Temporal Multi-Graph Convolutional Network for Traffic Flow Prediction

Mingqi Lv, Zhaoxiong Hong, Ling Chen, Tieming Chen, Tiantian Zhu, and Shouling Ji, *Member, IEEE*

***Abstract*— Traffic flow prediction plays an important role in ITS (Intelligent Transportation System). This task is challenging due to the complex spatial and temporal correlations (e.g., the constraints of road network and the law of dynamic change with time). Existing work tried to solve this problem by exploiting a variety of spatiotemporal models. However, we observe that more semantic pair-wise correlations among possibly distant roads are also critical for traffic flow prediction. To jointly model the spatial, temporal, semantic correlations with various global features in the road network, this paper proposes T-MGCN (*Temporal Multi-Graph Convolutional Network*), a deep learning framework for traffic flow prediction. First, we identify several kinds of semantic correlations, and encode the non-Euclidean spatial correlations and heterogeneous semantic correlations among roads into multiple graphs. These correlations are then modeled by a multi-graph convolutional network. Second, a recurrent neural network is utilized to learn dynamic patterns of traffic flow to capture the temporal correlations. Third, a fully connected neural network is utilized to fuse the spatiotemporal correlations with global features. We evaluate T-MGCN on two real-world traffic datasets and observe improvement by approximately 3% to 6% as compared to the state-of-the-art baseline.**

***Index Terms*— Traffic flow prediction, graph convolutional network, graph fusion.**

I. Introduction

As ONE of the key issues in ITS (Intelligent Transportation System), traffic flow prediction is a process of analyzing traffic conditions (e.g., speed, flow, and density) on urban road network, mining traffic patterns, and predicting the future traffic conditions on the road network. Traffic flow prediction could enable a variety of intelligent applications. For example, it could help private drivers for route planning and departure time scheduling [1] and help traffic manager to improve traffic efficiency and safety [2].

However, traffic flow prediction is a challenging task due to the complex spatial, temporal, and semantic correlations. (i) Spatial correlation. The urban traffic flow is dominated by the topological structure of the underlying road network. It is intuitive that the traffic flow of a road would greatly impact that of its adjacent roads. Moreover, the spatial correlation is directional. For example, the future traffic conditions are affected more by the downstream traffic than the upstream traffic [3]. (ii) Temporal correlation. The traffic conditions change dynamically over time. The temporal correlation can be reflected in closeness and periodicity [4]. Closeness means that traffic conditions of recent time slots are more relevant than those of distant time slots. Periodicity means that traffic conditions show periodic change patterns over certain time intervals. (iii) Semantic correlation. Distant roads may also have certain relevance with each other due to some latent semantic correlations. For example, roads in urban areas with similar functionality (e.g., residential areas and commercial areas) usually have similar traffic patterns.

The state-of-the-art methods apply data-driven approach for traffic flow prediction. Early methods only consider the temporal correlations, including the Kalman filtering based methods [5], the ARIMA (Auto-Regressive Integrated Moving Average) based methods [6], the SVR (Support Vector Regression) model based methods [7], the Bayesian model based methods [8], and deep learning based methods [9]–[12]. However, these methods ignore the spatial correlations, so that they cannot optimize the prediction performance for the entire road network. To characterize the spatial correlations, some works apply CNN (Convolutional Neural Network) for spatial modeling [4], [13]–[15]. However, CNN is originally designed for spatial structure in Euclidean space (e.g., 2D images and regular grids), so it cannot fully adapt to the complex topological structure of a road network. Aiming at this problem, several recent works study GCN (Graph Convolutional Network) for spatial modeling in road networks [3], [16], [17]. However, these methods only consider the topological relations between roads to build the graphs, but ignore all other semantic factors that could measure the correlations between roads (e.g., traffic behaviors and local functionality).

In this paper, we propose T-MGCN (**T**emporal **M**ulti-**G**raph **C**onvolutional **N**etwork), a unified deep learning framework, which integrates spatial, temporal, and semantic correlations with various global features for traffic flow prediction in a road network. The contributions of this paper are as follows.

(1) We identify two kinds of semantic correlations among roads (i.e., the historical traffic pattern correlation and the local area functionality similarity) and encode the heterogeneous spatial and semantic correlations using

Manuscript received April 17, 2019; revised September 18, 2019 and December 1, 2019; accepted March 23, 2020. This work was supported in part by the Joint Funds of the National Natural Science Foundation of China under Grant U1936215 and in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LY18F020033. The Associate Editor for this article was X. Ma. *(Corresponding author: Ling Chen.)*

Mingqi Lv, Zhaoxiong Hong, Tieming Chen, and Tiantian Zhu are with the College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China (e-mail: mingqilv@zjut.edu.cn; zhaoxionghong@outlook.com; tmchen@zjut.edu.cn; ttzhu@zjut.edu.cn).

Ling Chen and Shouling Ji are with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China (e-mail: lingchen@zju.edu.cn; sji@zju.edu.cn).

Digital Object Identifier 10.1109/TITS.2020.2983763

multiple graphs. Then, we propose a multi-graph convolutional network to model and fuse these graphs.

(2) We identify a variety of global features (i.e., time feature, periodicity feature, and event feature) and design a deep neural network to jointly model the spatial, semantic, temporal correlations, and the global features with multiple layers. Specifically, we stack a convolutional layer based on multi-graph convolutional network, a recurrent layer based on GRU (Gated Recurrent Unit), and an output layer based on fully connected neural network.

(3) We conduct extensive experiments using two real-world traffic datasets. The results show that T-MGCN reduces the prediction error by approximately 3% to 6% as compared to the best baseline.

The remainder of this paper is organized as follows. Section II reviews the related work. Section III details the proposed method. Section IV reports the experiment results. Section V concludes the paper.

II. Related Work

Traffic flow prediction is one of the major research issues in ITS. Some traditional traffic flow prediction methods use knowledge-driven approach, i.e., they analyze the physical properties of the traffic system and build models through traffic simulation and prior knowledge. The representative methods include the queuing theory model [18], the cell transmission model [19], and the microscopic fundamental diagram model [20]. However, traffic flow is often influenced by a variety of factors, so it is difficult to accurately simulate. In addition, these models usually require parameters that are difficult to obtain in real-world environment.

With the development of traffic data collection devices, data-driven traffic flow prediction methods have received considerable attention, where statistical models, shallow machine learning models, and deep learning models are three representative categories. Statistical models predict future values based on previously observed values through time-series analysis, e.g., historical average model [21], Kalman filtering model [5], ARIMA model [6] and its variations [22], [23]. However, statistical models typically rely on linear assumption, which cannot reflect the non-linear characteristics of traffic flow. Meanwhile, shallow machine learning methods can learn non-linear regularity from traffic data and external factors, yielding better traffic flow prediction results. For example, SVR model [7], Bayesian model [8], and k-nearest neighbor model [24] are mostly exploited methods. However, the performance of shallow machine learning methods heavily depends on the manually designed features. Thus, they usually cannot yield the best outcomes for prediction tasks with sophisticated regularity and complicated factors.

More recently, deep learning models have shown their superior capability for traffic flow prediction. The deep learning models could automatically extract features that capture the correlations of the data, e.g., FNN (Feed-forward Neural Network) [9], DBN (Deep Belief Network) [10], and SAE (Stacked Auto Encoder) [11]. Since traffic data are typically time-series data, it is crucial to capture the temporal correlations. Thus, RNN (Recurrent Neural Network), e.g., LSTM (Long Short-Term Memory) and GRU, is widely adopted as an essential component of traffic flow prediction models to predict a variety of traffic conditions (e.g., traffic speed [25], traffic flow [26], and travel time [27]). However, these methods ignore the spatial correlations, so that the traffic flow prediction is not constrained by the structure of the road network. Therefore, they often fail to achieve the best results for the entire road network.

