A Dynamic Spatial-temporal Deep Learning Framework for Traffic Speed Prediction on Large-scale Road Networks

Ge Zheng[a] (gzheng@bournemouth.ac.uk), Wei Koong Chai[a] (wchai@bournemouth.ac.uk), Vasilis Katos[a] (vkatos@bournemouth.ac.uk)

[a] Department of Computing and Informatics, Bournemouth University, Fern Barrow, Poole, BH12 5BB, Dorset, United Kingdom

Corresponding Author:
Ge Zheng
Department of Computing and Informatics, Bournemouth University, Fern Barrow, Poole, BH12 5BB, Dorset, United Kingdom
Phone Number: +44 07513627614
Email: gzheng@bournemouth.ac.uk

A Dynamic Spatial-temporal Deep Learning Framework for Traffic Speed Prediction on Large-scale Road Networks

Ge Zheng[a], Wei Koong Chai[a], Vasilis Katos[a]

[a]*Department of Computing and Informatics, Bournemouth University, Fern Barrow, Poole, BH12 5BB, Dorset, United Kingdom*

Abstract

Traffic prediction plays a crucial role in an intelligent transportation system (ITS) for enabling advanced transportation management and services. In this paper, we address the problem of multi-step traffic speed prediction, including both short- and long-term predictions. We assert that it is important to consider not just the *fixed spatial* dependency of the road network (i.e., the connections between road segments) but also the *dynamic spatial* dependency of traffic within the static topology that intertwines with the temporal evolution of traffic condition across the entire network. We propose a novel deep learning model, named Self-Attention Graph Convolutional Network with Spatial, Sub-spatial and Temporal blocks (SAGCN-SST) model, that specifically capture such complex dynamic spatial-temporal processes. In SAGCN-SST, we integrate self-attention mechanism into graph convolutional networks in a novel framework design while using a sequence-to-sequence model in an encoder-decoder architecture for extracting long-temporal dependency of traffic speed. Two real-world datasets with frequent traffic congestion and accidents from large-scale road networks (i.e., Seattle and Los Angeles) are used to train and test our model. Our experiment results indicate that the proposed deep learning model consistently achieves the most accurate predictions (higher than 98% accuracy on both datasets for the short- and long-term predictions) when compared against well-known existing models in recent literature. The results also indicate that SAGCN-SST is robust against emergent traffic situations.

Keywords: Intelligent transportation system, traffic prediction, deep learning, large-scale road networks

1. Introduction

Traffic prediction is an important function within an Intelligent Transportation System (ITS). Accurate traffic prediction offers ITS the ability to manage the transportation network in a more efficient manner and thus alleviating traffic congestion, reducing road accidents and protecting environment via carbon emission reduction. Via synchronous traffic information system, it also allows ITS to provide better services to public in terms of minimizing travel time and cost (Wang, 2010). These are especially crucial as there is an increasing trend in road traffic, exacerbating the traffic congestion problems. However, predicting traffic is known to be a challenging task with complex inter-dependencies temporally and spatially.

Temporally, it is found that there is non-linear temporal dynamics of traffic flows over time depending on the changing road conditions (e.g., (Zhang, 2003)). Furthermore, traffic data also show periodic patterns (e.g., weekly and seasonal changes). Thus, current traffic status depends not only on the immediate previous epochs but may also be affected by longer periodic patterns or trends. Hence, both long- and short-temporal dependencies in traffic time sequences should be taken into account when making traffic predictions.

Since road networks are inherently spatial networks, the traffic flowing within the network logically depends on the topological structure of the road interconnections. Local spatial dependency can be seen from observing that the traffic flow at one road segment is affected by both its immediate upstream and downstream road segments. Furthermore, since the vehicles are traveling within the same physical road network, the traffic condition of different road segments across the network are inter-dependent. This can be exemplified in various gridlock phenomena taking place in urban cities during peak hours whereby congestion at one road segment quickly cascade and spread to other locations.

