Learning Dynamics and Heterogeneity of Spatial-Temporal Graph Data for Traffic Forecasting

Shengnan Guo, Youfang Lin, Huaiyu Wan, Xiucheng Li, Gao Cong

Abstract—Accurate traffic forecasting is critical in improving safety, stability, and efficiency of intelligent transportation systems. Despite years of studies, accurate traffic prediction still faces the following challenges, including modeling the dynamics of traffic data along both temporal and spatial dimensions, and capturing the periodicity and the spatial heterogeneity of traffic data, and the problem is more difficult for long-term forecast. In this paper, we propose an Attention based Spatial-Temporal Graph Neural Network (ASTGNN) for traffic forecasting. Specifically, in the temporal dimension, we design a novel self-attention mechanism that is capable of utilizing the local context, which is specialized for numerical sequence representation transformation. It enables our prediction model to capture the temporal dynamics of traffic data and to enjoy global receptive fields that is beneficial for long-term forecast. In the spatial dimension, we develop a dynamic graph convolution module, employing self-attention to capture the spatial correlations in a dynamic manner. Furthermore, we explicitly model the periodicity and capture the spatial heterogeneity through embedding modules. Experiments on five real-world traffic flow datasets demonstrate that ASTGNN outperforms the state-of-the-art baselines.

Index Terms—Traffic forecasting, spatial-temporal graph data, self-attention, graph convolution.

1 Introduction

Recently, many countries have been committed to developing the Intelligent Transportation System (ITS) as part of the efforts of smart city. As an indispensable part of ITS, traffic forecast provides essential input to optimize the schedule of transportation resources, and helps people to better schedule their daily trips. Due to its great practical value, many efforts have been made towards an accurate and long-term traffic forecast for the past years.

Traffic data is a type of time-series data consecutively recorded with a fixed time duration by the deployed sensors. Therefore, it is natural to adapt the classic time series analysis models for the traffic forecasting problem in the earlier days, and the representatives include Auto-Regressive Integrated Moving Average (ARIMA) [1], and Vector Auto-Regressive (VAR) [2]. Such methods are based on the linear dependency assumption about the dynamics of time series data. Not surprisingly, they do not perform well in practice since the evolution of traffic data is often complex and nonlinear. To relax the linear assumption, traditional machine learning-based models such as SVR [3] and KNN [4] are adopted. These models deliver better results than the linear models, but their performance heavily relies on the handcrafted features, which is labor-intensive to extract. To sidestep this, increasing research efforts have been made in developing deep neural networks-based methods for the traffic forecasting problem recently. A widely adopted frame for traffic forecast is Spatial-Temporal Graph Neural Networks (STGNNs) [5], in which each node represents a traffic monitor station and the edges are indicated by the road networks. STGNNs combines a graph convolution block operating over the spatial dimension with a forward computation function modeling the dynamics across the temporal dimension. Depending on the choice of forward computation function, the STGNNs-based methods can be divided into two categories, RNNs-based approaches [6], [7], [8] and CNNs-based approaches [9], [10], [11], [12]. In comparison to the traditional machine learning methods, they are able to achieve better performance without relying on the intervention of human. Despite these achievements, the challenges still remain for an accurate and long-period traffic prediction.

First, it is still challenging to effectively model the dynamics of traffic data along both temporal and spatial dimension. Generally speaking, the concept dynamics is usually associated with a physical quantity that changes over time, and the dynamics describes how the quantity evolves in time [13]. In our case, the physical quantity is the traffic observation of interest made at each monitoring station. Denoting it by $x(t)$, the dynamics of $x(t)$ is then a black box function $f(t)$ that determines how $x(t)$ will change over time step t, i.e., we have a differential equation $\frac{\mathrm{d}x}{\mathrm{d}t} = f(t)$ describing the evolution of $x(t)$. Given the dynamics and current value $x(t_0)$, we can predict its value at any future time t_1 with $x(t_1) = x(t_0) + \int_{t_0}^{t_1} f(t)\mathrm{d}t$. The key part of traffic forecast is mostly about how to model

- *S. Guo, Y. Lin, H. Wan are with the Beijing Key Laboratory of Traffic Data Analysis and Mining, School of Computer and Information Technoloty, Beijing Jiaotong University, Beijing 100044, China, and the Key Laboratory of Intelligent Passenger Service of Civil Aviation, CAAC, Beijing, 101318, China.*
  E-mail: guoshn@bjtu.edu.cn; yflin@bjtu.edu.cn; hywan@bjtu.edu.cn.
- *S. Guo, X. Li and G. Cong are with the School of Computer Science and Engineering, Nanyang Technological University, 308232, Signapore.*
  E-mail: guoshn@bjtu.edu.cn; lixiucheng@ntu.edu.sg; gaocong@ntu.edu.sg

Manuscript received 19 April, 2020; revised 23 August, 2020; accepted 22 January, 2021.
(Corresponding author: Huaiyu Wan.)

this black box function or the dynamics f. The linear auto-regressive methods choose to model the dynamics with a linear function, $x(t) = \sum_{i=1}^{n} w_i x(t-i) + b$, in which the dynamics is completely determined by the parameter $w_1, \ldots, w_n$ and b. It is worthwhile noting two points: 1) the dynamics is linear; 2) the dynamics is invariant to the shift of time step. In other words, the dynamics does not depend on the inputs $x(t-i), i \in \{0, \ldots, n\}$. This two points greatly limit the flexibility of the model. Recurrent Neural Nets (RNNs) models the dynamics by maintaining a context-vector $\mathbf{h}_t$, a summary of past observations, which is recursively updated as $\mathbf{h}_t = \sigma(\mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_x x(t))$ with a nonlinear activation function σ. RNNs relax the linear limitation, but the dynamics is still invariant to the time shift since the parameters $\mathbf{W}_h, \mathbf{W}_x$ do not depend on inputs $x(t-i), i \in \{0, \ldots, n\}$ either. The same problem remains for Convolutional Neural Nets (CNNs) as their convolutional kernels are also invariant to the inputs.

So far we only consider modeling the dynamics along the temporal dimension for each individual traffic monitor station. However, the traffic congestion usually propagates from one road link to its neighboring streets. This implies that the evolution of observations made at a station is often correlated not only with the past history of itself but also with those of its neighboring stations. To model the spatial correlations, the de facto strategy is to perform the graph convolution over the graph induced by the road networks [6], [7], [8], [9], [12], whose structure is static. In other words, the existing methods all consider the spatial correlations to be constant, which might not hold in practice.

Second, the existing methods still struggle in making an accurate long-term forecast. As aforementioned, the existing deep learning approaches depend either on RNNs or CNNs to model the dynamics for the forward computation. It is widely known in the literature that RNNs tend to suffer from the gradient vanishing issues, especially in the long sequence modeling tasks [14]. On the other hand, the CNNs make forward computation in the sequential modeling by performing 1D convolution over the consecutive symbols with a sliding window [15]. It is also very hard for them to capture the long distance dependencies given the limited receptive fields imposed by the convolution kernels. Temporal Convolutional Network (TCN) [16], [17] as a special kind of 1D CNN is a generic architecture for sequence prediction. To capture long-term patterns, TCNs employ dilated convolutions to enable an exponentially large receptive field [18]. But it still requires a stack of several convolutional layers to connect any two positions in the sequence, which weaken its ability to learn the long-term dependencies [19], [20]. Therefore, the performance of existing approaches usually drops dramatically as the forecasting interval grows.

In addition, modeling the periodicity efficiently in long traffic sequence data and considering the spatial heterogeneity without detailed information about spatial regions still deserve more attention. Traffic data is generated by the human beings daily activities and reveals itself with clear periodic patterns. Figure 1-(1) shows the traffic flow of a detector, which repeats itself over days. Thus it is essential to consider such periodicity in an accurate traffic forecasting model design. Some works also show that explicitly modeling periodicity benefits forecasting [21], [22], [23]. However,

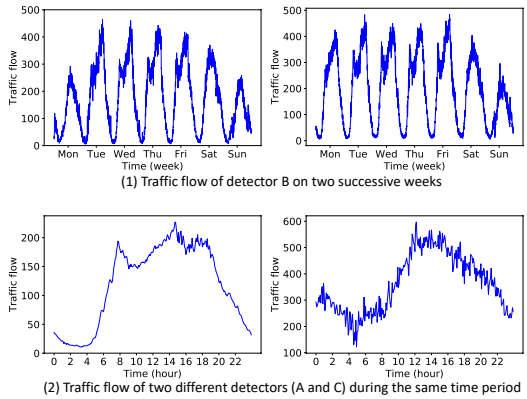


Fig. 1. (1) **Periodicity**: the traffic flow repeats its pattern over days. (2) **Spatial heterogeneity**: most of the records of detector **A** lie in the range $(0, 200)$ whereas the records of detector **C** mostly fluctuate in the range $(200, 600)$.

efficiently capturing the periodicity in long and redundant historical sequences needs clever design. Another important point in the traffic forecasting is the spatial heterogeneity [24]. Figure 1-(2) presents the changes of two detectors' traffic flow against time on the same day. The detector **A** (left one) is located on a living street and the detector **C** stays on a motorway (right one). Even during the same period, the two detectors demonstrate different traffic flow patterns—most of the records of detector **A** lie in the range $(0, 200)$ whereas the records of detector **C** mostly fluctuate in the range $(200, 600)$. This means that the traffic records vary across the spatial dimension, and different locations own their distinct traffic patterns. We refer to such observation as spatial heterogeneity, which could be explained by the fact that different spatial regions are usually associated with different static characteristics. However, the detailed static characteristics of spatial regions *e.g.*, road types, road width, speed limit and POIs (point-of-interest) might not always be accessible in practice. It remains open that how to capture such spatial heterogeneity in the cases where only road network structure is available.

To address these challenges, we propose a novel traffic forecasting model, called ***A**ttention based **S**patial-**T**emporal **G**raph **N**eural **N**etwork* (ASTGNN), which is based on self-attention mechanism. In contrast to RNNs or CNNs-based approaches, in ASTGNN the parameters of dynamics depend on the inputs, which is more flexible. To make prediction, the self-attention allows each symbol's representation to be directly informed by the representations of all other symbols in a sequence; this results in an effective global receptive field, and consequently enables our proposed model to make an accurate long-term prediction. Moreover, we explicitly model the periodicity and spatial heterogeneity, which further improves the performance. The contributions of this work are summarized as follows.