Aiming at this problem, many researchers exploited CNN to characterize spatial correlations. For example, Wu *et al.* [13] combined a 1-dimension CNN and two LSTMs for short-term traffic flow prediction. Yu *et al.* [14] converted network-wide traffic speeds into a series of images, and then input them into a model that inherits the advantages of CNN and LSTM. Wang *et al.* [15] proposed a deep learning method called eRCNN for traffic speed prediction. It leverages CNN to capture the nearby road segments to improve the predictive accuracy. Zhang *et al*. [4] designed an end-to-end model called ST-ResNet, which employs residual learning to build very deep CNNs, to predict the citywide crowd flows. These methods have to transform traffic network into regular grids, because CNN is originally designed for spatial structure in Euclidean space. However, traffic data are time-series data distributed over a road network, whose structure is non-Euclidean. As a result, CNNs cannot fully capture the spatial correlations of the traffic data.

To address this limitation, several recent studies apply GCN for spatial modeling in road networks. Generally, GCNs generalize the convolution operation to non-Euclidean domains based on the spectral graph theory [28]–[30]. For example, Li *et al.* [3] proposed a deep learning framework called DCRNN for traffic flow prediction. It captures the spatial correlations using bidirectional random walks on the graph and the temporal correlations using the encoder-decoder architecture. Zhao *et al.* [16] proposed T-GCN, which uses GCN to learn the topological structure of the road network and GRU to learn the dynamic changes of traffic flow. Yu *et al.* [17] used completely convolutional structures on traffic data to capture both the spatial and temporal correlations based on GCNs. Zhang *et al.* [39] integrated GCN into a sequence-to-sequence framework for multistep speed prediction. However, these studies only consider the topological structure of the road network to build the graphs. Actually, more semantic correlations between urban roads could be exploited from various aspects (e.g., historical behaviors, local area functionality, and road type). Effectively capturing these semantic correlations could further improve the prediction performance. Furthermore, GCN has also been applied in a variety of spatiotemporal data prediction tasks, e.g., crowd flow prediction [40], passenger demand prediction [41].

More recently, Geng *et al.* [38] proposed a multi-graph convolutional network based method for ride-hailing demand prediction. Chai *et al.* [42] proposed a multi-graph convolutional network based method for bike flow prediction. However, they build the spatial correlations in a grid space,

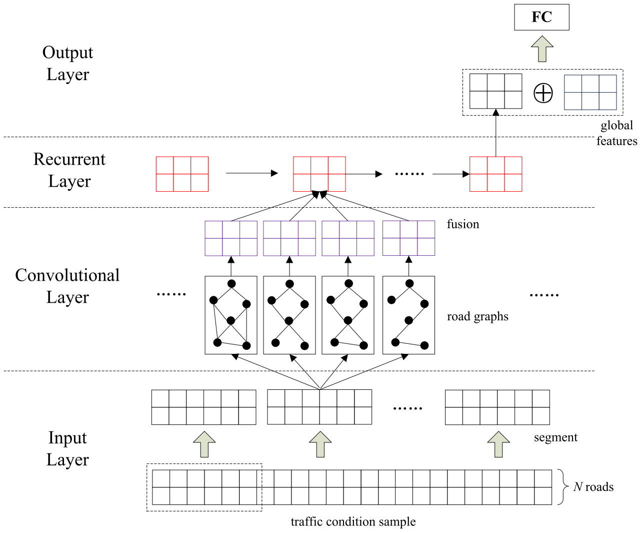


Fig. 1. The architecture of T-MGCN.

which is not suitable for traffic flow prediction in a road network.

III. Method

A. Preliminary

Definition 1 (Road Graph): A road graph is represented as a weighted graph $G = (V, E, W)$, where each node v_i represents a road and each edge e_{ij} represents the correlation between v_i and v_j. The weight of edge e_{ij} represents the correlation strength between v_i and v_j. A larger weight means that the two roads have higher correlation. In this paper, we build road graphs from three aspects, i.e., road network topological structure (represented by G_r and G_w), traffic pattern correlations (represented by G_p), and local area functionality similarities (represented by G_f), which will be elaborated in Section III.C.

Definition 2 (Traffic Condition Sample): We denote the traffic condition signal (e.g., speed and volume) on the road network at time t as an N-dimensional vector $\mathbf{x}_t$, where N is the number of roads. Then, a traffic condition sample is represented as $S_t = (In_t, Out_t)$, where the input part $In_t = [\mathbf{x}_{t-W-1}, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t]$ is a sequence of W historical traffic condition signals, the output part $Out_t = \mathbf{x}_{t+h}$ is the traffic condition signal at time $t + h$.

1) Problem: The traffic flow prediction problem can be considered as learning from a large number of traffic condition samples to obtain a function f that maps W historical traffic condition signals of the current time t to the future traffic condition signal at time $t+h$, on the premise of G_r, G_w, G_p, and G_f, as shown in Equation 1, where $\tilde{\mathbf{x}}_{t+h}$ is the predicted traffic condition signal at time $t + h$.

$$\tilde{\mathbf{x}}_{t+h} = f\left(G_r, G_w, G_p, G_f; (\mathbf{x}_{t-W-1}, \cdots, \mathbf{x}_{t-1}, \mathbf{x}_t)\right) \quad (1)$$

2) Architecture: Fig. 1 shows the architecture of T-MGCN that consists of four layers, i.e., an input layer, a convolutional layer, a recurrent layer, and an output layer. First, given a traffic condition sample S_t, the input layer uses a sliding window to process In_t to get a sequence of segments. Second, for each segment, the convolutional layer uses four individual GCNs (i.e., GCN_r, GCN_w, GCN_p, and GCN_f) to perform the convolution operation considering the four road graphs (i.e., G_r, G_w, G_p, and G_f), and fuse the results to get the feature matrix. Third, the recurrent layer applies GRU to process the sequence of feature matrices obtained from the sequence of segments. Finally, the output layer fuses the feature matrix with the global features and gets prediction result through a fully connected neural network.

B. The Input Layer

Given a traffic condition sample S_t, its input part In_t could be treated as an $N \times W$ matrix, where N is the number of roads and W is the number of historical traffic condition signals. We use a sliding window with window size w and step size d to process In_t. Then, we will obtain a sequence of K segments (i.e., $\mathbf{S}_t^1, \mathbf{S}_t^2, \cdots, \mathbf{S}_t^K$), each of which is an $N \times w$ matrix.

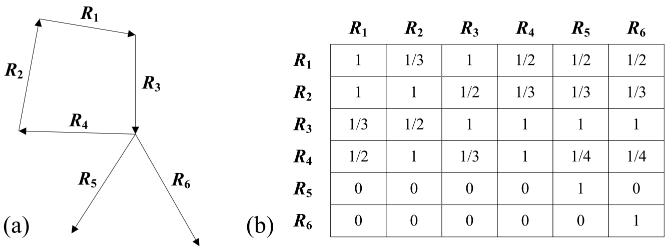
The reason of using such a processing strategy is as follows. First, each segment should contain multiple traffic condition signals ($w > 1$). It is because that time-series data have closeness feature (i.e., data in the neighboring time slots have strong local interactions) [4] and these local interactions in the neighboring time slots are captured by the convolutional layer for each segment separately. Thus, if $w = 1$, the local interactions would be ignored. Second, the input part of a traffic condition sample should be split into multiple segments (i.e., $w < W$), to facilitate the recurrent layer to capture the temporal dynamics. If $w = W$, the temporal dynamics would be ignored.

C. The Convolutional Layer

Convolution operation could well capture local dependency and maintain shift invariance, so it is an effective way to capture spatial correlations. However, the standard CNNs can only be used in Euclidean space (e.g., an image, a space with regular grids, etc.) [29], and thus it cannot adapt to the complex topological structure of the road network. Since the correlations between roads are defined in the form of graphs, we apply GCN to perform the convolution operation on road network.