Early attempts on traffic prediction focus on temporal dependencies of traffic over time. Both long- and short-term dependencies in traffic time sequences were studied. Spatial dependencies were later jointly considered, initially focusing on capturing dependencies from immediate upstream and downstream road segments and more recently, taking into account the influence of the entire road network topology. We review the evolution of the prediction models in the literature in Section 2. However, the literature has mostly considered that the physical road network topology (i.e., the road segments and their connectivity) is fixed and neighboring nodes contribute equally to the future traffic status of the targeted node (we refer this as *fixed spatial* dependency). In this work, we argue that each neighboring node is distinct and has different influence to the targeted node (i.e., the degree of the spatial dependency varies from one neighbor to another). In the space dimension, intuitively, closer neighboring nodes have stronger influence while traffic status at road segments further away gradually have less impact. However, this relationship is not strict as it also depends on the local connectivity of the different neighboring nodes and it is possible to have a node further away having greater influence on the targeted node. In the time dimension, a congestion lasting longer period should have wider spatial impact and thus, a road segment will be more influenced by neighboring nodes further away if the network suffers comparatively long congestion period and vice versa. As such, we assert that capturing and quantifying such *dynamic spatial* dependency is important to further improve traffic prediction accuracy.

Building on the most recent developments in deep learning, we thus develop a novel prediction model, named Self-Attention Graph Convolutional Network with Spatial, Sub-spatial and Temporal blocks (SAGCN-SST), and address multi-step traffic speed prediction problem for large-scale road networks. Apart from considering fixed spatial and long temporal dependencies, the learning architecture in our SAGCN-SST is designed to also simultaneously capture the dynamic spatial dependency for both short- and long-term traffic prediction. We integrate a self-attention mechanism into our graph convolutional layers to capture how different neighboring nodes contribute to the future traffic status

of the targeted node. In our framework, we use parallel sub-blocks for different neighborhoods, avoiding increasing model depth and complexity. We validate our proposed framework using two real-world traffic datasets from large-scale road networks (Seattle and Los Angeles) and also conduct a comparative study across well-known existing models in recent literature. The results indicate our SAGCN-SST performs better than other leading models in the literature.

The rest of this paper is organized as follows. Firstly, we review in Section 2 the related work, highlighting how traffic prediction approaches have evolved over time. We define our traffic prediction problem in Section 3. In Section 4. we detail the design rationale and the architecture of our novel deep learning framework. Section 5 evaluates our SAGCN-SST and compares it with well-known existing models in recent literature on two real-world large-scale road network datasets. Finally, we conclude our work in Section 6.

2. Related Work

Traffic prediction models in the literature have evolved over the years to be increasingly sophisticated. Early studies were mostly based on statistical models (Ahmed and Cook, 1979; Van Der Voort et al., 1996; Lee and Fambro, 1999; Williams and Hoel, 2003) and treated the traffic prediction problem as a linear problem. AutoRegressive Integrated Moving-Average (ARIMA) developed in (Ahmed and Cook, 1979) was one of the earliest one to solve traffic prediction problem adopting such approach. Thereafter, several variations of ARIMA including ARIMA with Kohonen maps (Van Der Voort et al., 1996), subset ARIMA (Lee and Fambro, 1999) and seasonal ARIMA (Williams and Hoel, 2003), were built to improve traffic prediction accuracy by attempting to extract more different traffic patterns (e.g., seasonal or cyclical patterns). Later, researchers argued that the traffic prediction problem should be considered as a non-linear problem (Zhang, 2003). Machine learning models with non-linear kernels or activation functions were then used to solve traffic prediction problem by first converting it to a linear problem. For example, Neural Networks (NNs)

was used for traffic prediction in (Gowrishankar and Satyanarayana, 2008) while Online Support Vector Regression (Online-SVR) was proposed for short-term traffic prediction under different traffic conditions (Castro-Neto et al., 2009). Nevertheless, these models are shallow for capturing detailed information hidden in the raw traffic data. With the recent advancement in deep learning, deep neural networks are increasingly used for predicting traffic since deeper networks can more effectively learn high dimensional features and capture hidden information in the traffic data. Huang (Huang et al., 2014) developed a deep architecture for traffic flow prediction using a Deep Belief Network (DBN) for unsupervised feature learning before making the final prediction via a multitask regression layer. On the other hand, (Jia et al., 2016) proposed a DBN for the short-term traffic speed prediction. This DBN model is trained in a greedy unsupervised fashion and fine-tuned using labelled data. Meanwhile, Recurrent Neural Network (RNN) (Mikolov et al., 2010) and its variants (e.g., Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Chung et al., 2014)) emerged in the field of natural language processing for solving problems with long-term dependency. This capability is attributed to the recycled units. These models were exploited in (Tian and Pan, 2015) and (Kang et al., 2017) for short-term traffic flow prediction while (Ma et al., 2015) and (Fu et al., 2016) proposed to use these models for traffic speed prediction.