- For the first time, we propose a self-attentive traffic forecasting model—ASTGNN, which captures the dynamics in a flexible manner and offers more accurate long-term prediction.
- We design a trend-aware self-attention module that enables the self-attention being aware of the local

context, and develop a dynamic graph convolution module that models the spatial correlations in a dynamic manner.
- In contrast to the existing methods, we explicitly takes the periodicity of traffic data and spatial heterogeneity into consideration, which further improves the model performance.
- We conduct extensive experiments on five real-world traffic flow datasets. The results show that our proposed ASTGNN significantly outperforms the state-of-the-art traffic forecasting methods.

Compared to the Attention based Spatial-Temporal Graph Convolutional Networks (ASTGCN) published in our preliminary work [11], ASTGNN has the following important improvements:

- Instead of adopting 1D-CNNs, ASTGNN is equipped with a novel block — Temporal Trend-Aware Multi-Head Self-Attention. It is able to capture the dynamics of traffic data more effectively.
- In ASTGNN, we design a new dynamic graph convolution module to capture the spatial correlations in a dynamic manner. Moreover, we explicitly model the spatial heterogeneity, which further improves the model performance.
- Experiments are conducted to evaluate the effectiveness of ASTGNN; we also conduct the ablation experiments to evaluate the impact of each component of ASTGNN on the performance.

2 Related Work

2.1 Spatial-Temporal Traffic Data Prediction

Traffic prediction is a fundamental problem in Intelligent Transportation System [25] and has attracted extensive research attention during the past decades [26], [27], [28], [29], [30]. Earlier work is usually based on linear time-series analysis methods. For example, VAR [2], a typical time series analysis method, as an extension of autoregressive (AR), can take the linear inter-dependencies among multiple time series into consideration. Not surprisingly, these linear models usually perform poorly, since the changes in traffic data are quite complex and the correlations are often nonlinear. To relax the linear dependency assumption, machine learning-based methods such as SVR [3] and KNN [4] are proposed. These methods can model more complex dependencies and deliver better results than the linear models, when given high-quality handcrafted features, which are labor-intensive and time-consuming to extract.

Recently, deep learning has proven very effective in automatic feature extraction or representation learning [31], and considerable research has been done in designing deep learning-based approaches for spatial-temporal data modeling. In particular, the proposals [21], [22], [23], [26], [32], [33], [34], [35] partition the region into 2D or 3D cells, and predict the traffic volume for a cell by taking the structured space as input with CNNs (Convolutional Neural Networks). For example, Zhang et al [32] propose ST-ResNet based on the residual convolution units to predict crowd flows. Yao et al [33] integrate CNN and LSTM to jointly capture spatial and temporal dependencies for traffic prediction. Guo et al [35] propose to use 3D-CNN to capture both spatial and temporal correlations. Zhang et al [34] combine ConvLSTM and 3D-CNN together to model long-term trends and short-term variations of the mobile traffic volumes.

The aforementioned CNNs-based methods partition the space into cells and perform convolution operation over the structured 2D/3D space, which fail to capture the topology induced by the road networks. To take the road network topology into consideration, the proposals [6], [8], [9], [10], [11], [12], [36] directly represent the traffic data with a graph and make the prediction by using the spatial-temporal graph neural networks (STGNNs) [5]. These STGNNs models can be divided into RNN-based approaches and CNN-based approaches, which use RNN and CNN, respectively, for making the forward computation along the temporal dimension.

RNN-based STGNNs employ recurrent units to store historical information and replace the original linear projections in the recurrent units with graph convolution operations [6], [7], [8], [37]. Although RNNs are designed to learn long-term correlations, theoretical and empirical evidence shows that it is difficult to learn to store information for very long sequence [14], [38]. Moreover, once an RNN is unfolded in time, it can be regarded as a deep neural network sharing the same weights at each time slice. Such a parameter sharing mechanism restricts the representation capacity of RNN-based models to describe the complex dynamics of temporal correlation.

In addition, many CNN-based STGNNs are proposed in the literature in which the temporal dimension is modeled by CNNs. Bai et al [17] perform an evaluation of convolutional and recurrent architectures for sequence modeling and their experiments indicate the convolutional architecture is usually superior to the recurrent network across various datasets. CNN-based STGNNs [9], [10], [11], [12] utilize 1D CNN along the temporal dimension to capture temporal features, and use graph convolutions to capture spatial features. CNN-based models make full use of parallel computing to speed up the training process. However, limited by the kernel size, it is also hard for 1D CNN to capture the long-term temporal correlation. In addition, the parameter sharing scheme, the key idea behind CNNs in order to control the number of parameters, restricts the representation capacity of CNN-based models. Actually, the parameter sharing assumption may not even hold when dealing with the correlation of traffic data with strong dynamics. To mitigate this, STSGCN [12] proposes a multiple-module mechanism to replace the original parameter sharing scheme in CNNs to detect multiple dynamics in data. But the parameters of these multiple dynamics are still invariant to the input, which is the second limitation of modeling dynamics discussed in Introduction. Graph WaveNet [10] is another CNN-based STGNN that proposes a novel dependency matrix for graph convolution operations in the spatial dimension. The dependency matrix is calculated based on the learned node embedding vectors. Once the training of the model finishes, the dependency matrix is fixed. Therefore, Graph WaveNet still treats the parameters of dynamics as invariant with respect to inputs.

Several STGNNs [21], [37] attempt to incorporate various side information to improve prediction accuracy, *e.g.*,

POI (Point-of-Interest) distributions, weathers, detailed road network information (such as road types, road width, road speed limit). However, such side information might not be available in many scenarios. In contrast, our work focuses on a more general setting of the spatial-temporal graph data prediction problem, where only the time series of each node and the graph structure are available.

2.2 Attention Mechanism

Attention mechanism provides a general method to model the dependency between a collection of values and the target under a query, by adaptively assigning to each value in the collection a weight that is determined by the query and keys associated to them. It has been widely used and achieved great successes in various tasks such as natural language processing (NLP) [39], [40], image caption [41] and speech recognition [42]. SAnD [43], DSANet [44] extend its usage into multivariate time series prediction. Notably, ASTGCN [11], MRA-BGCN [8] and GMAN [36] also demonstrate the effectiveness of attention mechanism in modeling the dynamics of traffic data.

Self-attention is a particular implementation of attention mechanism, where the queries, keys and values are the same sequence of symbol representations. The Transformer model which entirely relies on self-attention mechanism achieves superior performance in a range of sequence modeling tasks [19]. It transform a sequence by using self-attention to compute a series of context-informed vector representations of the symbols in the input sequence. As each symbol's representation is directly informed by all other symbols' representations, this results in an effective global receptive field, which stands in contrast to RNNs and CNNs. Hence, self-attention offers a more flexible mechanism to model the complex dynamics and long-term patterns of traffic data.

3 Problem Statement and Preliminaries

We first present the *Problem Statement*, and then briefly review the key idea of attention mechanism.

3.1 Problem Statement

Definition 1. **Traffic Network.** We define a traffic network as a directed or an undirected graph $G = (V, E)$, where V is a set of $|V| = N$ nodes and each node corresponds to a traffic surveillance, *e.g.*, traffic detector or observation station; E is a set of $|E| = M$ edges.

Our proposed solution is applied to both directed and undirected networks.

Definition 2. **Traffic Signal Matrix.** The observations on the traffic network G at time slice t are denoted by a traffic signal matrix $\mathbf{X}_t = (\mathbf{x}_{t,1}, \mathbf{x}_{t,2}, \ldots, \mathbf{x}_{t,N})^T \in \mathbb{R}^{N \times C}$, where $\mathbf{x}_{t,v} \in \mathbb{R}^C$ denotes the feature vector (a collection of variables of interest) of node v at time t and C is the number of features.

Problem Statement. *Traffic Forecasting.* Given a sequence of recent historical spatial-temporal traffic signal matrices $\boldsymbol{\mathcal{X}} = (\mathbf{X}_{t-T_h+1}, \mathbf{X}_{t-T_h+2}, \ldots, \mathbf{X}_t) \in \mathbb{R}^{N \times C \times T_h}$ over the past T_h time slices, a global periodic sequence $\boldsymbol{\mathcal{X}_g}$, and a local periodic sequence $\boldsymbol{\mathcal{X}_l}$ (which will be detailed in Section 4.3.1), we aim to predict the sequence of future traffic signal matrices $\boldsymbol{\mathcal{Y}} = (\mathbf{X}_{t+1}, \mathbf{X}_{t+2}, \ldots, \mathbf{X}_{t+T_p}) \in \mathbb{R}^{N \times C \times T_p}$ over the next T_p time slices.

Most of related work [6], [12], [45] on the traffic forecasting problem only takes the recent historical traffic $\boldsymbol{\mathcal{X}}$ as the input. Therefore, to make a fair comparison, we first follow the same input setting, and compare our model ASTGNN with these baselines. In addition, as mentioned in Section 1, the traffic data is related to human daily activities and reveals with periodic patterns. We further explicitly model the periodicity of traffic data and include a global periodic sequence $\boldsymbol{\mathcal{X}_g}$ and a local periodic sequence $\boldsymbol{\mathcal{X}_l}$ into the input. To distinguish the two settings, we use ASTGNN(p) to denote our model when the input includes the two additional sources.

3.2 Attention Mechanism

Attention [46] is a fundamental operation in our model. Figure 2 illustrates the high-level idea of attention mechanism. It aims to map a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is a weighted sum of the values, where the weight assigned to each value is determined by the corresponding key and the query together. Each weight denotes the relation strength between the query and each key-value pair.

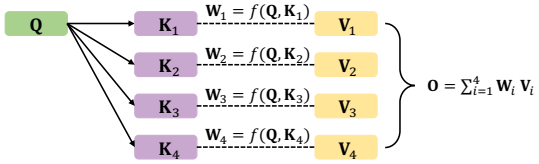


Fig. 2. The high-level idea of attention mechanism.