1) Graph Construction: Graph construction is the key step for GCNs. If the generated graphs could not well encode the correlations between nodes, it would not help the model learning and might even degrade the prediction performance. In general, we assign larger weights to edges between roads with stronger correlations from different aspects. Specifically, we build four road graphs (i.e., G_r, G_w, G_p, and G_f), which are detailed as follows.

a) Topological graph G_r: The topological structure of a road network can be represented as a directed graph $G_r = (V, E, W_r)$, where the weight $w_r(i, j)$ of edge e_{ij} is the reciprocal of the number of hops it takes at least to travel from road v_i to road v_j. Thus, roads that are closer in the road network will be correlated with higher weights. Then, the adjacent matrix $\mathbf{X}_r$ of G_r can be represented as Equation 2.



	R_1	R_2	R_3	R_4	R_5	R_6
R_1	1	1/3	1	1/2	1/2	1/2
R_2	1	1	1/2	1/3	1/3	1/3
R_3	1/3	1/2	1	1	1	1
R_4	1/2	1	1/3	1	1/4	1/4
R_5	0	0	0	0	1	0
R_6	0	0	0	0	0	1

Fig. 2. An example of the topological graph of a road network.

Fig. 2 gives an example of a toy road network and the weight matrix of its topological structure.

$$\mathbf{X}_r = \begin{pmatrix} 0 & w_r(1,2) & \cdots & w_r(1,N) \\ w_r(2,1) & 0 & \cdots & w_r(2,N) \\ \vdots & \vdots & \ddots & \vdots \\ w_r(N,1) & w_r(N,2) & \cdots & 0 \end{pmatrix} \tag{2}$$

b) Weighted topological graph G_w: The topological graph G_r only considers the number of intermediate links from road v_i to v_j. However, the lengths of those links might also affect the correlation of v_i and v_j. Therefore, we define the weighted topological graph $G_w = (V, E, W_w)$. The weight $w_w(i, j)$ of edge e_{ij} is calculated as Equation 3, where $len(v_k)$ calculates the length of path v_k (or road v_k). Then, the adjacent matrix $\mathbf{X}_w$ of G_w can be represented as Equation 4.

$$w_w(i,j) = \frac{median\,\{len(v_k)|v_k \in V\}}{len(the\ shortest\ path\ from\ v_i\ to\ v_j)} \tag{3}$$

$$\mathbf{X}_w = \begin{pmatrix} 0 & w_w(1,2) & \cdots & w_w(1,N) \\ w_w(2,1) & 0 & \cdots & w_w(2,N) \\ \vdots & \vdots & \ddots & \vdots \\ w_w(N,1) & w_w(N,2) & \cdots & 0 \end{pmatrix} \tag{4}$$

c) Traffic pattern graph G_p: Roads with similar traffic patterns may not necessarily be close in space. With the historical traffic condition data of each road, we exploit the inter-road correlations by directly measuring the similarity of the historical traffic condition patterns of each road pair. Specifically, the traffic pattern graph is represented as $G_p = (V, E, W_p)$. The weight $w_p(i, j)$ of edge e_{ij} is the similarity between the historical condition pattern of road v_i and that of road v_j, which is calculated as follows.

First, given a road v_i, we use the average historical weekly traffic conditions sv_i as its traffic pattern, where sv_i is a sequence and $sv_i[j]$ is the average traffic condition at the j-th time slot of a week over the historical traffic condition data of v_i. Second, given two roads v_i and v_j, we use DTW (Dynamic Time Warping) [31] to calculate the distance between sv_i and sv_j, denoted as $dtw(i, j)$. Finally, we transform the distance measurement to a similarity measurement based on Equation 5, which is treated as the weight $w_p(i, j)$. In the equation, α is used to control the decay rate of the distance, which should be set according to the range of the concerned traffic condition. Then, the adjacent matrix $\mathbf{X}_p G_p$ can be represented as Equation 6.

$$w_p(i,j) = e^{-\alpha \times dtw(i,j)} \tag{5}$$

$$\mathbf{X}_p = \begin{pmatrix} 0 & w_p(1,2) & \cdots & w_p(1,N) \\ w_p(2,1) & 0 & \cdots & w_p(2,N) \\ \vdots & \vdots & \ddots & \vdots \\ w_p(N,1) & w_p(N,2) & \cdots & 0 \end{pmatrix} \tag{6}$$

d) Functionality graph G_f: Roads in urban areas sharing similar functionality usually have similar traffic patterns, e.g., industrial areas may have heavy traffic flows in rush hours of weekdays, and downtown areas may have heavy traffic flows on weekends. The functionality graph is represented as $G_f = (V, E, W_f)$. The weight $w_f(i, j)$ of edge e_{ij} is the similarity of the functionality of the local area of road v_i and that of road v_j, which is calculated as follows.

First, it has been proven in the previous studies that POI (Point Of Interest) distribution can measure the functionality of an urban area [43]. Given a road v_i, from the POIs around v_i, we calculate the density of POIs of the following eight categories, i.e., Residence, Work (e.g., company, office building, etc.), Commerce (e.g., mall, shop, etc.), Restaurant, School, Transportation (e.g., railway station, metro station, etc.), Entertainment (e.g., theatre, bar, etc.), and Scenery (e.g., park, lake, etc.) to form a feature vector $\mathbf{pv}_i$, where $\mathbf{pv}_i[j]$ denotes the density of POI category j around road v_i and is calculated as Equation 7. In the equation, m^i_j is the number of POIs of category j around road v_i, m^i is the total number of POIs around v_i, M_j is the number of POIs of category j in the POI dataset, and M is the total number of POIs in the POI dataset. The equation is designed by referring to TF-IDF in the natural language processing domain, which assigns a higher weight for POI category with less overall number. Second, given two roads v_i and v_j, we use *cosine similarity* to measure the similarity between $\mathbf{pv}_i$ and $\mathbf{pv}_j$, which is treated as the weight $w_f(i, j)$. Then, the adjacent matrix $\mathbf{X}_f G_f$ can be represented as Equation 8.

$$\mathbf{pv}_i[j] = \frac{m^i_j}{m^i} \times \log \frac{M}{M_j} \tag{7}$$

$$\mathbf{X}_f = \begin{pmatrix} 0 & w_f(1,2) & \cdots & w_f(1,N) \\ w_f(2,1) & 0 & \cdots & w_f(2,N) \\ \vdots & \vdots & \ddots & \vdots \\ w_f(N,1) & w_f(N,2) & \cdots & 0 \end{pmatrix} \tag{8}$$

2) Multi-Graph Convolutional Networks: We utilize the GCN in [30] to perform the convolution operation to capture the interactions between nodes. Each road graph is input to an individual GCN. The propagation rule of GCN can be expressed as Equation 9, where $\tilde{\mathbf{X}} = \mathbf{X} + \mathbf{I}_N$ is the adjacent matrix of a road graph with added self-connections, and $\mathbf{I}_N$ is an identity matrix. Here, $\mathbf{X}$ can be $\mathbf{X}_r$, $\mathbf{X}_w$, $\mathbf{X}_p$, or $\mathbf{X}_f$. $\tilde{\mathbf{D}}$ is a diagonal matrix such that $\tilde{\mathbf{D}}[i,i] = \sum_j \tilde{\mathbf{X}}[i,j]$. $\mathbf{W}^{(l)}$ is a layer-specific trainable weight matrix (e.g., $\mathbf{W}^{(0)}$ and $\mathbf{W}^{(1)}$ are the weight matrices of the first and second layers, respectively), and we can stack multiple layers to model higher order neighborhood interactions in the graph. The original input to

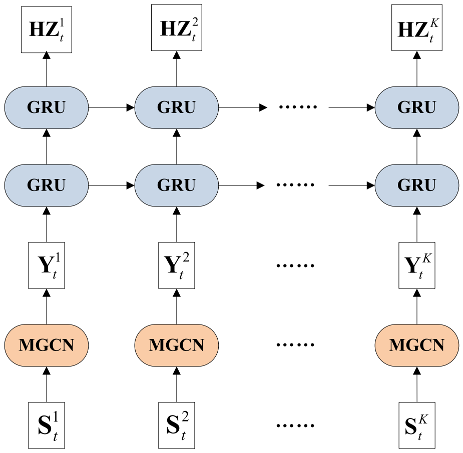


Fig. 3. The architecture of the recurrent layer.

