. IEEE Intell. Transp. Syst.*, Aug. 2001, pp. 1206–1211.

[9] Y. Wang, J. Guo, G. Currie, A. Ceder, W. Dong, and B. Pender, "Bus bridging disruption in rail services with frustrated and impatient passengers," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 5, pp. 2014–2023, Oct. 2014.

[10] N.-E.-E. Faouzi, H. Leung, and A. Kurian, "Data fusion in intelligent transportation systems: Progress and challenges—A survey," *Inf. Fusion*, vol. 12, no. 1, pp. 4–10, Jan. 2011.

[11] S. Clark, "Traffic prediction using multivariate nonparametric regression," *J. Transp. Eng.*, vol. 129, no. 2, pp. 161–168, Mar. 2003.

[12] C. Ding, J. Duan, Y. Zhang, X. Wu, and G. Yu, "Using an ARIMA-GARCH modeling approach to improve subway short-term ridership forecasting accounting for dynamic volatility," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 4, pp. 1054–1064, Apr. 2018.

[13] H. Sun, H. X. Liu, H. Xiao, R. R. He, and B. Ran, "Use of local linear regression model for short-term traffic forecasting," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1836, no. 1, pp. 143–150, Jan. 2003.

[14] Z. Shan, D. Zhao, and Y. Xia, "Urban road traffic speed estimation for missing probe vehicle data based on multiple linear regression model," in *Proc. 16th IEEE Intell. Transp. Syst.*, Oct. 2013, pp. 118–123.

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

[16] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.

[17] W. Ouyang *et al.*, "DeepID-net: Deformable deep convolutional neural networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2403–2412.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[19] Y. Wang, P. Coppola, A. Tzimitsi, A. Messmer, M. Papageorgiou, and A. Nuzzolo, "Real-time freeway network traffic surveillance: Large-scale field-testing results in Southern Italy," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 548–562, Jun. 2011.

[20] B. Du *et al.*, "Deep irregular convolutional residual LSTM for urban traffic passenger flows prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 972–985, Mar. 2020.

[21] D. A. Tedjopurnomo, Z. Bao, B. Zheng, F. Choudhury, and A. K. Qin, "A survey on modern deep neural network for traffic prediction: Trends, methods and challenges," *IEEE Trans. Knowl. Data Eng.*, early access, Jun. 9, 2020, doi: 10.1109/TKDE.2020.3001195.

[22] S. Fang, Q. Zhang, G. Meng, S. Xiang, and C. Pan, "GSTNet: Global spatial-temporal network for traffic flow prediction," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 2286–2293.

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

[24] S. Dunne and B. Ghosh, "Weather adaptive traffic prediction using neurowavelet models," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 1, pp. 370–379, Mar. 2013.

[25] J. Sun, J. Zhang, Q. Li, X. Yi, Y. Liang, and Y. Zheng, "Predicting citywide crowd flows in irregular regions using multi-view graph convolutional networks," 2019, *arXiv:1903.07789*. [Online]. Available: http://arxiv.org/abs/1903.07789

[26] J. Wang, J. Wu, Z. Wang, F. Gao, and Z. Xiong, "Understanding urban dynamics via context-aware tensor factorization with neighboring regularization," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 11, pp. 2269–2283, Nov. 2020.

[27] H. Yao *et al.*, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proc. 32nd AAAI*, 2018, pp. 2588–2595.

[28] M. Van Der Voort, M. Dougherty, and S. Watson, "Combining kohonen maps with arima time series models to forecast traffic flow," *Transp. Res. C, Emerg. Technol.*, vol. 4, no. 5, pp. 307–318, Oct. 1996.

[29] S. Lee and D. B. Fambro, "Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1678, no. 1, pp. 179–188, Jan. 1999.

[30] W. Y. Szeto, B. Ghosh, B. Basu, and M. O'Mahony, "Multivariate traffic forecasting technique using cell transmission model and SARIMA model," *J. Transp. Eng.*, vol. 135, no. 9, pp. 658–667, Sep. 2009.