The aforementioned models only take temporal features into consideration. However, (Lv et al., 2014) indicated that a large-scale transportation network is spatially correlated and sharing information among neighboring stations can improve prediction accuracy. Therefore, (Lv et al., 2014) proposed the Stacked AutoEncoder (SAE) model to extract spatio-temporal features from real-world traffic data. SAE uses a Softmax layer as a predictor for its final prediction. Along the same line, Convolutional Neural Network (CNN), with the ability to extract local spatial feature via its convolutional kernels, has also been used to improve prediction performance on datasets with hidden spatial information. Taking the advantage of CNN, (Ma et al., 2017) developed a CNN-based model

that learns traffic as images that encode the time and space relation of traffic speed data via a two dimensional time-space matrix and demonstrated that it can accurately predict traffic speed on two real-world large-scale road networks. Using a similar concept, (Yu et al., 2017) proposed a network grid representation framework that converts network-wide traffic speed into a series of static images for prediction. This framework was based on Deep Convolutional Neural Networks (DCNNs) and LSTM networks. DCNNs were used to extract the spatial dependency of road networks while LSTMs were utilized to learn the dynamic-temporal dependency. A similar model based on LSTM, CNN and Deep AutoEncoder (DAE) was developed in (Zheng et al., 2019) to analyze and extract temporal-spatial features for short-term traffic prediction in smart city. The Fusion Convolutional Long short-term memory Network (FCL-Net) (Ke et al., 2017), another deep learning model, enhances the contribution of spatial and temporal features towards the final prediction by stacking and fusing convolutional layers, Convolutional LSTM layers (Conv-LSTM) and multiple standard LSTM. On the other hand, (Lv et al., 2018) proposed the Look-up Convolution Recurrent Neural Network (LC-RNN) model by combining both RNN and CNN models. In (Zheng et al., 2021b), a joint temporal-spatial ensemble model combining ARIMA, LSTM, SAE and Capsule Network is proposed to analyze and extract multiple features including short-, medium- and long-term temporal features, and global- and local-spatial features.

CNN-based models consider road networks as regular grids and traffic data having regular Euclidean structure. However, road networks are inherently irregular and traffic data should instead be treated as non-Euclidean data (Ahmed et al., 2019). To overcome this limitation, the Graph Convolutional Network (GCN) (Kipf and Welling, 2017) was proposed. GCN allows convolutional operations on non-Euclidean data structure and has been advocated for traffic prediction problems. For example, (Li et al., 2018) developed the Diffusion Convolutional Recurrent Neural Network (DCRNN) operating on a directed graph, a deep learning framework based on the encoder-decoder architecture, for traffic prediction on a transportation network graph. DCRNN extracts spa-

tial dependency using bidirectional random walks on the network graph and temporal dependency using the encoder-decoder architecture. GCN is also at the heart of the design in the Spatio-Temporal Graph Convolutional Networks (STGCN) (Yu et al., 2018) which is composed of two Spatio-Temporal Convolutional blocks (namely ST-Conv blocks) and a fully-connected output layer. Each ST-Conv block contains two temporal gated convolution layers and one spatial graph convolution layer in the middle. In (Cui et al., 2019), the road network is abstracted as an undirected graph. The proposed Traffic Graph Convolutional Long Short-Term (TGC-LSTM) deep learning framework uses GCN to capture the spatial dependencies of neighboring stations of different distances while LSTM is used to extract temporal dependencies. Instead of LSTM, (Zhao et al., 2019) proposed to combine GCN with GRU to further improve the prediction accuracy. Meanwhile, (Zhang et al., 2019) addresses the problem of multi-step traffic speed prediction. The authors proposed to use the sequence-to-sequence learning architecture with attention mechanism for extracting dynamic temporal dependency by computing the different influences of each previous time step on the future time step. For capturing spatial dependency, they have suggested the use of GCN. Furthermore, (Guo et al., 2020) proposed an Optimized Graph Convolution Recurrent Neural Network (OGCRNN) model for traffic predictions considering network-wide traffic flow, speed and travel time. In this work, the proposed model learns an optimized graph in a data-driven manner during the model training phase. This is said to reveal the latent relationship among different road segments from the traffic data.