Scaled Dot-Product Attention. Scaled Dot-Product Attention [19] is a type of attention function where the weights are calculated as the dot-product between queries and values, and thus it enjoys the appealing properties such as space and time efficient. Formally, it is defined as follows,

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_{\text{model}}}})\mathbf{V} \quad (1)$$

where $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$ and d_{model} are queries, keys, values and their dimension respectively.

4 Attention Based Spatial-Temporal Graph Neural Networks

The high level idea of our proposed method is to directly model the complex dynamics of correlation with self-attention along both the temporal and spatial dimension. Our proposed model ASTGNN builds on the general encoder-decoder framework [47], and Figure 3 shows its overall architecture. Both the encoder and decoder consist of a stack of identical layers. To ensure effective training as the model goes deeper, the residual connection [48] and layer normalization [49] are used inside the blocks, and thus all layers in the encoder share the same dimension

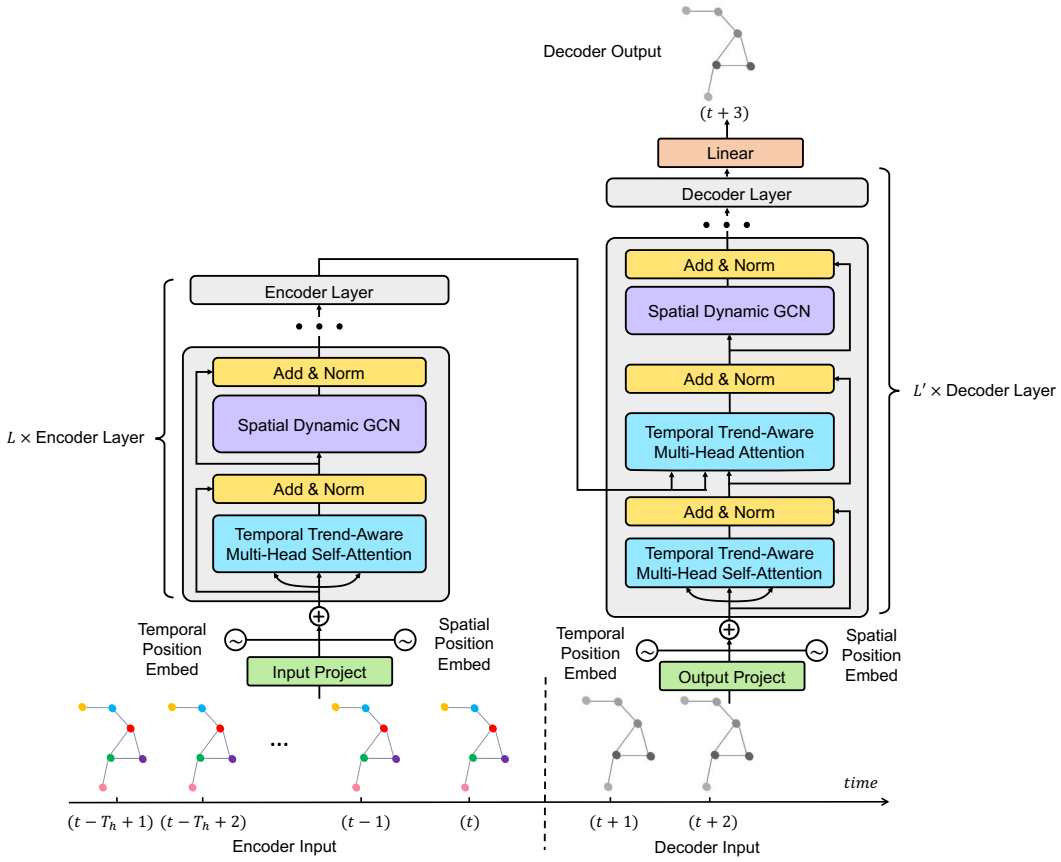


Fig. 3. ASTGNN architecture. ASTGNN follows an encoder-decoder structure. Both the encoder and decoder alternately stack multiple temporal trend-aware self-attention blocks (i.e., black blocks) and spatial dynamic GCN blocks (i.e., purple blocks). At each step, the model is auto-regressive, using the previously generated data as additional input when generating the next.

d_{model}. Denoting the input to the l^{th} encoder layer by $\boldsymbol{\mathcal{X}}^{(l-1)} = (\mathbf{X}_{t-T_h+1}^{(l-1)}, \mathbf{X}_{t-T_h+2}^{(l-1)}, \ldots, \mathbf{X}_{t}^{(l-1)}) \in \mathbb{R}^{N\times d_{\text{model}}\times T_h}$ with $l \in \{1, 2, \ldots, L\}$, the pipeline of ASTGNN works as follows.

First, the raw input $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{N\times C\times T_h}$ is converted into a high dimensional representation $\boldsymbol{\mathcal{X}}^{(0)} \in \mathbb{R}^{N\times d_{\text{model}}\times T_h}$ via linear projection with a spatial embedding layer and a temporal embedding layer, where $d_{\text{model}} > C$. Then the encoder maps the input sequence $\boldsymbol{\mathcal{X}}^{(0)} = (\mathbf{X}_{t-T_h+1}^{(0)}, \mathbf{X}_{t-T_h+2}^{(0)}, \ldots, \mathbf{X}_{t}^{(0)})$ to $\boldsymbol{\mathcal{X}}^{(L)} = (\mathbf{X}_{t-T_h+1}^{(L)}, \mathbf{X}_{t-T_h+2}^{(L)}, \ldots, \mathbf{X}_{t}^{(L)})$, a sequence of intermediate representations, through L encoder layers. By conditioning on the encoder final output $\boldsymbol{\mathcal{X}}^{(L)}$, the decoder employs another L' decoder layers to generate the output sequence $\boldsymbol{\mathcal{Y}}^{(L')} = (\mathbf{X}_{t+1}^{(L')}, \mathbf{X}_{t+2}^{(L')}, \ldots, \mathbf{X}_{t+T_p}^{(L')})$ of future traffic signals. In the end, $\boldsymbol{\mathcal{Y}}^{(L')}$ is mapped into the target outputs $\boldsymbol{\mathcal{Y}}$ by a linear projection. The generation process is auto-regressive, *i.e.*, in order to generate $\boldsymbol{\mathcal{Y}}_i^{(L')}$ with $i \in \{1, \ldots, T_p\}$ at each step, the decoder takes $\boldsymbol{\mathcal{X}}^{(L)}$ and all previously generated traffic signal matrices $\boldsymbol{\mathcal{Y}}_{1:i-1}^{(L')}$ as inputs.

4.1 Spatial-Temporal Encoder

The spatial-temporal encoder consists of a stack of identical layers, and each layer contains two basic blocks, namely, temporal trend-aware multi-head self-attention block and spatial dynamic GCN block. The temporal trend-aware multi-head self-attention block aims to model the dynamics of traffic data across the temporal dimension, and the spatial dynamic GCN block seeks to capture the spatial-relevant dynamics of traffic data.

4.1.1 *Temporal Trend-Aware Multi-Head Self-Attention*

Self-attention is a particular implementation of attention mechanism in which the queries, keys, and values are the same sequence of symbol representations. Multi-Head Self-Attention [19] is the most widely adopted self-attention in practice, and it enables to jointly attend to information from different representation subspaces. The basic operation in the multi-head self-attention is the scaled dot-product attention defined in Eq. 1, where all the queries, keys and values are the same sequence of representations, *i.e.*, $\mathbf{Q} = \mathbf{K} = \mathbf{V}$. The multi-head self-attention first linearly projects the queries, keys and values into different representation subspaces and then performs the attention function (Eq. 1) in parallel. At last, the outputs are concatenated and further projected, resulting in the final output. Formally,

$$\text{MHSelfAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \oplus(\text{head}_1, \ldots, \text{head}_h)W^O \tag{2}$$

$$\text{head}_j = \text{Attention}(\mathbf{Q}W_j^Q, \mathbf{K}W_j^K, \mathbf{V}W_j^V) \tag{3}$$

where h is the number of attention heads. W_j^Q, W_j^K, W_j^V are projection matrices applied on $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ and W^O is the final output projection matrix. The multi-head self-attention

allows modeling the correlation of elements in sequences regardless of their distance, and this results in an effectively global receptive fields. It provides a flexible manner to capture the complex dynamics of correlation in traffic data, and thus enables an accurate long-term prediction.

However, the multi-head self-attention was initially proposed to process the discrete tokens (*e.g.*, words) and fails to consider the local trend information inherent in continuous data. Therefore, simply applying it to the traffic signal sequence transformation might lead to the mismatching issue. We illustrate this with an example. The curve in Fig. 4 shows a traffic signal series of a traffic detector, where A, B and C refer to three data points at different time slices. In this case, the traditional self-attention mechanism will wrongly match point A with B, since they have the same point-wise numerical value. However, A and B have different local trends where A is in a plateau while B is at the peak of a fluctuation. In other words, the recent historical trends of A and B are significantly different. Therefore, if we apply the multi-head self-attention to this piece of numerical data, we will assign wrong correlation strengths to data pairs in the sequence. Then we may get a bad sequence representation, which will affect the final prediction performance.

To address the local trend agnostics issue of traditional multi-head self-attention in numerical data prediction, we design a temporal trend-aware multi-head self-attention mechanism, which takes the local contextual information into consideration. The temporal trend-aware multi-head self-attention is shared among nodes. It is a variant of Convolutional Self-Attention [50], which replaces the projection operations on the queries and keys in Eq. 3 with the 1D convolutions. Since the convolution operation computes the representations by taking the local contextual context as input, this allows our model to be aware of the local changing trend hidden in the traffic data series. Formally, our temporal trend-aware multi-head self-attention (TrSelfAttention) is defined as follows,

$$\begin{aligned}\text{TrSelfAttention}(\mathbf{Q},\mathbf{K},\mathbf{V}) &= \oplus(\text{Trhead}_1,\ldots,\text{Trhead}_h)W^O \\ \text{Trhead}_j &= \text{Attention}(\Phi_j^Q \star \mathbf{Q}, \Phi_j^K \star \mathbf{K}, \mathbf{V}W_j^V),\end{aligned} \tag{4}$$

which is shared among nodes and $\star$ indicates the convolution operation and Φ_j^Q, Φ_j^K are the parameters of convolution kernels. Consider the example (right) in Fig. 4. With our trend-aware self-attention, points B and C can be correctly matched since they show analogous local changing trends.