[31] J. Guo, W. Huang, and B. M. Williams, "Adaptive Kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification," *Transp. Res. C, Emerg. Technol.*, vol. 43, pp. 50–64, Jun. 2014.

[32] B. M. Williams, "Multivariate vehicular traffic flow prediction: Evaluation of ARIMAX modeling," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1776, no. 1, pp. 194–200, Jan. 2001.

[33] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, Nov. 2003.

[34] I. Okutani and Y. J. Stephanedes, "Dynamic prediction of traffic volume through Kalman filtering theory," *Transp. Res. B, Methodol.*, vol. 18, no. 1, pp. 1–11, Feb. 1984.

[35] Y. Wang and M. Papageorgiou, "Real-time freeway traffic state estimation based on extended Kalman filter: A general approach," *Transp. Res. B, Methodol.*, vol. 39, no. 2, pp. 141–167, Feb. 2005.

[36] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015.

[37] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," 2016, *arXiv:1612.01022*. [Online]. Available: http://arxiv.org/abs/1612.01022

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

[39] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, and J. Liu, "LSTM network: A deep learning approach for short-term traffic forecast," *IET Intell. Transp. Syst.*, vol. 11, no. 2, pp. 68–75, Mar. 2017.

[40] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu, "Deep learning: A generic approach for extreme condition traffic forecasting," in *Proc. 17th ICDM*, 2017, pp. 777–785.

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

[42] X. Zhou, Y. Shen, Y. Zhu, and L. Huang, "Predicting multi-step citywide passenger demands using attention-based neural networks," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, Feb. 2018, pp. 736–744.

[43] L. Liu, R. Zhang, J. Peng, G. Li, B. Du, and L. Lin, "Attentive crowd flow machines," in *Proc. 26th ACM Int. Conf. Multimedia*, Oct. 2018, pp. 1553–1561.

[44] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li, "Modeling spatial-temporal dynamics for traffic prediction," 2018, *arXiv:1803.01254*. [Online]. Available: http://arxiv.org/abs/1803.01254

[45] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li, "Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction," in *Proc. AAAI*, vol. 33, 2019, pp. 5668–5675.

[46] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," 2018, *arXiv:1812.04202*. [Online]. Available: http://arxiv.org/abs/1812.04202

[47] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proc. ICLR*, 2014, pp. 1–14.

[48] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," 2015, *arXiv:1506.05163*. [Online]. Available: http://arxiv.org/abs/1506.05163

[49] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. NIPS*, 2016, pp. 3844–3852.

[50] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, 2017, pp. 1–141

[51] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. 32th AAAI*, 2018, pp. 3538–3545.

[52] G. Li, M. Müller, A. Thabet, and B. Ghanem, "Deepgcns: Can GCNS go as deep as CNNS?" in *Proc. CVPR*, 2019, pp. 9267–9276.

[53] R. Li and J. Huang, "Learning graph while training: An evolving graph convolutional neural network," 2017, *arXiv:1708.04675*. [Online]. Available: http://arxiv.org/abs/1708.04675

[54] J. Atwood and D. Towsley, "Diffusion convolutional neural networks," in *Proc. NIPS*, 2016, pp. 1993–2001.

[55] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 2014–2023.

[56] Y. Hechtlinger, P. Chakravarti, and J. Qin, "A generalization of convolutional neural networks to graph-structured data," 2017, *arXiv:1704.08165*. [Online]. Available: http://arxiv.org/abs/1704.08165

[57] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5425–5434.

[58] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. ICLR*, 2017, pp. 1–12.

[59] S. Abu-El-Haija *et al.*, "Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing," in *Proc. 36th ICML*, 2019, pp. 21–29.

[60] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. ICLR*, 2018, pp. 1–16.

[61] D. Chai, L. Wang, and Q. Yang, "Bike flow prediction with multi-graph convolutional networks," in *Proc. 26th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, Nov. 2018, pp. 397–400.

[62] Z. Lv, J. Xu, K. Zheng, H. Yin, P. Zhao, and X. Zhou, "LC-RNN: A deep learning model for traffic speed prediction," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 3470–3476.