On the other hand, (Vaswani et al., 2017) introduced a novel concept of transformer which exploits the attention mechanism. Following this, (Shi et al., 2020) suggested that the convolution operator in CNN-based and GCN-based models may not adequately model the non-Euclidean pair-wise correlations of road segments, and developed a novel Attention-based Periodic-Temporal neural Network (APTN) for traffic prediction by analysing spatial, short-term, and long-term periodical dependencies using the attention mechanism. Meanwhile,

Xu *et al.* (Xu et al., 2020) developed a Spatial-Temporal Transformer Network (STTN) for long-term traffic prediction. STTN consists of spatial transformer with self-attention mechanism for modelling spatial dependencies and temporal transformer for modelling long-range temporal dependencies across previous time steps. Focusing on improving the accuracy for long-term traffic prediction, (Zheng et al., 2021a) built a Sequence-to-Sequence architecture with an embedded module based on GCN. This model uses the $k-hop$ neighbourhood matrix for local-spatial feature extraction and transformer for global-spatial feature extraction. A Graph Multi-Attention Network (GMAN) (Zheng et al., 2020) is another transformer-based model developed under the encoder-decoder architecture. Both encoder and decoder consist of multiple spatial-temporal attention blocks and they are used to model the impact of the spatial-temporal factors on traffic state. In addition, it also includes a spatial-temporal embedding to encode vertices into vectors using the *node2vec* approach introduced in (Grover and Leskovec, 2016) for vertex representation learning. Based on these latest models, we further advance the literature in this paper by advocating the need to capture dynamic spatial features for traffic prediction and design a novel SAGCN-SST for this purpose.

3. Problem Formulation

3.1. Road Network Representation

Graph theory is an effective tool to capture spatial features of data by analyzing the connectivity between detectors or road segments (Adolf, 1990). In this paper, we model the road network as an undirected graph since the traffic speed propagates the downstream while the traffic congestion propagates upstream (Long et al., 2008) and the datasets used in this paper includes traffic congestion. Considering a road network represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of nodes representing sensor locations or road segments with $|\mathcal{V}| = N$. $\mathcal{E}$ is the set of edges representing physical connectivity between sensor locations or road segments. $\mathcal{G}$ can be represented by $A \in \mathbb{R}^{N \times N}$, the $N \times N$ symmetric

adjacency matrix, with its element $A_{i,j} = 1$ if there exists a link between node i and j and 0 otherwise. The degree matrix of graph $\mathcal{G}$, $D \in \mathbb{R}^{N \times N}$ is then defined as $D_{i,i} = \sum_j A_{i,j}$, which sums the number of edges connected to each node.

The future traffic status of a node is influenced by its own current status. For this, we further define $\tilde{A} = (A + I) \in \mathbb{R}^{N \times N}$ where I is the $N \times N$ identity matrix. This accounts for the fact that each node is also self-influenced. As such, the trace $tr(\tilde{A}) = \sum_{i=1}^{N} A_{i,i} = N$. $\tilde{A}$ only describes the connectivity of neighbors one hop away from each node (i.e., $1-hop$ neighborhood). We further introduce the notion of $k-hop$ neighborhood to represent the set of nodes that are reachable within k hops from the targeted node. We define the $k-hop$ neighborhood matrix as $\tilde{A}^k \in \mathbb{R}^{N \times N}$. The reason for introducing the $k-hop$ matrix is to account for the fact that traffic congestion not only propagate to its immediate upstream and downstream road segments but also often spread in a certain area in the network (Nguyen et al., 2016).