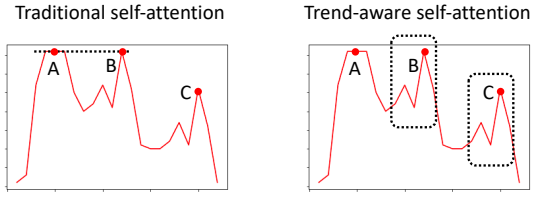


Fig. 4. The comparison between traditional self-attention (left) and our temporal trend-aware self-attention (right) when they are applied on a traffic signal series of a traffic detector. The traditional self-attention (left) will wrongly match points A with B since they have the same point-wise numerical value. In contrast, our temporal trend-aware self-attention (right) can correctly match the most relevant points (B and C) based on their local trends.

In the l^{th} encoder layer, given input $\boldsymbol{\mathcal{X}}^{(l-1)}$, after performing the temporal trend-aware multi-head self-attention on all the node, we obtain an intermediate sequence representation denoted by $\boldsymbol{\mathcal{Z}}^{(l-1)} = (\mathbf{Z}_{t-T_h+1}^{(l-1)}, \mathbf{Z}_{t-T_h+2}^{(l-1)}, \ldots, \mathbf{Z}_t^{(l-1)}) \in \mathbb{R}^{N\times d_{\text{model}}\times T_h}$.

Due to the trend-aware self-attention mechanism, in our model the parameters of temporal dynamics are computed based on the inputs. In contrast, the temporal dynamics are assumed to be invariant to the shift of time step in the previous RNN-based and CNN-based STGNNs.

4.1.2 Spatial Dynamic Graph Convolution

To capture the dynamics across spatial dimension, we further design a Dynamic Graph Convolution Net (DGCN) block based on GCNs [51]. GCNs generalize the traditional convolution operation from structured data to graphs, and they are capable of capturing the non-structured patterns hidden in the graphs. The general idea of GCNs is to learn node representations by exchanging information among them. Specifically, given a node, GCNs first aggregate its neighboring representations to produce the node an intermediate representation, and then they transform the aggregated representation with a linear projection followed by a non-linear activation. In our case,

$$\text{GCN}(\mathbf{Z}_t^{(l-1)}) = \sigma(\mathbf{A}\mathbf{Z}_t^{(l-1)}\mathbf{W}^{(l)}) \tag{5}$$

where $\mathbf{Z}_t^{(l-1)} \in \mathbb{R}^{N\times d_{\text{model}}}$, $\mathbf{W}^{(l)} \in \mathbb{R}^{d_{\text{model}}\times d_{\text{model}}}$, and σ are node representations, projection matrix, and nonlinear activation, respectively. $\mathbf{A} \in \mathbb{R}^{N\times N}$ represents the interaction relationship among nodes, which is defined as follows,

$$\mathbf{A} = \begin{cases} \widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}, & \text{undirected graph} \\ \widetilde{\mathbf{D}}^{-1}\widetilde{\mathbf{A}}, & \text{directed graph} \end{cases}$$

in which $\widetilde{\mathbf{A}}$ is the graph adjacency matrix, and $\widetilde{\mathbf{D}}_{ii} = \sum_j \widetilde{\mathbf{A}}_{ij}$.

Such a traditional graph convolution operation is time invariant, *i.e.*, given a graph G the corresponding weight matrix $\mathbf{A}$ is a constant. However, for the traffic network the correlation among nodes is very likely to change over time, and simply applying the GCNs to the traffic network will fail to capture such dynamics. For this reason, we develop a dynamic graph convolution net, DGCN, which is able to adaptively adjust the correlation strengths among nodes.

The idea is to employ self-attention to dynamically calculate the spatial correlation strengths among nodes. In our case, given the node representations $\mathbf{Z}_t^{(l-1)} \in \mathbb{R}^{N\times d_{\text{model}}}$ (*i.e.*, the output of temporal trend-aware multi-head attention block) as input, the spatial correlation weight matrix $\mathbf{S}_t$ is calculated as follow,

$$\mathbf{S}_t = \text{softmax}(\frac{\mathbf{Z}_t^{(l-1)}{\mathbf{Z}_t^{(l-1)}}^T}{\sqrt{d_{\text{model}}}}) \in \mathbb{R}^{N\times N} \tag{6}$$

Intuitively, the element S_{ij} of $\mathbf{S}_t$ represents the correlation strength between node i and node j – a large value indicates a strong correlation while a small one implies a weak correlation. Once the spatial correlation weight matrix $\mathbf{S}_t$ is obtained, we utilize it to adjust the static weight matrix $\mathbf{A}$ with an element-wise dot-product operation,

$$\mathbf{X}_t^{(l)} = \text{DGCN}(\mathbf{Z}_t^{(l-1)}) = \sigma((\mathbf{A}\odot\mathbf{S}_t)\mathbf{Z}_t^{(l-1)}\mathbf{W}^{(l)}). \tag{7}$$

Note that with $\mathbf{S}_t$ the spatial dynamics captured by our model depend on the input while all the previous work consider the the spatial correlations among nodes to be invariant to the input. Our proposed dynamic graph convolution blocks aggregate neighbor information based on the varying correlation matrix decided by the inputs $\boldsymbol{\mathcal{Z}}^{(l-1)}$. Finally, we get a spatially informed output $\boldsymbol{\mathcal{X}}^{(l)} = (\mathbf{X}^{(l)}_{t-T_h+1}, \mathbf{X}^{(l)}_{t-T_h+2}, \ldots, \mathbf{X}^{(l)}_{t}) \in \mathbb{R}^{N \times d_{\text{model}} \times T_h}$.

4.2 Spatial-Temporal Decoder

The spatial-temporal decoder generates output sequences in an auto-regressive manner. To prevent from using the information of future subsequence, the masking mechanism is used in the decoder. The spatial-temporal decoder is composed of a stack of L' identical decoder layers. Each decoder layer consists of two temporal trend-aware multi-head attention blocks and a spatial dynamic GCN block. Specifically, the first temporal trend-aware multi-head self-attention block captures the correlation in the decoder sequence. To mask future information, following [50] the 1D convolutions on the queries and keys are replaced with the causal convolutions [16]. Because the causal convolution performs the convolution operation by only filtering on the positions to the left of the current position, it guarantees that the operation will not peek from the future in the output sequence. Figure 5 illustrates the difference between the causal convolution and the traditional 1D convolution. The second temporal trend-aware multi-head self-attention block is used to capture the correlations between the decoder sequence (queries) and the encoder output sequence (keys), in which the causal convolutions are applied on queries while the 1D convolutions are applied on keys.

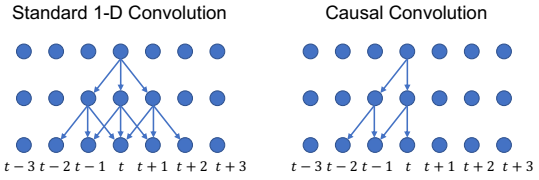


Fig. 5. The difference between the standard 1D convolution (left) and the causal convolution (right).

4.3 Handling Periodicity and Position Embedding

4.3.1 Handling Periodicity

In this work, we further consider two types of periodic patterns hidden in the traffic data, namely, the global periodicity and local periodicity. The global periodicity is due to the regularity of human activities, *e.g.*, a commuter departs from home at 8:00am every Monday, and thus the traffic conditions falling into the same time slot on the same day of a week tend to be similar. The local periodicity is often caused by the change of climate or weather, *e.g.*, the traffic speed in three days of heavy snowing significantly differs from that in the other days. To consider the two periodic patterns in forecasting the traffic over the next T_p time steps, we introduce another two data sources in addition to the historical records in the past T_h time steps.

Global periodic tensor. To capture the global periodicity, we consider the T_p slices of traffic records from the same day in the past w weeks as the present day, and this results in a tensor $\boldsymbol{\mathcal{X}}_g \in \mathbb{R}^{N \times C \times w * T_p}$. For instance, assume the time step is one hour and we wish to predict the traffic conditions of next four hours ($T_p = 4$) from 7:00am to 11:00am on Monday. We take the traffic records of 7:00am to 11:00am from the past three Mondays ($w = 3$), and we obtain a global periodic tensor $\boldsymbol{\mathcal{X}}_g \in \mathbb{R}^{N \times C \times 12}$.

Local periodic tensor. Analogously, to capture the local periodicity, we consider the T_p slices of traffic records from each of the past d consecutive days, which results in a tensor $\boldsymbol{\mathcal{X}}_l \in \mathbb{R}^{N \times C \times d * T_p}$. For instance, suppose again the time step is one hour and we intend to predict the traffic conditions of next four hours ($T_p = 4$) from 7:00am to 11:00am of the present day, we take the traffic records of 7:00am to 11:00am from the past two days ($d = 2$), and we obtain a local periodic tensor $\boldsymbol{\mathcal{X}}_l \in \mathbb{R}^{N \times C \times 8}$.

Once we obtain the global periodic tensor $\boldsymbol{\mathcal{X}}_g$ and local periodic tensor $\boldsymbol{\mathcal{X}}_l$, we concatenate them with the past T_h time steps tensor $\boldsymbol{\mathcal{X}}$, resulting in a new input tensor with shape $\mathbb{R}^{N \times C \times (w * T_p + d * T_p + T_h)}$ as the input of ASTGNN(p).