[63] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. 33rd AAAI*, 2019, pp. 922–929.

[64] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 3634–3640.

[65] B. Yu, H. Yin, and Z. Zhu, "ST-UNet: A spatio-temporal U-Network for graph-structured time series modeling," 2019, *arXiv:1903.05631*. [Online]. Available: http://arxiv.org/abs/1903.05631

[66] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "Urban traffic prediction from spatio-temporal data using deep meta learning," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 1720–1730.

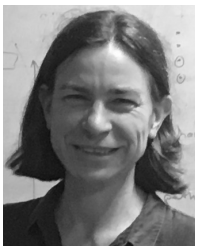
[67] En.wikipedia.org. (Nov. 25, 2020). *Beijing Wikipedia*. [Online]. Available: https://en.wikipedia.org/wiki/Beijing

[68] Q. Zhang, J. Chang, G. Meng, S. Xu, S. Xiang, and C. Pan, "Learning graph structure via graph convolutional networks," *Pattern Recognit.*, vol. 95, pp. 308–318, Nov. 2019.

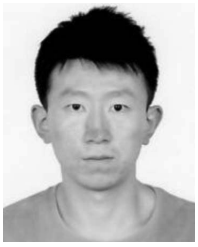
[69] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proc. 32nd AAAI*, 2018, pp. 7444–7452.

[70] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

Shen Fang received the B.S. degree from the School of Automation, Huazhong University of Science and Technology of China, Wuhan, China, in 2016. He is currently pursuing the Ph.D. degree with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His research interests include pattern recognition, deep learning, and urban computing.



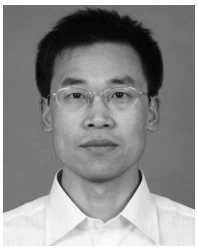
Véronique Prinet (Member, IEEE) received the Ph.D. degree in computer science from INRIA, University Paris-Saclay (formerly University Paris-11 Orsay). From 2000 to 2012, she was an Associate Professor with the Institute of Automation, Chinese Academy of Sciences, Beijing. From 2011 to 2013 and 2017 to 2018, she was a Visiting Fellow with The Hebrew University of Jerusalem, Israel. From 2014 to 2016, she was a Senior Researcher with the General Motors Advanced Technical Center, Israel. She is currently a Visiting Professor with the Institute of Automation, Chinese Academy of Sciences. Her current research interests include data-driven approaches to image processing, synthesis, and computer vision.



Jianlong Chang received the B.S. degree from the School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, China, in 2015, and the Ph.D. degree from the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2020. His research interests include pattern recognition, machine learning, and deep reinforcement learning.

Michael Werman (Member, IEEE) received the Ph.D. degree from The Hebrew University, in 1986, where he is currently a Professor of computer science. His current research interests include designing computer algorithms and mathematical tools for analyzing, understanding, and synthesizing pictures.

Chunxia Zhang received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, China, in 2005. From January 2010 to February 2011, she visited the University of Vermont, as an Academic Visitor. She is currently an Associate Professor with the School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China. Her research interests include information extraction, knowledge-based systems, and deep learning.



Shiming Xiang (Member, IEEE) received the B.S. degree in mathematics from Chongqing Normal University, Chongqing, China, in 1993, the M.S. degree from Chongqing University, Chongqing, in 1996, and the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2004. From 1996 to 2001, he was a Lecturer with the Huazhong University of Science and Technology, Wuhan, China. He held a post-doctoral position with the Department of Automation, Tsinghua University, Beijing, until 2006. He is currently a Professor with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing. His research interests include pattern recognition, machine learning, and computer vision.

Chunhong Pan (Member, IEEE) received the B.S. degree in automatic control from Tsinghua University, Beijing, China, in 1987, the M.S. degree from the Shanghai Institute of Optics and Fine Mechanics, Chinese Academy of Sciences, China, in 1990, and the Ph.D. degree in pattern recognition and intelligent system from the Institute of Automation, Chinese Academy of Sciences, Beijing, in 2000. He is currently a Professor with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences. His research interests include computer vision, image processing, computer graphics, and remote sensing.