3.2. Traffic information

Let v_t^i denotes the traffic speed measured at node i at t^{th} time step. Typically, a time step can represent 5, 15, 30, 45 and 60 mins (Bickel et al., 2007). In our work, we use 5-min time step. Given a large-scale road network, the traffic speed on N detectors is then written as $v_t = \{v_t^1, v_t^2, \ldots, v_t^i, \ldots, v_t^{N-1}, v_t^N\}; v_t \in \mathbb{R}^N, (i = 1, 2, 3, \ldots, N)$. Then $V = \{v_{t-T+1}, v_{t-T+2}, \ldots, v_{t-1}, v_t\}; V \in \mathbb{R}^{T \times N}$, $(T = 1, 2, 3, \ldots)$ gives the traffic speed collected from N detectors in the network for the past T time steps. Conversely, the traffic speed for the future is written as $V' = \{v_{t+1}, v_{t+2}, \ldots, v_{t+T'}\} \in \mathbb{R}^{T' \times N}$ where T' is the prediction horizon. Generally, traffic prediction problems can be categorized into short- ($T' < 30$ mins) and long-term ($T' \geq 30$ mins). Since we are addressing multi-step prediction problem, our solution covers both timescales whereby $T' = \{1, 3\}$ for short-term and $T' = \{6, 9, 12\}$ for long-term traffic speed prediction corresponding to $\{5, 15\}$ mins and $\{30, 45, 60\}$ mins respectively (Min and Wynter, 2011).

3.3. Problem Definition

Given past traffic speed, V and the road network $\mathcal{G}$, our aim is to predict traffic speed, V', in the future T' time steps. The problem can then be represented as in Eq. (1).

$$V' = F\left(V; \mathcal{G}\left(\mathcal{V}, \mathcal{E}, \tilde{A}^k\right)\right) \tag{1}$$

where the objective is to learn the mapping function $F(.)$ and compute the traffic speed in the next T' time steps given the traffic speed in the past T time steps and network information including the different k neighborhood matrices as input.

4. A Novel Proposed Framework

4.1. Design Overview

Figure 1 presents the overall learning architecture of our framework. It consists of three main blocks:

- **Input block** – This block is responsible for preparing the raw traffic and graph data into a trainable format as input to the spatial block.
- **Spatial block** – This block extracts both fixed and dynamic spatial features. Based on GCN, we construct $k - hop$ neighborhoods for each node in the network and utilize a self-attention mechanism to learn the degree of influence of different individual neighbor to the targeted node. The k spatial features extracted in this block is concatenated (i.e., $SAGCN = \{SAGCN^1, SAGCN^2, \ldots, SAGCN^k\}$) as input to the temporal block.
- **Temporal block** – This block then captures the temporal features. Its inputs include the k spatial features and V'. It aims to obtain the long-temporal relationship of the past and future data. Specifically, we propose

to integrate a sequence-to-sequence model within an encoder-decoder architecture (Sutskever et al., 2014) for extracting the long-temporal dependency of the traffic speed. The output of this block is the final prediction.

4.2. Input Block

The two input data are the past traffic speed measurements and the road network graph data. The traffic speed data is $V = \{v_{t-T+1}, v_{t-T+2}, \ldots, v_t\}; V \in \mathbb{R}^{T \times N \times B \times \mathcal{F}}$ where B and $\mathcal{F}$ represent the batch size and the number of considered traffic features respectively. In our work, without loss of generality, traffic feature only consists of traffic speed. Therefore, $\mathcal{F} = 1$.

The road network graph data refers to the $k - hop$ neighborhood matrices, $\{\tilde{A}^1, \tilde{A}^2, \ldots, \tilde{A}^k\}$. Since we only need the connectivity information of nodes within the neighborhood rather than the actual hop distance to the neighbors, we follow (Cui et al., 2019) and clip all elements in $\tilde{A}^k$ to be within {0,1}. We can then rewrite the $k - hop$ neighborhood matrix $\tilde{A}^k$ hereafter as follows:

$$\tilde{A}^k = Ci(\tilde{A}^k) \tag{2}$$

where $Ci(.)$ is the clip function such that $\tilde{A}^k_{i,j} = \min\left(\tilde{A}^k_{i,j}, 1\right)$. Note that when $k = 1$, $\tilde{A}^1 = \tilde{A}$ reverts back to the adjacency matrix itself describing the connectivity relationship of nodes in the $1 - hop$ neighborhood. All spatial features from $k - hop$ neighborhoods are concatenated as a matrix vector $SAGCN \in \mathbb{R}^{B \times T \times N \times k}$.