4.3.2 Temporal Position Embedding

In our temporal trend-aware module, the dynamics are entirely modeled by the self-attention mechanism. Since attention builds the dependency between inputs and target with the weighted sum function, the attention mechanism is completely agnostic to the order of symbols in the sequence. However, the order information plays an important role in time series modeling task since the nearby observations are often more correlated, *e.g.*, the traffic flow at 5:00pm is more informative for predicting the flow at 5:30 than that at 4:00pm. Hence, explicitly inducing the order bias to the model potentially results in a more accurate prediction. To this end, we equip each element representation in $\boldsymbol{\mathcal{X}}^{(0)}$ with a position embedding such that the nearby elements tend to possess close position embeddings. For simplicity, we choose the fixed position embedding [19] for an input element at position t and each vector-dimension $1 \leq d \leq d_{\text{model}}$ is presented as follows.

$$\begin{aligned} E_{\text{TP}}(t, 2d) &= \sin(t/10000^{2d/d_{\text{model}}}) \\ E_{\text{TP}}(t, 2d+1) &= \cos(t/10000^{2d/d_{\text{model}}}) \end{aligned} \tag{8}$$

where t is the relative index of each element in the input. As a side benefit, when the input includes the global periodic tensor and the local periodic tensor, introducing the temporal position embedding helps our model better recognize the relative position relationship among $\boldsymbol{\mathcal{X}}$, $\boldsymbol{\mathcal{X}}_l$ and $\boldsymbol{\mathcal{X}}_g$, event though they are simply concatenated.

4.3.3 Spatial Position Embedding

In Section 4.1.2 we present how to capture the dynamically changing correlations among different nodes (monitoring stations), which are caused by the evolving traffic conditions. However, one key observation we made is that, in addition to the dynamically changing traffic conditions, each spatial node is also associated with some static characteristics, which are mostly determined by the spatial features including the local topology of the node, and the

corresponding road types such as motorway, secondary way, living street, etc.

These static spatial characteristics are invariant over time but vary over the space, and thereby account for the spatial heterogeneity. They are also very informative for the traffic prediction, *e.g.*, the traveling speed of a motorway are often much faster than that of a living street.

However, the detailed characteristics of spatial nodes (*i.e.*, road types, road width, road speed limit, POIs) are often not available. In this situation, several possible ways can be employed to model the spatial heterogeneity on the graph. One direct way to represent the unique spatial heterogeneity of each node is one-hot encoding. However, the high-dimensional sparse feature not only results in expensive computation but also loses the graph structure information. An alternative way [52] is employing unsupervised graph embedding methods such as DeepWalk [53] to learn representations for spatial nodes. Although they can preserve the similarities among neighbors on the graph, mostly they fail to suit the specific supervised tasks. Inspired by the well-known practice of unsupervised pre-training [54], we also tried to learn the representations of nodes through unsupervised graph embedding techniques, and then use the learned representations as the initialization of the node embedding vectors which will be fine-tuned later based on supervised signals. But we did not observe any improvement in our preliminary experiments. Graph Laplacian Regularization [55] is another approach to learn representations preserving graph structure. It adds the graph Laplacian as an additional unsupervised regularization loss to the original supervised loss, but the hyper-parameter controlling the strength of the regularization needs carefully tuning.

Li et al [56] prove GCN is actually a special form of Laplacian smoothing, which enforces the representation of each node close to its neighbors. Therefore, to explicitly model the spatial heterogeneity while reflecting the graph structure information, we first assign each node an additional embedding vector, resulting in an initial spatial position embedding matrix $E_{\mathrm{SP}}^{(0)} \in \mathbb{R}^{N \times d_{\mathrm{model}}}$. Then a GCN layer is applied subsequently to perform Laplacian smoothing and get the final spatial position embedding matrix E_{SP}.

Both the temporal position embedding matrix E_{TP} and spatial position embedding matrix E_{SP} will be added to the raw representations $\mathcal{X}^{(0)}$, as illustrated in Figure 3. To the best of our knowledge, no existing method explicitly considers the spatial heterogeneity for the traffic forecasting problem. As we will show in Section 5.4.2.2, our proposed spatial position embedding is able to yield a fair model performance improvement.

5 Experiments

To evaluate the performance of our proposed model [1], we conducted extensive experiments on two categories of real-world spatial-temporal traffic graph datasets. One is about the highway traffic flow, which has four sub-datasets, and the other one is about the crowd flow of the metro system.

1. We will release the code as we do for our preliminary work ASTGCN [11].

5.1 Datasets

We evaluate our model on two types of traffic datasets whose statistical information is summarized in Table 1.

The first kind of datasets are about highway traffic flow in California, which is collected by the Caltrans Performance Measurement System (PeMS) [57] in real time every 30 seconds. The raw traffic flow data is aggregated into 5-minute interval. Geographic information about the sensor stations is recorded in the datasets. We construct four datasets from four districts respectively, namely PEMS03, PEMS04, PEMS07 and PEMS08. In our experiments, the four highway traffic networks are defined as undirected graphs and $C = 1$ denotes traffic flow. The second kind of datasets is about metro crowd flow of the Hangzhou metro system[2]. The raw crowd flow data is aggregated into 5-minute interval. In our experiments, the traffic network of the metro system is defined as an undirected graph and $C = 1$ denotes the inflow or the outflow.

The two types of datasets exhibit different spatial correlations due to their different generation process. On the highway traffic network, the vehicle flows are usually recorded by a collection of spatially-close traffic monitor stations. As a result, the traffic flows collected at these neighboring monitor stations typically show very strong correlation. In contrast, only the check-in and check-out events of the passengers are recorded in the metro system, and such recording gaps are usually much larger than that of traffic flow. Hence, the spatial correlation between the neighboring metro stations is greatly reduced. We evaluate the performance of ASTGNN via conducting experiments on the two types of datasets.

In the experiments, we aim to predict the data for the next hour (i.e., 12 steps).

TABLE 1
Dataset Description.

Data type	Datasets	# of sensors	Time range
Highway traffic flow	PEMS03	358	09/01/2018 - 11/30/2018
	PEMS04	307	01/01/2018 - 02/28/2018
	PEMS07	883	05/01/2017 - 08/31/2017
	PEMS08	170	07/01/2016 - 08/31/2016
Metro crowd flow	HZME	80	01/01/2019 - 01/26/2019

5.2 Baseline Methods

- VAR [58]: Vector Auto-Regression is an advanced time series model, which captures the pairwise relationships among multiple time series.
- SVR [59]: Support Vector Regression utilizes a linear support vector machine to perform regression.
- LSTM [60]: Long Short-Term Memory network, a special RNN model.
- DCRNN [6]: Diffusion Convolutional Recurrent Neural Network employs diffusion graph convolutional networks and GRU based on seq2seq to predict traffic graph series data.

2. https://tianchi.aliyun.com/competition/entrance/231708/information

- STGCN [45]: Spatial-Temporal Graph Convolutional Network uses ChebNet in the spatial dimension and 2D convolutional networks in the temporal dimension to model the correlations in spatial-temporal graph data.
- ASTGCN [11]: Attention Based Spatial-Temporal Graph Convolutional Networks designs spatial attention and temporal attention mechanisms to model spatial and temporal dynamics.
- Graph WaveNet [10]: Graph WaveNet combines graph convolutions with temporal convolutions to capture spatial-temporal dependencies.
- STSGCN [12]: Spatial-Temporal Synchronous Graph Convolutional Network proposes a new kind of convolution operation to capture spatial and temporal correlations simultaneously.

DCRNN and STGCN are two representative baselines that are widely used for traffic forecasting. ASTGCN is the method in our preliminary work. Graph WaveNet and STSGCN are two recent approaches.

5.3 Settings

We split all datasets at ratio $6:2:2$ into training sets, validation sets, and test sets by the time. We normalize all data into range $[-1, 1]$ with Min-Max method and feed the normalized data into the model, which is optimized by reverse mode automatic-differentiation and Adam [61]. During training, we feed the historical data into the encoder and the decoder generates predictions given previous ground truth observations. Thus, all the attention in encoder and decoder can be parallelized. During training, we feed the historical data into the encoder and the decoder generates predictions given previous ground truth observations. Thus, all the attention operations in the encoder and decoder can be parallelized. During testing, the ground truth fed into the decoder is replaced by the prediction generated by the model itself. To mitigate the discrepancy between the inputs to the decoder in the training stage and testing stage, we fine-tune the model for several epochs on the training sets by using the predictions as the input to the decoder.

For evaluation, we re-transform the predicted values back to the actual values and compare them with the ground truth. Mean absolute error (MAE), root mean square error (RMSE) and mean absolute percentage error (MAPE) are used as the evaluation metrics. All the experiments are repeated 5 times, and the means and standard deviations are reported.

We implemented the ASTGNN model based on the PyTorch[3] framework. We choose Mean absolute error (MAE) as the loss function. The hyperparameters and the best models are determined by the performance on the validation sets. The model dimension d_{model} is 64 and the number of attention heads h is 8. Learning rate lr is 0.001. Other detailed settings of the ASTGNN model for the five datasets are described in Table 2.

To make a fair comparison, for all methods, we first use the historical data over the past hour (12 steps) to predict the next hour's data, to show the effectiveness of ASTGNN. Then we evaluate how the model performance improves when the periodicity of traffic data is considered explicitly. ASTGNN(p) denotes our model with global or local periodic tensors in the input.

3. https://pytorch.org

TABLE 2
Settings of the ASTGNN model in five datasets.

Dataset	encoder	decoder	convolution
	# layers L	# layers L'	kernel size k
PEMS03	3	3	3
PEMS04	4	4	3
PEMS07	3	3	3
PEMS08	4	4	3
HZME (inflow)	4	4	3
HZME (outflow)	3	3	5

5.4 Comparison and Analysis of Results on the Highway Traffic Flow Prediction

Table 3 shows the average results over the next hour (12 steps predictions). Without considering the periodicity, our ASTGNN outperforms all the baseline methods on all the datasets for most of cases. In PEMS04 (resp. PEMS07), ASTGNN improves the state-of-the-art method Graph WaveNet by 3.9%, 2.5%, 7.1% (resp. 2.9%, 0.4%, 2.2%) in terms of MAE, RMSE and MAPE, respectively. In PEMS03, ASTGNN has the best performs in terms of MAE and RMSE. In PEMS08, ASTGNN has the best performs in terms of MAE and MAPE.

Figure 6 shows the effect of increasing prediction intervals on prediction performance for different methods. In general, as the prediction intervals become larger, the tasks becomes more difficult, and thus the performance of all models drops. However, the performance of our ASTGNN degrades the least in most cases, *i.e.*, the advantage of our ASTGNN becomes more evident for long-term forecast.