the GCN is $\mathbf{H}^{(0)}$, which is a segment represented by an $N \times w$ matrix as discussed in Section III.B, and $ReLU(\cdot)$ denotes the ReLU activation function. The propagation rule can be considered as a spectral filter in the Fourier domain. Each segment is input to four GCNs (i.e., GCN_r, GCN_w, GCN_p, and GCN_f), together with the corresponding road graphs (i.e., G_r, G_w, G_p, and G_f), resulting in four feature matrices, denoted as $\mathbf{H}_r$, $\mathbf{H}_w$, $\mathbf{H}_p$, and $\mathbf{H}_f$. In this paper, only one layer is used in the GCN.

$$\mathbf{H}^{(l+1)} = ReLU(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{X}}\tilde{D}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}) \tag{9}$$

Next, we merge the feature matrices $\mathbf{H}_r$, $\mathbf{H}_w$, $\mathbf{H}_p$, and $\mathbf{H}_f$ into one feature matrix $\mathbf{Y}$ using a parametric matrix based fusion method in Equation 10 and 11, where $\circ$ is the element-wise product, W_r, W_w, W_p, and W_f are the learnable parameters that adjust the weights of $\mathbf{H}_r$, $\mathbf{H}_w$, $\mathbf{H}_p$, and $\mathbf{H}_f$, respectively. The softmax operation is used to normalize the parametric matrices.

$$W'_r, W'_w, W'_p, W'_f = softmax\left(W_r, W_w, W_p, W_f\right) \tag{10}$$

$$\begin{aligned}\mathbf{Y} = {} & W'_r \circ \mathbf{H}_r + W'_w \circ \mathbf{H}_w + W'_p \circ \mathbf{H}_p \\ & + W'_f \circ \mathbf{H}_f\end{aligned} \tag{11}$$

D. The Recurrent Layer

The recurrent layer is utilized to capture the temporal correlations in the sequence of segments represented by the sequence of feature matrices. In this paper, GRU is used to implement the recurrent layer. GRU is a variant of RNN, where the outputs of the previous units are a part of the input to the current unit. This mechanism allows the information to be passed step by step, and thus RNN is capable of capturing temporal correlations. GRU is proposed to address the problem of vanishing or exploding of gradient, so that it could better learn long-term temporal correlations.

We choose a stacked GRU structure with two layers, which is an efficient way to increase model capacity [32]. We apply dropout between GRU layers for regularization. As shown in Fig. 3, given a traffic condition sample S_t split into K segments (i.e., $\mathbf{S}_t^1, \mathbf{S}_t^2, \cdots, \mathbf{S}_t^K$), the convolution layer in Section III.C is applied to each segment, producing a sequence of feature matrices $\mathbf{Y}_t^1, \mathbf{Y}_t^2, \cdots, \mathbf{Y}_t^K$. Then, these feature matrices are fed into the recurrent layer chronologically. Finally, we take the output feature matrix of the last hidden state $\mathbf{HZ}_t^K$ as the output of the recurrent layer, denoted as $\mathbf{RZ}_t$ (i.e., $\mathbf{RZ}_t = \mathbf{HZ}_t^K$).

E. The Output Layer

The output of the recurrent layer $\mathbf{RZ}_t$ encodes the spatial correlations of the topological structure of road network, the inter-road semantic correlations, and the temporal correlations of closeness. However, there are other global features that can affect the traffic conditions (e.g., time and event). Thus, we extract the following global features before outputting the final feature matrix.

1) Time Feature: Given a traffic condition sample S_t, we consider time of day, day of week, and holiday event (i.e., holiday or normal day) as time feature, since they can affect urban traffic flows [4]. We extract the time feature at future time $t+h$, for which the traffic condition is predicted. We denote the time feature matrix of the road network for S_t as $\mathbf{TZ}_t$.

2) Periodicity Feature: Urban traffic flow shows a strong periodicity (e.g., traffic conditions during rush hours are usually similar on consecutive workdays) [37]. However, the recurrent layer could only capture short-term periodicity, since W (i.e., the length of the input part of a traffic condition sample) cannot be too long. This is because that a too long RNN is too difficult to train and the gradient vanishing of too long sequence also weakens the effect of periodicity. Thus, given a traffic condition sample S_t, we consider the traffic conditions at the same time in the last day $t - n^d$ and the same time in the last week $t - n^w$ as the long-term periodicity features, where n^d and n^w are the durations of a day and a week, respectively. In addition, instead of using a single traffic condition signal at $t - n^d$ and $t - n^w$, we consider a small window of traffic condition signals centered at $t - n^d$ and $t - n^w$. The reason is that the traffic conditions are usually not strictly periodic [33]. For example, although almost all the weekdays have morning rush hours, the traffic flow peak could come at different time slots on each morning. We denote the periodicity feature matrix of the road network for S_t as $\mathbf{PZ}_t$, which is a $N \times (2 \times w^p)$ matrix. Here, w^p is the number of traffic condition signals in the small window.

3) Event Feature: In some cases, an event (e.g., a traffic accident and a festival gathering) would cause the traffic pattern to change sharply in a few hours around the target area. However, the presence of such cases is very infrequent and it is almost impossible to capture the regularity by only considering the traffic condition data [35]. To address this issue, we exploit the textual information with location and time attributes shared in some event publication websites (e.g., location-based social networks and official websites of public facilities) to identify the events and capture their effect on the future traffic patterns based on a deep neural network. Specifically, given a traffic condition sample S_t, we aggregate the textual snippets published during the period of the input part In_t near each road v_i to form a document $D(t, i)$, and

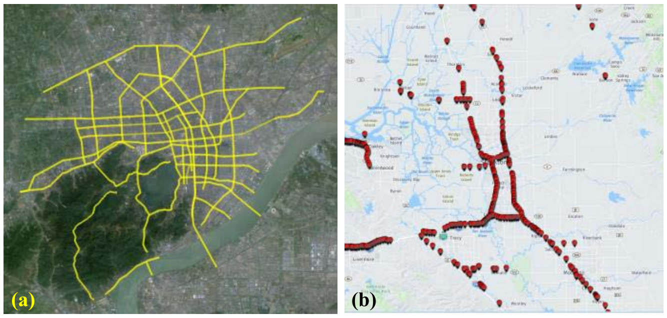


Fig. 4. An overview of the datasets: (a) the road distribution of HZJTD; (b) the sensor distribution of PEMSD10.

then use TextCNN [36] to extract a feature vector from $D(t, i)$ through a pre-trained word embedding layer, a convolutional layer, a max-over-time pooling layer, and a fully connected layer. After that, we combine all the extracted feature vectors to form the event feature matrix of the road network for S_t, denoted as $\mathbf{EZ}_t$.

Second, we fuse $\mathbf{RZ}_t$, $\mathbf{TZ}_t$, $\mathbf{PZ}_t$, and $\mathbf{EZ}_t$. To be specific, we concatenate $\mathbf{RZ}_t$, $\mathbf{TZ}_t$, $\mathbf{PZ}_t$, and $\mathbf{EZ}_t$ into a long feature matrix $\mathbf{Z}_t = \mathbf{RZ}_t \oplus \mathbf{TZ}_t \oplus \mathbf{PZ}_t \oplus \mathbf{EZ}_t$, and then stack two fully-connected layers upon $\mathbf{Z}_t$, where the first layer can be viewed as an embedding layer and the second layer is used to ensure that the output matrix $\tilde{\mathbf{Z}}_t$ has the same shape as $\mathbf{X}_t$. Note that the global features are optional and we could choose any of them to form the final $\mathbf{Z}_t$ according to the data availability. Finally, we use *sigmoid* as the activation function upon $\tilde{\mathbf{Z}}_t$ to output the predicted traffic condition signal at time $t + h$, denoted as $\tilde{\mathbf{x}}_{t+h}$. Our model is trained to minimize the RMSE (Root Mean Square Error) between the predicted traffic condition signal and the true traffic condition signal.