4.3. The Spatial Block

The spatial block is composed of k parallel sub-spatial blocks corresponding to k different neighborhoods (see Figure 1). Each sub-spatial block consists of m Graph Convolutional Network blocks (GCN blocks) where each GCN block comprises a Graph Convolutional Neural layer with Self-Attention mechanism (Veličković et al., 2018) (namely Self-AGCN) and a Feed Forward Neural Network (FFNN) layer. The last GCN block is followed by an additional Self-AGCN layer. At the end of parallel sub-spatial blocks, the k spatial features

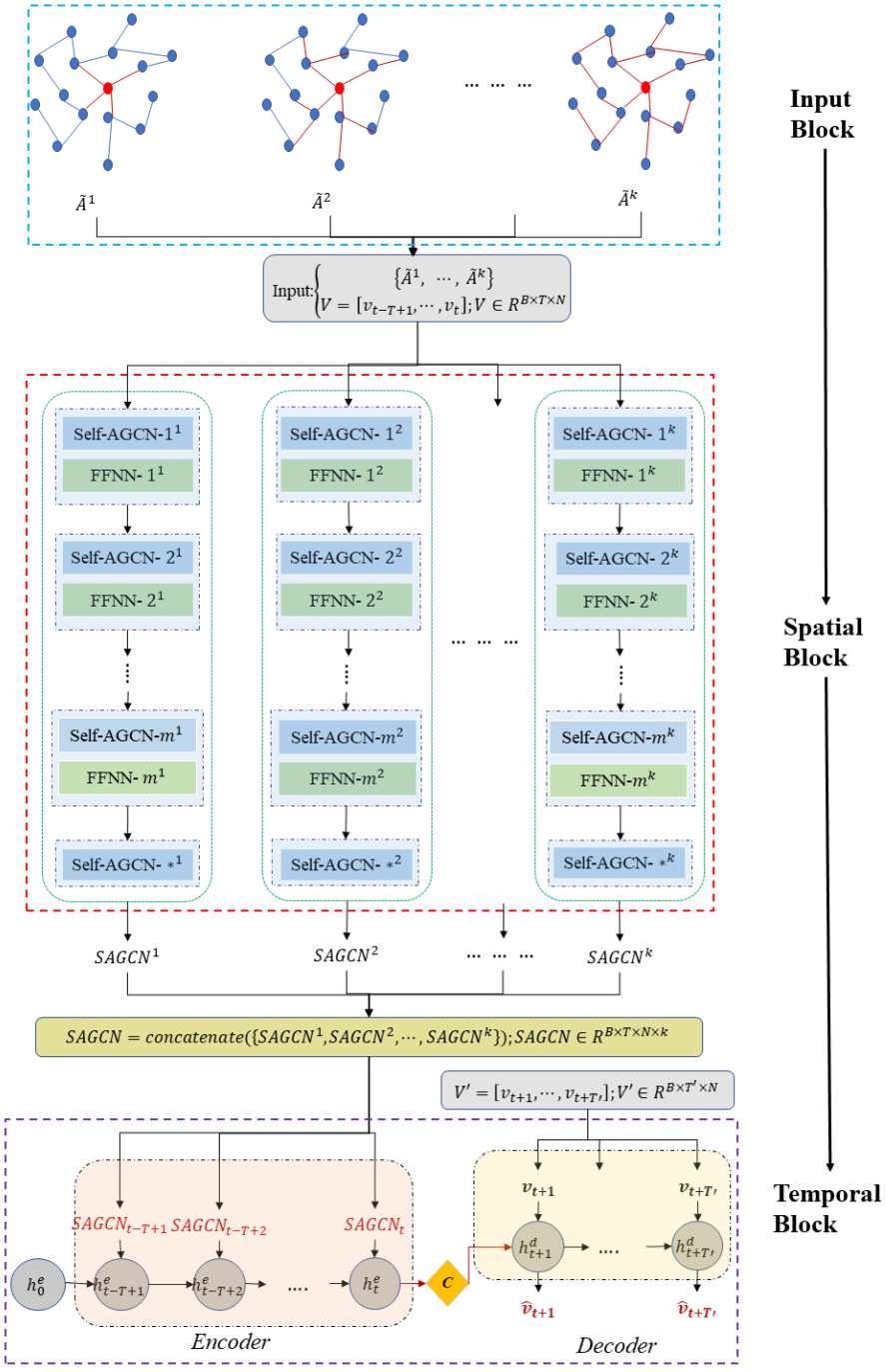


Figure 1: Our novel deep learning framework.