SVR and LSTM only take the temporal features into consideration, but do not consider the spatial correlations, which are also important for the spatial-temporal traffic forecasting. Therefore, their forecasting performance is the worst. Although VAR models both spatial and temporal correlations among multiple time series, its representation ability to capture nonlinear and dynamic spatial-temporal correlations is weak. Thus, its forecasting performance fluctuates significantly. Especially, as shown in Table 3, the average forecasting performance of VAR drops significantly on PEMS07 which has a large number of nodes. Since the performance of VAR on PEMS07 is much worse than other methods, we do not show it in Figure 6 for clarity.

DCRNN is a typical RNN-based method for spatial-temporal graph data forecasting. Limited by the ability of RNN to capture long-term temporal correlations, its forecasting accuracy is much lower than our method ASTGNN, especially for the long-term predictions. STGCN, ASTGCN, Graph WaveNet and STSGCN are four typical CNN-based methods, which employ 1D CNN or TCN along the temporal dimension to capture the temporal correlations. Limited by the size of the convolution kernel, it is hard for 1D CNN to attend long-term temporal information. Besides, although

TABLE 3
Performance comparison on the four highway traffic flow datasets.
(the best results are in bold and * denotes the second-best results. † denotes our models.)

Baseline methods											
Datasets	Metrics	VAR	SVR	LSTM	DCRNN	STGCN	ASTGCN(r)	Graph WaveNet	STSGCN	**ASTGNN**†	**ASTGNN(p)**†
PEMS03	MAE	21.08	22.01 ± 0.07	20.62 ± 0.19	18.39 ± 0.17	18.28 ± 0.39	17.85 ± 0.45	14.79 ± 0.08	17.51 ± 0.13	14.78* ± 0.05	**14.55 ± 0.07**
	RMSE	34.75	35.28 ± 0.08	33.54 ± 0.34	30.56 ± 0.17	30.73 ± 0.78	29.88 ± 0.65	25.51 ± 0.17	29.05 ± 0.40	25.00* ± 0.18	**24.96 ± 0.31**
	MAPE (%)	22.35	22.93 ± 1.09	28.94 ± 2.76	20.22 ± 2.83	17.52 ± 0.32	17.65 ± 0.79	14.32* ± 0.24	16.92 ± 0.22	14.79 ± 0.22	**13.66 ± 0.14**
PEMS04	MAE	23.75	28.66 ± 0.01	26.81 ± 0.31	23.65 ± 0.04	22.27 ± 0.18	22.42 ± 0.19	19.36 ± 0.02	21.08 ± 0.14	18.60* ± 0.06	**18.44 ± 0.08**
	RMSE	36.66	44.59 ± 0.02	40.74 ± 0.17	37.12 ± 0.07	35.02 ± 0.19	34.75 ± 0.19	31.72 ± 0.13	33.83 ± 0.27	**30.91 ± 0.22**	31.02* ± 0.18
	MAPE (%)	18.09	19.15 ± 0.04	22.33 ± 1.60	16.05 ± 0.10	14.36 ± 0.12	15.87 ± 0.36	13.31 ± 0.19	13.88 ± 0.07	**12.36 ± 0.11**	12.37* ± 0.08
PEMS07	MAE	101.20	32.97 ± 0.98	29.71 ± 0.09	23.60 ± 0.05	27.41 ± 0.45	25.98 ± 0.78	21.22 ± 0.24	23.99 ± 0.14	20.62* ± 0.12	**19.26 ± 0.17**
	RMSE	155.14	50.15 ± 0.15	45.32 ± 0.27	36.51 ± 0.05	41.02 ± 0.58	39.65 ± 0.89	34.12 ± 0.18	39.32 ± 0.31	34.00* ± 0.21	**32.75 ± 0.25**
	MAPE (%)	39.69	15.43 ± 1.22	14.14 ± 1.00	10.28 ± 0.02	12.23 ± 0.38	11.84 ± 0.69	9.07 ± 0.20	10.10 ± 0.08	8.86* ± 0.10	**8.54 ± 0.19**
PEMS08	MAE	22.32	23.25 ± 0.01	22.19 ± 0.13	18.22 ± 0.06	18.04 ± 0.19	18.86 ± 0.41	15.07 ± 0.17	17.10 ± 0.04	15.00* ± 0.35	**12.72 ± 0.09**
	RMSE	33.83	36.15 ± 0.02	33.59 ± 0.05	28.29 ± 0.09	27.94 ± 0.18	28.55 ± 0.49	23.85* ± 0.18	26.83 ± 0.06	24.70 ± 0.53	**22.60 ± 0.13**
	MAPE (%)	14.47	14.71 ± 0.16	18.74 ± 2.79	11.56 ± 0.04	11.16 ± 0.10	12.50 ± 0.66	9.51 ± 0.22	10.90 ± 0.05	9.50* ± 0.11	**8.78 ± 0.20**

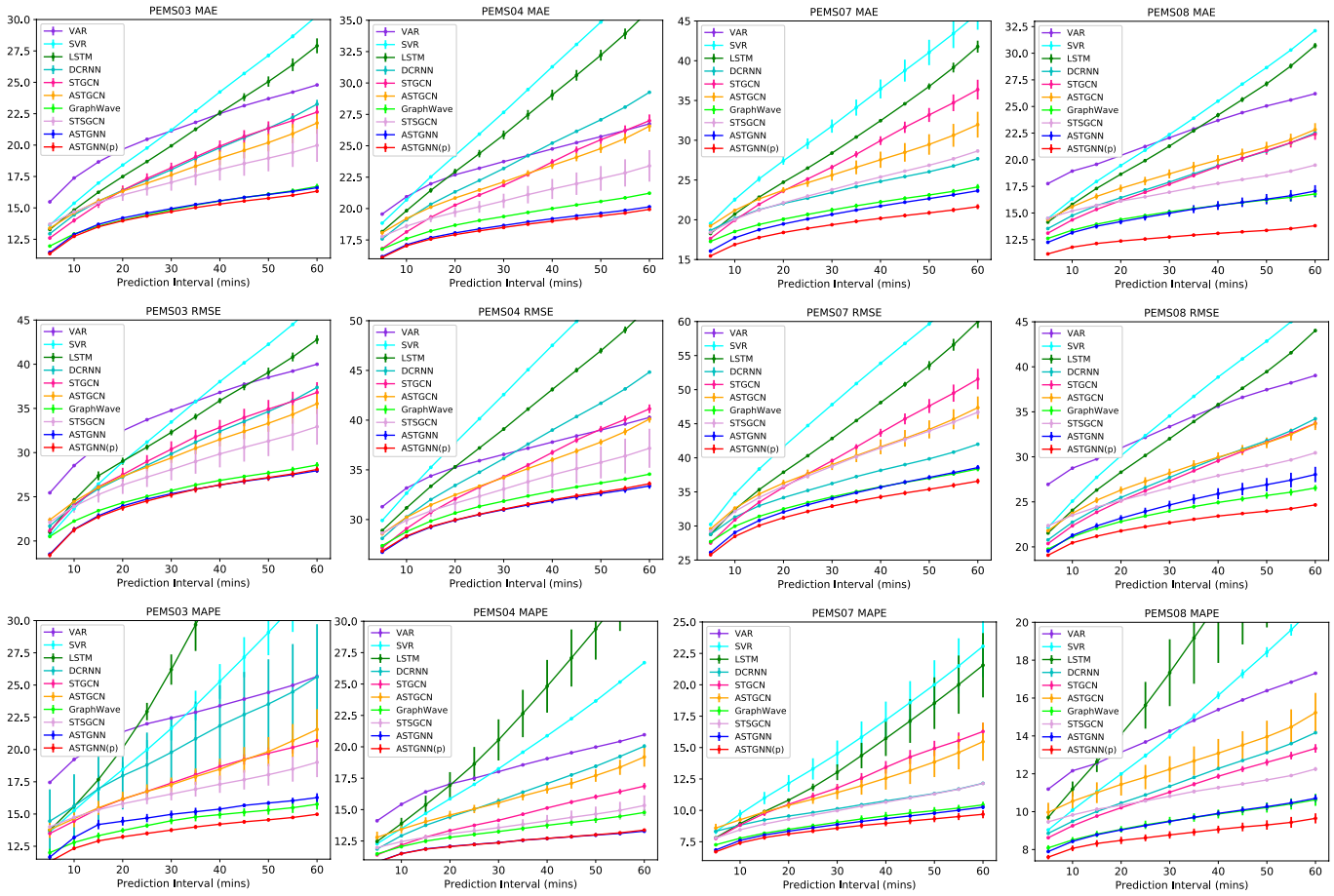


Fig. 6. Performance changes of different methods on the four datasets as the prediction interval increases.

TCN employs dilated convolutions to enable an exponentially large receptive field, compared to the self-attention mechanism, TCN still requires a stack of $O(log_k(T_h))$ convolutional layers to connect any two positions in the sequence, where k is the kernel size of convolutions [19]. Thus, it is also hard for them to make accurate long-term prediction [20]. In contrast, in the temporal dimension, ASTGNN utilizes attention mechanisms to attend information at all time slices for data at each time slice, so that ASTGNN can well capture long-term correlations.

ASTGCN in our preliminary work is the only baseline in which the parameters capturing the dynamics in traffic data are dependent on the input data while the parameters in all the other baseline methods are invariant to the input after being learned. ASTGCN employs traditional attention mechanism in both the spatial and temporal dimensions to capture the dynamics in traffic data. However, as discussed in Section 4.1.1, traditional self-attention mechanism might suffer from the mismatching issue in learning sequence representations. In contrast, ASTGNN utilizes a novel trend-aware self-attention mechanism in the temporal dimension to flexibly model the dynamics of traffic data and applies self-attention mechanism on the adjacency matrix to assign

importance to neighbors dynamically according to the inputs. By elaborately modeling the dynamics of traffic data, ASTGNN achieves the best forecasting performance.