IV. Experiment

A. Experiment Setup

1) Datasets: We evaluate our method based on two real-world traffic datasets, i.e., HZJTD and PEMSD10.

HZJTD was collected by Hangzhou Integrated Transportation Research Center.[1] It was sampled from 202 roads in the major urban areas of Hangzhou city by loop detectors. The time period of HZJTD is from 16th October, 2013 to 3rd October, 2014. HZJTD contains traffic conditions including traffic speed and traffic congestion index. Traffic speed is taken as the traffic condition to be predicted. We aggregate traffic speed on the roads every 15 minutes. Finally, HZJTD contains 30353 records for each road. The topological structure of the road network of Hangzhou city was manually constructed (as shown in Fig. 4(a)). Note that a two-way road in the figure is treated as two distinct roads. The POIs in Hangzhou city were collected based on Baidu Map API.[2]

PEMSD10 was collected by California Department of Transportation.[3] There are totally 39,000 sensors deployed on the freeway system across all major metropolitan areas of the state of California. We randomly select 608 sensors among District 10 of California as data sources (as shown in Fig. 4(b)). The time period of PEMSD10 is from 1st January, 2018 to 31st March, 2018. The dataset is also aggregated into 15 minutes interval and traffic speed is taken as the traffic condition to be predicted. Finally, PEMSD10 contains 8640 records for each sensor. The POIs in California were collected based on Google Map API.[4] There are two things needed to be noted. First, the sensors and roads are not corresponding one to one in PEMSD10 (e.g., there may be multiple sensors deployed in the same road), so we ignore the topological graph and build the weighted topological graph by using a thresholded Gaussian kernel [3], i.e., the weight $w_w(i, j)$ between sensor v_i and v_j is calculated based on Equation 12, where $\text{dist}(v_i, v_j)$ denotes the distance between sensor v_i and v_j, σ is the standard deviation of distances, and κ is the distance threshold. Second, PEMSD10 also collects incident reports from Traffic Incident Information Page.[5] An incident report contains information about the date, time, location, and textual description of the incidents.

$$w_r(i, j) = \begin{cases} e^{-\frac{\text{dist}(v_i, v_j)^2}{\sigma^2}} & \text{if } \text{dist}(v_i, v_j) < \kappa \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

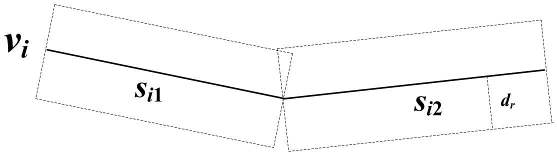


Fig. 5. An illustration of the POI projection.

2) POI Collection: First, both Baidu Map API and Google Map API do not allow to query all the POIs within an area for once, and they only accept a keyword and a central location as input and return a pre-defined number of POIs satisfying the keyword around the central location. Thus, we collected POIs based on a move-and-scan strategy, i.e., querying POIs around a central location multiple times by providing different POI categories as keywords, and then moving the central location to the next nearby location and repeating the query step. Second, the POIs are projected into a road as Fig. 5. A road v_i is represented by a polyline. For each segment s_{ij} of v_i, we draw a rectangle that wraps around s_{ij} (the orthogonal distance is d_r). Then, all the POIs within these rectangles are treated as around v_i to build the functionality graph G_f.

3) Preprocessing: First, we use linear interpolation method to fill the missing values of traffic speed data. Second, we use Min-Max normalization method to scale the traffic speed values into the range [0, 1] before inputting into the model. In the evaluation, we re-scale the predicted traffic speed value back to normal to compare with the ground-truth value. Third, since the time features are all categorical variables, we use one-hot encoding to form binary feature vectors for them.

[1] http://www.hzjtydzs.com/index.html

[2] http://lbsyun.baidu.com/

[3] http:// http://pems.dot.ca.gov/

[4] http://maps.google.com/

[5] http://cad.chp.ca.gov/

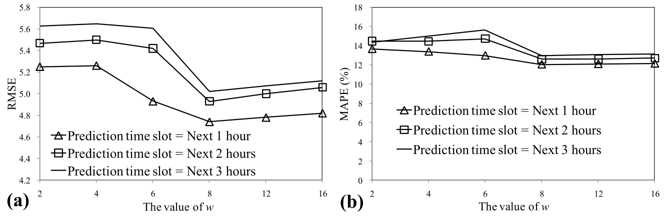


Fig. 6. The effect of parameter w: (a) the effect on RMSE; (b) the effect on MAPE.

The periodicity features are also normalized into the range [0, 1] based on Min-Max normalization method.

4) Parameter Setting: The default values of the parameters are set as follows. The number of roads is $N = 202$ for HZJTD and $N = 608$ for PEMSD10. We set the length of the input part of a traffic condition sample $W = 96$ (i.e., 24 hours), the decay rate of the DTW distance $\alpha = 0.1$, the orthogonal distance to project the POIs $d_r = 1$ km, the number of traffic condition signals in the small window of the periodicity features $w^p = 4$ (i.e., one hour), and $\kappa = 2$ km. For TextCNN, the dimension of the word embeddings is 100 and the dimension of the outputted feature vectors is 50. In the experiment, we employed a 5-fold cross validation strategy. Specifically, the traffic condition samples are chronologically split into five sets, where each set is used as testing set once (the other four sets are used as training sets) and the average performance is reported. The deep neural network is trained using the Adam optimizer. We set the learning rate as 0.001, the batch size as 32, and the training epoch as 2000.

5) Evaluation Metric: We measure the performance of the prediction methods based on RMSE and MAPE (Mean Absolute Percentage Error) as Equations 13 and 14, where $\tilde{\mathbf{X}}_i$ and $\mathbf{X}_i$ are the predicted value and the ground-truth value, and n is the number of all predicted values.

$$RMSE = \sqrt{\frac{1}{n}\sum\nolimits_{i=1}^{n}\left(\mathbf{X}_i - \tilde{\mathbf{X}}_i\right)^2} \tag{13}$$

$$MAPE = \frac{1}{n}\sum\nolimits_{i=1}^{n}\left|\frac{\mathbf{X}_i - \tilde{\mathbf{X}}_i}{\mathbf{X}_i}\right| \times 100\% \tag{14}$$

B. Experiment 1: Parameter Tuning

In this experiment, we investigate the effect of parameter w (i.e., the size of the sliding window used to segment the input, as discussed in Section III.B). The length of the input is fixed as $W = 96$ (i.e., 24 hours), the step size $d = w/2$, and we increase w from 2 (i.e., 30 minutes) to 16 (i.e., 4 hours). As shown in Fig. 6, when w increases, both RMSE and MAPE show a decreasing phase and then an increasing phase is followed. On one hand, each segment obtained by the sliding window is processed by an individual GCN. Thus, if w is too small, it is difficult for the GCNs to capture the local interactions in the neighboring time slots. On the other hand, when w is too large, the number of segments would be significantly reduced, so it is difficult for the RNN to capture the temporal correlations among segments. To the end, we set $w = 8$ in the following experiments.

C. Experiment 2: Effect of Components

In the first experiment, we try to evaluate the effectiveness of the key components of T-MGCN, i.e., the four road graphs (Section III.C). Specifically, we compare the performance of the following three variants of T-MGCN.

- T-RGCN: It performs the graph convolution operation using only G_r, the topological graph.
- T-WGCB: It performs the graph convolution operation using only G_w, the weighted topological graph.
- T-PGCN: It performs the graph convolution operation using only G_p, the traffic pattern graph.
- T-FGCN: It performs the graph convolution operation using only G_f, the functionality graph.