5.4.1 Explicit Periodicity Modeling

Figure 1-(1) shows that traffic data exhibits strong periodicity over days and weeks. ASTGNN takes such periodic patterns into consideration to improve forecasting accuracy. To evaluate the effectiveness of the component for periodicity, we next perform experiments on ASTGNN(p) where letter p stands for *periodicity*. The inputs to ASTGNN(p) consist of $\boldsymbol{\mathcal{X}}_g$, $\boldsymbol{\mathcal{X}}_l$ and the original $\boldsymbol{\mathcal{X}}$ as described in Section 4.3.1.

In ASTGNN(p), T_h still equals to T_p, that is one hour. We consider the hyperparameters in the local and global periodic tensors $d, w \in \{0, 1, 2\}$. The settings of hyperparameters are shown in Table 4, which are determined by the performance on the validation sets. The right two columns in Table 3 compares the average results (over the next hour) of ASTGNN and ASTGNN(p). Figure 6 also shows the detailed comparison as the prediction interval increases. The results show that explicitly modeling the periodicity further improves the forecasting accuracy in general. The improvements are more significant on PEMS07 and PEMS08.

TABLE 4
Settings of the ASTGNN(p) model in five datasets.

Dataset	d	w	encoder	decoder	convolution
			# layers L	# layers L'	kernel size k
PEMS03	1	0	3	3	3
PEMS04	0	1	4	4	3
PEMS07	1	1	3	3	3
PEMS08	1	1	4	4	3
HZME (inflow)	0	1	4	4	3
HZME (outflow)	0	1	4	4	3

5.4.2 Ablation experiments

To further evaluate the effects of different components in ASTGNN, we conduct ablation experiments and analyze experimental results on the PEMS03 dataset.

Temporal Dimension. In the temporal dimension, we design three variant versions of ASTGNN, including:

- ASTGNN-noTE: It removes the temporal position embedding to study the benefits of modeling order information of the sequences.
- ASTGNN-conv: It replaces the temporal trend-aware self-attention layer with the convolutions (1D conv in the encoder and causal conv in the decoder) to study the advantages of self-attention mechanism over convolutions in capturing the temporal dependencies.
- ASTGNN-noTrA: It replaces the temporal trend-aware multi-head self-attention block with tradition multi-head self-attention operation to study the usefulness of considering trends when forecasting.

All the variant models have the same settings as ASTGNN except the differences mentioned above. Figure 7 gives the average prediction results of the models over next hour and the detailed results of prediction performance at each time slice. By comparing ASTGNN-noTE and ASTGNN, we can see that the temporal position embedding layer is an important component. ASTGNN-noTE, which performs attention operations without position embedding and order information, perfoms much worse than ASTGNN. Moreover, ASTGNN-conv performs worse than ASTGNN-noTrA, indicating that the self-attention mechanism is superior to convolutions in capturing the temporal correlations. ASTGNN is better than ASTGNN-noTrA and their performance differences become larger as the prediction intervals increase. This result proves the usefulness of modeling the local trend in using multi-head self-attention for traffic prediction, especially for long-term forecasting.

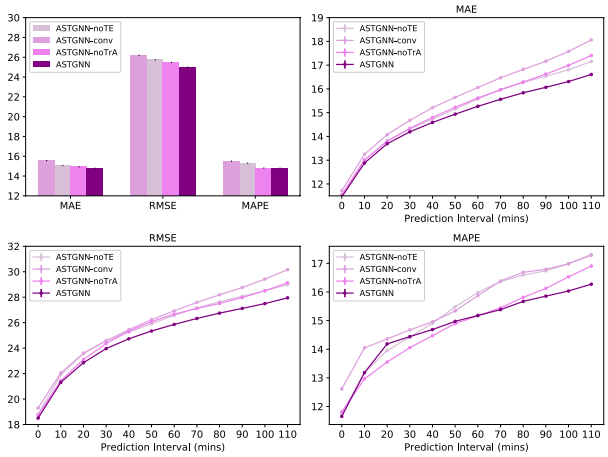


Fig. 7. Components analysis in the temporal dimension.

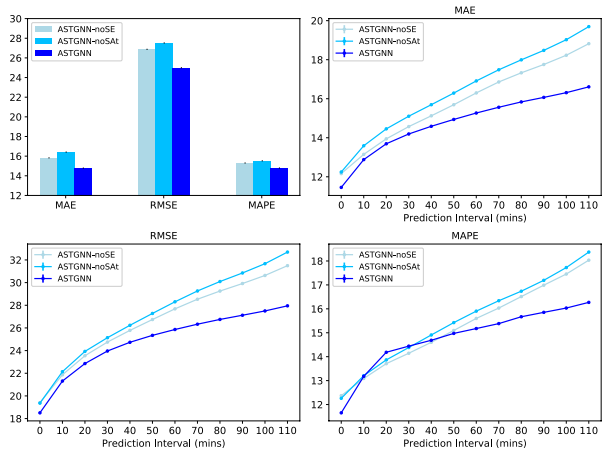


Fig. 8. Components analysis in the spatial dimension.

Spatial Dimension. In the spatial dimension, we design two variant versions of ASTGNN, including:

- ASTGNN-noSE: It removes the spatial position embedding to study the benefits of modeling the inherent static spatial characteristics of the traffic network.
- ASTGNN-noSAt: It removes the spatial self attention layer to study the usefulness of dynamically adjusting spatial correlation strength instead of only based on the static topological relationship.

All the variant models have the same settings as ASTGNN except the aforementioned differences. Figure 8 shows

TABLE 5
Performance comparison on the metro crowd flow dataset.
(the best results are in bold and * denotes the second-best results. † denotes our models.)

Baseline methods	MAE		RMSE		MAPE (%)	
Metrics	inflow	outflow	inflow	outflow	inflow	outflow
VAR	17.65	22.35	28.10	37.96	58.07	96.68
SVR	21.94 ± 0.0173	25.59 ± 0.1187	40.73 ± 0.0205	50.07 ± 0.1732	49.40 ± 0.0679	91.71 ± 3.1788
LSTM	22.53 ± 0.5089	26.18 ± 0.3193	39.33 ± 0.3506	48.91 ± 0.4458	60.12 ± 2.4417	103.06 ± 8.5229
DCRNN	12.25 ± 0.1316	18.02 ± 0.1579	20.91 ± 0.3290	31.45 ± 0.3879	25.53 ± 0.3811	**66.98± 1.6475**
STGCN	12.88 ± 0.2810	19.12 ± 0.2312	22.86 ± 0.3884	33.12 ± 0.3596	29.66 ± 1.4970	73.66 ± 1.4909
ASTGCN	13.10 ± 0.4733	19.35 ± 0.5071	23.23 ± 0.8127	33.20 ± 1.0714	33.29 ± 3.6336	88.75 ± 3.9984
Graph WaveNet	11.20* ± 0.1116	17.50* ± 0.1234	19.73* ± 0.4610	**30.65 ± 0.4059**	23.75* ± 0.7082	73.65 ± 2.7181
STSGCN	12.85 ± 0.1006	18.74 ± 0.1289	23.20 ± 0.3773	33.12 ± 0.4260	28.02 ± 0.1947	76.85 ± 1.0094
ASTGNN†	11.46 ± 0.0841	17.94 ± 0.1093	20.84 ± 0.2498	31.91 ± 0.3163	24.42 ± 0.3029	72.46 ± 2.4177
ASTGNN(p)†	**10.94 ± 0.0393**	**17.47 ± 0.0310**	**18.89 ± 0.1081**	30.78* ± 0.0774	**23.33 ± 0.1400**	70.52*± 0.2739

the results. We can observe that ASTGNN-noSE is worse than ASTGNN and this demonstrates that explicitly considering static spatial characteristics contributes to capturing the spatial heterogeneity in traffic data. ASTGNN-noSAt performs much worse than ASTGNN. This demonstrates the usefulness of dynamically adjusting spatial correlation strength in ASTGNN.

5.4.3 Effect of Different Network Configurations

To further investigate the influences of hyper-parameter settings and how the residual connection and layer normalization affect the model performance, we conduct experiments with different network configurations. Except for the studied varying factor, all the models follow the same settings as described in Section 5.3. Figure 9 illustrates the results. Generally, ASTGNN is not sensitive to hyper-parameter settings. Increasing d_{model}, L and h can slightly improve the performance. Additionally, we find that (1) if the model is neither equipped with residual connections nor normalization layers, it is hard to train and the loss does not convergence; (2) The model with residual connections and normalization layers together yields the best performance.

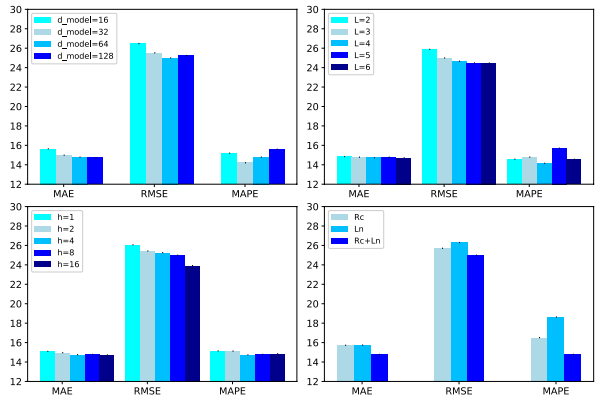


Fig. 9. Network configuration analysis. The four figures respectively show the effect of different network configurations, including the model dimension d_{model}, the number of encoder/decoder layers L, the number of attention heads h, whether the model is equipped with the residual connection or layer normalization.

5.4.4 Time Cost Study

In the temporal trend-aware multi-head self-attention block, the computational complexity is $O(T_h^2 d_{\text{model}})$ and $O(kT_h d_{\text{model}}^2)$ for the multi-head self-attention operation and the convolution operation, respectively, for each node; the computational complexity of the spatial dynamic GCN block is $O(Md_{\text{model}})$ and $O(N^2 d_{\text{model}})$ for the GCN operation and the spatial attention operation, respectively, for every time step.

The training time of ASTGNN grows linearly with the number of data-points. It takes around 0.35s per batch, with a batch size 16 on PEMS03 using a Tesla V100 GPU card. In the prediction stage, ASTGNN can process a sample batch with size 16 within 0.8 seconds through auto-regressive generation for the next 12-steps forecast.

5.5 Comparison and Analysis of Results on the Metro Crowd Flow Prediction

Table 5 shows the average results over the next hour (12 steps predictions). On the metro crowd flow dataset, ASTGNN is still competitive. Without modeling the periodicity, ASTGNN, Graph WaveNet achieve the similar best performance and ASTGNN shows advantages on MAPE. When taking the periodicity into consideration, ASTGNN(p) is able to improve all the metrics further.

In addition, we find that the improvements of ASTCNN and ASTGNN(p) over other baselines on metro crowd flow dataset are not as obvious as those on the highway traffic flow datasets. The reason could be that the excellent performance of ASTCNNs can be largely explained by its great ability in capturing the dynamical correlations among the neighboring spatial nodes and consecutive time steps; however, as discussed in Section 5.1, the recording gaps of the metro crowd flow are much larger than that of the traffic flows, which leads to weak spatial correlations among the neighboring metro stations and subsequently, a degraded performance of ASTGNNs. But it is worth noting that even in the weak spatial correlation cases, ASTGNNs are still very competitive.

The experimental results on the two kinds of datasets show that the proposed ASTGNNs are more suitable for the prediction tasks on spatial-temporal graph traffic data with strong neighboring spatial-temporal correlations.

6 Conclusions

In this paper, we propose an effective deep learning based neural network ASTGNN for traffic forecasting. In ASTGNN, we propose novel approaches in modelling the dynamics of traffic data along both the temporal and spatial dimensions, as well as consider the periodicity and spatial heterogeneity of traffic data. Specifically, we design a trend-aware multi-head attention mechanism specialized for time series prediction task, which captures local context in time series. To capture the dynamics along the spatial dimension, we develop a novel dynamic GCN. It can adaptively adjust the spatial correlation strength. Moreover, we explicitly model the periodicity and spatial heterogeneity of traffic data. Experiments on five real-world traffic datasets demonstrate ASTGNN is superior to the state-of-the-art baselines.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities (Grant No. 2019JBM024).

References

[1] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results," *Journal of transportation engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[2] Z. Lu, C. Zhou, J. Wu, H. Jiang, and S. Cui, "Integrating granger causality and vector auto-regression for traffic prediction of large-scale wlans." *KSII Transactions on Internet & Information Systems*, vol. 10, no. 1, 2016.

[3] C.-H. Wu, J.-M. Ho, and D.-T. Lee, "Travel-time prediction with support vector regression," *IEEE transactions on intelligent transportation systems*, vol. 5, no. 4, pp. 276–281, 2004.

[4] J. Van Lint and C. Van Hinsbergen, "Short-term traffic and travel time prediction models," *Artificial Intelligence Applications to Critical Transportation Issues*, vol. 22, no. 1, pp. 22–41, 2012.

[5] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *arXiv preprint arXiv:1901.00596*, 2019.

[6] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *ICLR*, 2017.

[7] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson, "Structured sequence modeling with graph convolutional recurrent networks," in *International Conference on Neural Information Processing*. Springer, 2018, pp. 362–373.

[8] W. Chen, L. Chen, Y. Xie, W. Cao, Y. Gao, and X. Feng, "Multi-range attentive bicomponent graph convolutional network for traffic forecasting," *Proceedings of the AAAI Conference on Artificial Intelligence3*, 2020.

[9] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[10] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in *IJCAI*, 2019.

[11] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 922–929.

[12] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[13] S. Strogatz, "Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering (studies in nonlinearity)," 2001.

[14] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, 2013, pp. 1310–1318.

[15] D. Li, J. Zhang, Q. Zhang, and X. Wei, "Classification of ecg signals based on 1d convolution neural network," in *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*. IEEE, 2017, pp. 1–6.

[16] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[17] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[18] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[20] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber *et al.*, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," 2001.

[21] Z. Lin, J. Feng, Z. Lu, Y. Li, and D. Jin, "DeepSTN+: Context-aware spatial-temporal neural network for crowd flow prediction in metropolis," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1020–1027.

[22] A. Zonoozi, J.-j. Kim, X.-L. Li, and G. Cong, "Periodic-CRN: A convolutional recurrent model for crowd density prediction with recurring periodic patterns." in *IJCAI*, 2018, pp. 3732–3738.

[23] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li, "Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 5668–5675.

[24] Z. Jiang, "A survey on spatial prediction methods," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 9, pp. 1645–1664, 2018.

[25] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1624–1639, 2011.

[26] J. Zhang, Y. Zheng, J. Sun, and D. Qi, "Flow prediction in spatio-temporal networks based on multitask deep learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 3, pp. 468–478, 2019.

[27] Y. Li and Y. Zheng, "Citywide bike usage prediction in a bike-sharing system," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 6, pp. 1079–1091, 2019.

[28] T. Anwar, C. Liu, H. L. Vu, M. S. Islam, and T. Sellis, "Capturing the spatiotemporal evolution in road traffic networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 8, pp. 1426–1439, 2018.

[29] L. Lin, J. Li, F. Chen, J. Ye, and J. Huai, "Road traffic speed prediction: a probabilistic model fusing multi-source data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 7, pp. 1310–1323, 2017.

[30] X. Zhan, Y. Zheng, X. Yi, and S. V. Ukkusuri, "Citywide traffic volume estimation using trajectory data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 2, pp. 272–285, 2016.

[31] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[32] J. Zhang, Y. Zheng, D. Qi, R. Li, X. Yi, and T. Li, "Predicting city-wide crowd flows using deep spatio-temporal residual networks," *Artificial Intelligence*, vol. 259, pp. 147–166, 2018.

[33] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, and J. Ye, "Deep multi-view spatial-temporal network for taxi demand prediction," in *AAAI Conference on Artificial Intelligence*, 2018, pp. 2588–2595.

[34] C. Zhang and P. Patras, "Long-term mobile traffic forecasting using deep spatio-temporal neural networks," in *Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2018, pp. 231–240.

[35] S. Guo, Y. Lin, S. Li, Z. Chen, and H. Wan, "Deep spatial-temporal 3d convolutional neural networks for traffic data forecasting," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[36] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: A graph multi-attention network for traffic prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[37] X. Geng, Y. Li, L. Wang, L. Zhang, Q. Yang, J. Ye, and Y. Liu, "Spatiotemporal multi-graph convolution network for ride-hailing

demand forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 3656–3663.
[38] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.
[39] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *ieee Computational intelligenCe magazine*, vol. 13, no. 3, pp. 55–75, 2018.
[40] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," *arXiv preprint arXiv:1901.02860*, 2019.
[41] M. Z. Hossain, F. Sohel, M. F. Shiratuddin, and H. Laga, "A comprehensive survey of deep learning for image captioning," *ACM Computing Surveys (CSUR)*, vol. 51, no. 6, pp. 1–36, 2019.
[42] L. Deng, G. Hinton, and B. Kingsbury, "New types of deep neural network learning for speech recognition and related applications: An overview," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 8599–8603.
[43] H. Song, D. Rajan, J. J. Thiagarajan, and A. Spanias, "Attend and diagnose: Clinical time series analysis using attention models," in *Thirty-second AAAI conference on artificial intelligence*, 2018.
[44] S. Huang, D. Wang, X. Wu, and A. Tang, "Dsanet: Dual self-attention network for multivariate time series forecasting," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 2129–2132.
[45] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *IJCAI*, 2018.
[46] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.
[47] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to sequence learning with neural networks," *Advances in NIPS*, 2014.
[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[49] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.
[50] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and X. Yan, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," in *Advances in Neural Information Processing Systems*, 2019, pp. 5244–5254.
[51] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
[52] D. Xu, H. Dai, Y. Wang, P. Peng, Q. Xuan, and H. Guo, "Road traffic state prediction based on a graph embedding recurrent neural network under the scats," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 29, no. 10, p. 103125, 2019.
[53] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 701–710.
[54] Y. Li, K. Fu, Z. Wang, C. Shahabi, J. Ye, and Y. Liu, "Multi-task representation learning for travel time estimation," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1695–1704.
[55] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in neural information processing systems*, 2002, pp. 585–591.
[56] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
[57] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, "Freeway performance measurement system: mining loop detector data," *Transportation Research Record: Journal of the Transportation Research Board*, no. 1748, pp. 96–102, 2001.
[58] E. Zivot and J. Wang, "Vector autoregressive models for multivariate time series," *Modeling Financial Time Series with S-Plus®*, pp. 385–429, 2006.
[59] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," in *Advances in neural information processing systems*, 1997, pp. 155–161.
[60] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[61] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

Shengnan Guo received the B.S. degree in computer science from Beijing Jiaotong University, Beijing, China, in 2015.

She is currently working toward the Ph.D. degree in the School of Computer and Information Technology, Beijing Jiaotong University. Her interest falls in area of deep learning and data mining, especially their applications in spatial-temporal data mining.

Youfang Lin received the Ph.D. degree in signal and information processing from Beijing Jiaotong University, Beijing, China, in 2003.

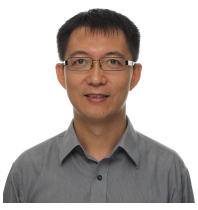
He is a Professor with the School of Computer and Information Technology, Beijing Jiaotong University. His main fields of expertise and current research interests include big data technology, intelligent systems, complex networks, and traffic data mining.

Huaiyu Wan received the Ph.D. degree in computer science and technology from Beijing Jiaotong University, Beijing, China, in 2012.

He is an Associate Professor with the School of Computer and Information Technology, Beijing Jiaotong University. His current research interests focus on spatio-temporal data mining, social network mining, and user behavior analysis.

Xiucheng Li is a Ph.D. student at School of Computer Science and Engineering, Nanyang Technological University.

At present, he focuses on representation learning, differentiable generative models (differentiable functions supported hierarchical Bayesian models, variational inference, Hamilton Monte Carlo), and in particular, extending their applications in spatial and temporal data analytics.



Gao Cong received the Ph.D. degree from the National University of Singapore, in 2004.

He is a professor with Nanyang Technological University, Singapore. Before he relocated to Singapore, he worked with Aalborg University, Microsoft Research Asia, and the University of Edinburgh. His current research interests include geo-textual data management and data mining.