The four variants work without the GCN fusion step, i.e., $\mathbf{Y} = \mathbf{H}_r$ for T-RGCN, $\mathbf{Y} = \mathbf{H}_w$ for T-WGCN, $\mathbf{Y} = \mathbf{H}_p$ for T-PGCN, and $\mathbf{Y} = \mathbf{H}_f$ for T-FGCN (Section III.C). The experiment results are shown in Table I. First, T-MGCN outperforms all the other variants. It means that all the four road graphs contribute to the final results. Second, besides T-MGCN, T-PGCN has the best overall performance. It shows that the historical traffic pattern has a strong indication to the future traffic condition. Third, T-FGCN has the worst overall performance. One potential cause is that the POIs collected by Baidu Map API or Google Map API are represented equally as points. Actually, POIs with different sizes might have different degrees of influence on traffic conditions (e.g., a big mall usually has a stronger influence on traffic conditions than a small shop does).

In the second experiment, we try to investigate the effectiveness of the global features, i.e., the time features, the periodicity features, and the event features (Section III.E). Specifically, we compare the performance of the following three variants of T-MGCN.

- T-MGCN-Time: It considers only the time features in the output layer (i.e., $\mathbf{Z}_t = \mathbf{RZ}_t \oplus \mathbf{TZ}_t$).
- T-MGCN-Periodicity: It considers only the periodicity features in the output layer (i.e., $\mathbf{Z}_t = \mathbf{RZ}_t \oplus \mathbf{PZ}_t$).
- T-MGCN-Event: It considers only the event features in the output layer (i.e., $\mathbf{Z}_t = \mathbf{RZ}_t \oplus \mathbf{EZ}_t$).
- T-MGCN-None: All the global features are not considered in the output layer (i.e., $\mathbf{Z}_t = \mathbf{RZ}_t$).

The experiment results are shown in Table II. Note that only PEMSD10 has event features. First, T-MGCN has the best overall performance and T-MGCN-None has the worst. It demonstrates the effectiveness of the global features. Second, T-MGCN-Periodicity outperforms T-MGCN-Time, and it has almost the same performance as T-MGCN. It shows that the periodicity feature is the most effective among all the global features. This result is in accordance with many existing works that urban traffic flow usually has a strong periodicity [4], [13], [33], [37]. Third, T-MGCN-Event has only a trivial advantage over T-MGCN-None. By analyzing the experiment data, we found that the events (e.g., traffic accidents) had very limited impact on the future traffic flow in PEMSD10. It might be because that the car density in California is relatively low, so an event would not be likely to cause a heavy traffic congestion.

TABLE I
THE EVALUATION OF THE ROAD GRAPHS

	Prediction horizon									
	1 hour		**2 hours**		**3 hours**		**4 hours**		**5 hours**	
	RMSE	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**
HZJTD dataset:										
T-RGCN	5.35	13.82%	5.42	14.22%	5.54	14.37%	5.56	14.44%	5.59	14.39%
T-WGCN	5.47	13.90%	5.51	14.34%	5.60	14.53%	5.62	14.68%	5.66	14.59%
T-PGCN	5.34	13.12%	5.36	**12.88%**	5.44	13.60%	5.45	13.65%	5.48	13.61%
T-FGCN	5.58	14.32%	5.58	14.62%	5.67	14.76%	5.67	14.66%	5.70	14.75%
T-MGCN	**4.87**	**12.33%**	**5.05**	12.93%	**5.13**	**13.12%**	**5.14**	**13.14%**	**5.19**	**13.22%**
PEMSD10 dataset:										
T-WGCN	2.88	3.01%	2.89	3.04%	2.90	3.07%	2.92	3.07%	2.89	3.04%
T-PGCN	2.80	2.97%	2.83	3.03%	2.83	3.03%	2.84	3.03%	2.82	3.00%
T-FGCN	3.09	3.27%	3.17	3.35%	3.21	3.40%	3.27	3.40%	3.26	3.44%
T-MGCN	**2.72**	**2.88%**	**2.74**	**2.89%**	**2.77**	**2.92%**	**2.79**	**2.92%**	**2.80**	**2.98%**

TABLE II
THE EVALUATION OF THE ADDITIONAL FEATURES

	Prediction horizon									
	1 hour		**2 hours**		**3 hours**		**4 hours**		**5 hours**	
	RMSE	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**
HZJTD dataset:										
T-MGCN-Time	5.04	12.92%	5.27	13.54%	5.27	13.48%	5.30	13.62%	5.33	13.58%
T-MGCN-Periodicity	4.91	12.46%	5.09	12.95%	**5.12**	**13.09%**	5.16	13.16%	5.22	13.24%
T-MGCN-None	5.05	12.93%	5.29	13.57%	5.31	13.71%	5.34	13.66%	5.41	13.98%
T-MGCN	**4.87**	**12.33%**	**5.05**	**12.93%**	5.13	13.12%	**5.14**	**13.14%**	**5.19**	**13.22%**
PEMSD10 dataset:										
T-MGCN-Time	2.94	3.19%	2.98	3.19%	3.01	3.24%	3.03	3.25%	3.07	3.30%
T-MGCN-Periodicity	2.73	2.92%	2.77	2.94%	2.78	2.96%	2.81	2.97%	2.84	3.05%
T-MGCN-Event	2.95	3.20%	3.00	3.23%	3.04	3.31%	3.06	3.28%	3.08	3.32%
T-MGCN-None	2.95	3.21%	3.02	3.23%	3.05	3.31%	3.07	3.29%	3.10	3.33%
T-MGCN	**2.72**	**2.88%**	**2.74**	**2.89%**	**2.77**	**2.92%**	**2.79**	**2.95%**	**2.80**	**2.98%**

D. Experiment 3: Evaluation of Different Conditions

In this experiment, we try to investigate the performance of T-MGCN under different conditions (e.g., different time spans, different road types). Specifically, we evaluate T-MGCN under the following seven conditions. The prediction horizon is set as 5 hours.

- T-MGCN-RH: It refers to the performance of T-MGCN during peak hours (i.e., 7:00-9:00 and 16:30-18:30).
- T-MGCN-NP: It refers to the performance of T-MGCN during non-peak hours.
- T-MGCN-WD: It refers to the performance of T-MGCN on weekdays.
- T-MGCN-HD: It refers to the performance of T-MGCN on holidays.
- T-MGCN-ER: It refers to the performance of T-MGCN on urban expressways.
- T-MGCN-AR: It refers to the performance of T-MGCN on arterial roads.
- T-MGCN-SR: It refers to the performance of T-MGCN on roads in scenic areas.

The experiment results are shown in Fig. 7. Note that there are only freeways involved in PEMSD10, so T-MGCN-ER, T-MGCN-AR, and T-MGCN-SR are only evaluated on HZJTD. It can be seen that the experiment results on the two datasets

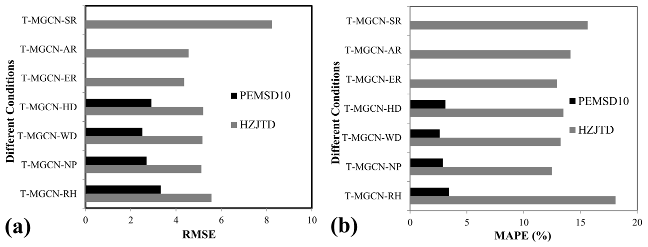


Fig. 7. The evaluation of T-MGCN under different conditions: (a) RMSE; (b) MAPE.

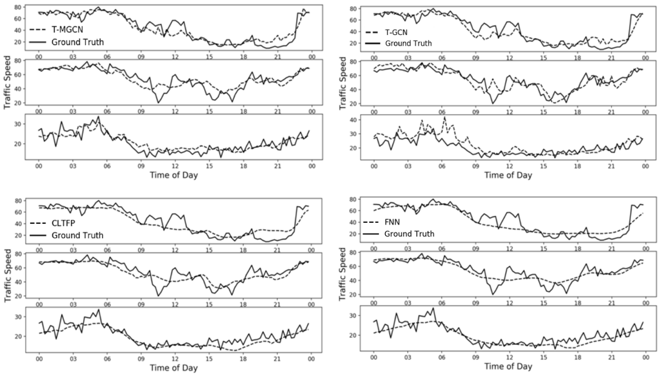


Fig. 8. A case study of traffic flow prediction: (a) prediction by T-MGCN; (b) prediction by T-GCN; (c) prediction by CLTFP; (d) prediction by FNN.

are consistent with each other. The performance is better under "normal" conditions (i.e., on non-peak hours, weekdays, urban expressways, and arterial roads). It is because that the traffic flow patterns are more unstable under "abnormal" conditions (i.e., on peak hours, holidays, and roads in scenic areas).

E. Experiment 4: Comparison With Baselines

To evaluate the competitive performance of the proposed method (i.e., T-MGCN), we compare it with the following baselines. In these baselines, HA and ARIMA have not considered the global features, while SVR, FNN, LSTM, CLTFP, T-GCN, GAT, and DCRNN have considered the global features. All the baselines have been optimized to output their best performance.

HA: It refers to the historical average model [21], which views the traffic speed as a strictly periodic process, and uses the average of previous periods as the prediction. In this paper, the period is set as one week, and thus the prediction is the average traffic speed of the same time in previous weeks.

ARIMA: It refers to the standard ARIMA model [6].

SVR: It trains the traffic flow prediction model based on the SVR algorithm [7]. Here, we use the linear kernel.

FNN: It trains the traffic flow prediction model based on the feed-forward neural network [9]. Here, we use two hidden layers. The first hidden layer contains 64 units and the second hidden layer contains 32 units.

LSTM: It refers to the method in [25], which uses the LSTM model for traffic flow prediction. Here, we stack two LSTM layers, both of which contain 32 units.

CLTFP: It refers to the method in [13], which combines CNN and LSTM for traffic flow prediction. It uses a 1-dimensional CNN to capture the spatial correlations, and then uses two LSTMs to mine the temporal correlations.

T-GCN: It refers to the method in [16], which combines GCN and GRU for traffic flow prediction. It performs the graph convolution operation considering only the topological graph.

GAT: It is similar with T-GCN, but it utilizes a cutting-edge graph neural network in [34], instead of the GCN in T-GCN.

DCRNN: It refers to the method in [3], which models the traffic speed as a diffusion process on a graph and proposes a deep learning framework for traffic flow prediction by considering both spatial and temporal correlations.

The results are shown in Table III, and the following tendencies could be discerned from the results. First, deep learning based methods (including FNN, LSTM, CLTFP, T-GCN, GAT, DCRNN, and T-MGCN) have a much better performance than traditional methods (including HA, ARIMA, and SVR). It shows that deep learning based methods can better capture the non-linear spatiotemporal correlations. Second, the recurrent deep learning based methods (including LSTM, CLTFP, T-GCN, GAT, DCRNN, and T-MGCN) outperform FNN. It indicates that traffic flow has a strong temporal correlation, which could not be ignored. Third, the spatiotemporal deep learning based methods (including CLTFP, T-GCN, GAT, DCRNN, and T-MGCN) outperform LSTM. It demonstrates the effectiveness of spatial correlation in traffic flow prediction. Fourth, the graph deep learning based methods (including T-GCN, GAT, DCRNN, and T-MGCN) outperform CLTFP. It shows that graphs can better model the spatial and semantic correlations in road network than CNNs. Fifth, all the methods behave much better on PEMSD10 than that on HZJTD. The reasons might be as follows. PEMSD10 was collected from freeways and HZJTD was collected from various types of urban roads (i.e., urban expressways, arterial roads, and roads in scenic areas). Thus, the traffic condition patterns in HZJTD would be much more complex due to a variety of urban events (e.g., traffic congestion and traffic light). Finally, T-MGCN has the best performance in both datasets. It reduces the prediction error by approximately 3% to 6% as compared to the best baseline. It shows that it could achieve a better performance by integrating various correlations (i.e., spatial, temporal, and semantic correlations) and global features (i.e., time, periodicity, and event features) jointly in T-MGCN.

F. Experiment 5: Case Study

In this section, we conduct a case study to intuitively show the performance of T-MGCN. We select three adjacent roads in the HZJTD dataset and plot the traffic flow prediction results based on several methods in Fig. 8. The traffic flow prediction is performed for the next 5 hours. It can be observed that T-MGCN captures the temporal trend of traffic speed more accurately than the other three methods. As compared to FNN and CLTFP that output smooth prediction curves, T-MGCN and T-GCN can well adapt to the sharp changes of traffic speed. In addition, the prediction curve of T-MGCN could

TABLE III
THE COMPARISON OF DIFFERENT METHODS FOR TRAFFIC SPEED PREDICTION

	Prediction horizon									
	1 hour		**2 hours**		**3 hours**		**4 hours**		**5 hours**	
	RMSE	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**
HZJTD dataset:										
HA	6.35	16.83%	6.35	16.83%	6.35	16.83%	6.35	16.83%	6.35	16.83%
ARIMA	6.22	16.46%	7.78	20.21%	8.75	23.51%	9.57	26.33%	10.29	28.66%
SVR	5.84	16.23%	6.39	17.96%	6.49	18.23%	6.51	18.27%	6.53	18.34%
FNN	5.15	13.06%	5.44	14.12%	5.54	14.58%	5.58	14.63%	5.63	14.89%
LSTM	5.29	13.67%	5.33	13.81%	5.52	14.51%	5.56	14.59%	5.61	14.81%
CLTFP	5.37	13.85%	5.43	14.09%	5.46	14.15%	5.47	14.21%	5.49	14.25%
T-GCN	5.11	13.02%	5.34	13.80%	5.44	14.11%	5.46	14.17%	5.46	14.17%
GAT	5.14	13.15%	5.32	13.76%	5.38	13.86%	5.41	14.02%	5.43	14.12%
DCRNN	5.01	12.84%	5.21	13.43%	5.27	13.61%	5.34	13.85%	5.39	13.96%
T-MGCN	**4.87**	**12.33%**	**5.05**	**12.93%**	**5.13**	**13.12%**	**5.14**	**13.14%**	**5.19**	**13.22%**
PEMSD10 dataset:										
HA	4.18	4.09%	4.18	4.09%	4.18	4.09%	4.18	4.09%	4.18	4.09%
ARIMA	3.88	3.72%	5.01	4.34%	5.84	4.74%	6.72	5.12%	7.66	5.48%
SVR	3.49	3.51%	3.62	3.69%	3.70	3.78%	3.75	3.84%	3.78	3.89%
FNN	3.03	3.23%	3.17	3.35%	3.44	3.65%	3.47	3.75%	3.52	3.81%
LSTM	**2.69**	**2.85%**	2.89	3.15%	3.09	3.49%	3.31	3.67%	3.35	3.74%
CLTFP	2.83	3.01%	2.92	3.13%	3.02	3.25%	3.02	3.28%	3.05	3.31%
T-GCN	2.9	3.04%	2.92	3.07%	2.95	3.07%	2.96	3.08%	2.97	3.10%
GAT	2.89	3.02%	2.92	3.05%	2.94	3.06%	2.96	3.09%	2.96	3.10%
DCRNN	2.85	2.98%	2.91	3.03%	2.93	3.05%	2.94	3.07%	2.94	3.09%
T-MGCN	2.72	2.88%	**2.74**	**2.89%**	**2.77**	**2.92%**	**2.79**	**2.95%**	**2.80**	**2.98%**

better align to the ground-truth curve for all the three roads in general. It shows that T-MGCN could better capture the spatial and semantic correlations of the road network to make a more accurate prediction.

V. CONCLUSION AND FUTURE WORK

In this paper, we investigate the traffic flow prediction problem on road network. We propose T-MGCN, a novel deep learning based model that encodes the non-Euclidean spatial correlations and the potential semantic correlations among roads using multiple graphs and explicitly captures them by fusing multiple graph convolutional networks. Then, the results are modeled by a recurrent neural network to encode the temporal correlations, and global features (e.g., time of day, day of week, periodicity, and events) are also incorporated into the model. Through a series of experiments based on real-world traffic datasets, we demonstrate that T-MGCN could achieve better performance than the state-of-the-art baselines.

In the future, we will extend our work from the following directions. First, most traffic flow prediction models tend to learn the general traffic patterns. However, the traffic conditions could change sharply in a few hours, which are usually caused by abnormal events (e.g., traffic accidents, concerts, etc.). We will extend our method to learn the correlations between abnormal events and traffic conditions. Second, we will extend our method for multi-step sequential traffic flow prediction.

REFERENCES

[1] J. Yuan, Y. Zheng, X. Xie, and G. Sun, “Driving with knowledge from the physical world,” in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2011, pp. 316–324.

[2] F. Belletti, D. Haziza, G. Gomes, and A. M. Bayen, “Expert level control of ramp metering based on multi-task deep reinforcement learning,” *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 4, pp. 1198–1207, Apr. 2018.

[3] 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.

[4] J. Zhang, Y. Zheng, D. Qi, R. Li, X. Yi, and T. Li, “Predicting citywide crowd flows using deep spatio-temporal residual networks,” *Artif. Intell.*, vol. 259, pp. 147–166, Jun. 2018.

[5] 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.

[6] M. M. Hamed, H. R. Al-Masaeid, and Z. M. B. Said, “Short-term prediction of traffic volume in urban arterials,” *J. Transp. Eng.*, vol. 121, no. 3, pp. 249–254, 1995.

[7] C.-H. Wu, J.-M. Ho, and D. T. Lee, "Travel-time prediction with support vector regression," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 276–281, Dec. 2004.

[8] S. Sun, C. Zhang, and G. Yu, "A Bayesian network approach to traffic flow forecasting," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 1, pp. 124–132, Mar. 2006.

[9] D. Park and L. R. Rilett, "Forecasting freeway link travel times with a multilayer feedforward neural network," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 14, no. 5, pp. 357–367, Sep. 1999.

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

[11] 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.

[12] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *Proc. 31st Youth Academic Annu. Conf. Chin. Assoc. Autom. (YAC)*, Nov. 2016, pp. 324–328.

[13] 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

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

[15] J. Wang, Q. Gu, J. Wu, G. Liu, and Z. Xiong, "Traffic speed prediction and congestion source exploration: A deep learning method," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, Dec. 2016, pp. 499–508.

[16] L. Zhao *et al.*, "Temporal graph convolutional network for urban traffic flow prediction method," 2017, *arXiv:1811.05320*. [Online]. Available: https://arxiv.org/abs/1811.05320v1

[17] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," 2017, *arXiv:1709.04875*. [Online]. Available: http://arxiv.org/abs/1709.04875

[18] X.-Y. Xu, J. Liu, H.-Y. Li, and J.-Q. Hu, "Analysis of subway station capacity with the use of queueing theory," *Transp. Res. C, Emerg. Technol.*, vol. 38, pp. 28–43, Jan. 2014.

[19] P. Wei, Y. Cao, and D. Sun, "Total unimodularity and decomposition method for large-scale air traffic cell transmission model," *Transp. Res. B, Methodol.*, vol. 53, pp. 1–16, Jul. 2013.

[20] F. F. Xu, Z. C. He, and Z. R. Sha, "Impacts of traffic management measures on urban network microscopic fundamental diagram," *J. Transp. Syst. Eng. Inf. Technol.*, vol. 13, no. 2, pp. 185–190, Apr. 2013.

[21] J. Liu and W. Guan, "A summary of traffic flow forecasting methods," *J. Highway Transp. Res. Develop.*, vol. 3, pp. 82–85, Mar. 2004.

[22] 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.

[23] 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.

[24] X. Zhang, G. He, and H. Lu, "Short-term traffic flow forecasting based on K-nearest neighbors non-parametric regression," *J. Syst. Eng.*, vol. 24, no. 2, pp. 178–183, Feb. 2009.

[25] 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.

[26] 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.

[27] Y. Duan, Y. Lv, and F.-Y. Wang, "Travel time prediction with LSTM neural network," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2016, pp. 1053–1058.

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

[29] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," 2016, *arXiv:1606.09375*. [Online]. Available: http://arxiv.org/abs/1606.09375

[30] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*. [Online]. Available: http://arxiv.org/abs/1609.02907

[31] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh, "Experimental comparison of representation methods and distance measures for time series data," *Data Mining Knowl. Discovery*, vol. 26, no. 2, pp. 275–309, Mar. 2013.

[32] S. Yao, S. Hu, Y. Zhao, A. Zhang, and T. Abdelzaher, "DeepSense: A unified deep learning framework for time-series mobile sensing data processing," in *Proc. Int. Conf. World Wide Web*, 2017, pp. 351–360.

[33] H. Yao, X. Tang, H. Wei, G. Zheng, Y. Yu, and Z. Li, "Modeling spatial-temporal dynamics for traffic prediction," 2018, *arXiv:1803.01254v1*. [Online]. Available: https://arxiv.org/abs/1803.01254v1

[34] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Li, and Y. Bengio, "Graph attention networks," 2018, *arXiv:1710.10903*. [Online]. Available: https://arxiv.org/abs/1710.10903

[35] F. C. Pereira, F. Rodrigues, and M. Ben-Akiva, "Using data from the Web to predict public transport arrivals under special events scenarios," *J. Intell. Transp. Syst.*, vol. 19, no. 3, pp. 273–288, Jul. 2015.

[36] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1746–1751.

[37] J. Ke *et al.*, "Hexagon-based convolutional neural network for supply-demand forecasting of ride-sourcing services," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 11, pp. 4160–4173, Nov. 2019.

[38] X. Geng *et al.*, "Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting," in *Proc. AAAI Conf. Artif. Intell.*, Jul. 2019, pp. 3656–3663.

[39] Z. Zhang, M. Li, X. Lin, Y. Wang, and F. He, "Multistep speed prediction on traffic networks: A deep learning approach considering spatio-temporal dependencies," *Transp. Res. C, Emerg. Technol.*, vol. 105, pp. 297–322, Aug. 2019.

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

[41] J. Ke, H. Zheng, H. Yang, and X. Chen, "Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach," *Transp. Res. C, Emerg. Technol.*, vol. 85, pp. 591–608, Dec. 2017.

[42] D. Chai, L. Wang, and Q. Yang, "Bike flow prediction with multi-graph convolutional networks," 2018, *arXiv:1807.10934*. [Online]. Available: http://arxiv.org/abs/1807.10934

[43] T. Zhang, L. Sun, L. Yao, and J. Rong, "Impact analysis of land use on traffic congestion using real-time traffic and POI," *J. Adv. Transp.*, vol. 2017, pp. 1–8, Oct. 2017.

Mingqi Lv received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 2012.

He is currently an Associate Professor with the College of Computer Science and Technology, Zhejiang University of Technology, China. His research interests include spatiotemporal data mining and ubiquitous computing.

Zhaoxiong Hong received the B.S. degree in network engineering from the Zijin College, Nanjing University of Science and Technology, China, in 2017.

Her current research interests include machine learning and neural networks.

Ling Chen received the B.S. and Ph.D. degrees in computer science from Zhejiang University in 1999 and 2004, respectively.

He is currently an Associate Professor with the College of Computer Science and Technology, Zhejiang University, China. His research interests include ubiquitous computing, human–computer interaction, and pattern recognition.

Tieming Chen received the Ph.D. degree in software engineering from Beihang University, China.

He is currently a Full Professor with the College of Computer Science and Technology, Zhejiang University of Technology, China. His research interests include data mining and cyberspace security.

Tiantian Zhu received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 2019.

He is currently a Lecturer with the College of Computer Science and Technology, Zhejiang University of Technology, China. His research interests include data mining, artificial intelligence, and information security.

Shouling Ji (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology and the Ph.D. degree in computer science from Georgia State University. He is currently a ZJU 100-Young Professor with the College of Computer Science and Technology, Zhejiang University, and also a Research Faculty with the School of Electrical and Computer Engineering, Georgia Institute of Technology.

His current research interests include AI and security, data-driven security, and data analytics. He is a member of ACM.