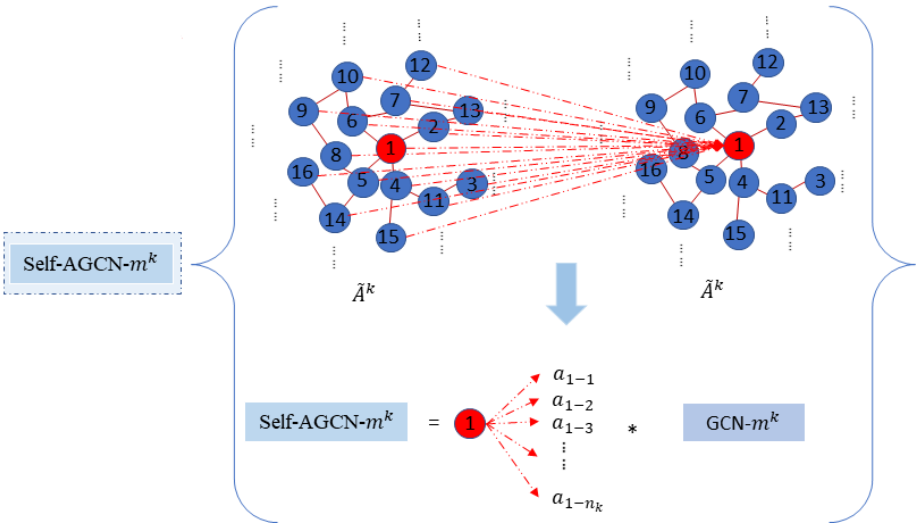


Figure 2: The working principle of the m^{th} Self-AGCN layer on the $k-hop$ neighborhood.

(i.e., $SAGCN^1$, $SAGCN^2$, ..., $SAGCN^k$) corresponding to k different neighborhoods are passed to the temporal block for extracting temporal features.

Figure 2 illustrates the working principle of the m^{th} Self-AGCN layer on the $k-hop$ neighborhood where it has n_k nodes. For example, traffic speed of node 1 in Figure 2, as the targeted node, is affected by other (n_k-1) nodes differently. As such, we need to quantify and compute the different weights of these neighbors with respect to the targeted node (i.e., $(a_{1-1}, a_{1-2}, \ldots, a_{1-n_k})$ (namely, attention weights)). To achieve this, the convolutional operation (i.e., Eq. (3)) is first conducted on the graph of the road network for fixed spatial feature extraction:

$$GC_t^{m;k} = \left(W_{gc_t^{m;k}} \circ \tilde{A}^k\right) v_t \tag{3}$$

where $\circ$ is the Hadamard product operator, $v_t \in \mathbb{R}^N$ is the traffic speed at the t^{th} time step. $W_{gc_t^{m;k}} \in \mathbb{R}^{N\times N}$ is a trainable weight matrix in the m^{th} Self-AGCN layer on the $k-hop$ neighborhood. The output matrix $GC_t^{m;k} \in \mathbb{R}^N$ represents the fixed spatial features at current time step.

As prior mentioned, each neighboring node contributes differently to the

future traffic status of a targeted node due to its distance as well as its own spatial neighborhood in relation with the targeted node. Intuitively, traffic status of a targeted node within a road network is more heavily influenced by their immediate adjacent neighbors and less affected by nodes further away. However, this is not strictly so. Moreover, traffic volume in a road network also affects the influence of neighboring nodes. For instance, neighboring nodes may not have strong influence on the targeted node in a relatively quiet road network with low traffic flow. Conversely, in a congestion-prone road network, the traffic status of neighboring nodes will have impact on the future status of the targeted node. In fact, neighboring nodes will have increasing influence as the congestion period lengthens. To extract such dynamic spatial features, we apply the self-attention mechanism at each GCN block to compute the contribution of each node in the $k-hop$ neighborhood and assign a weight to these neighboring nodes. The weight is computed based on the similarity between the neighboring node and the targeted node. Specifically, the similarity of traffic data between two nodes i and j, $u_t^{m;k}(i,j) \in \mathbb{R}^{N \times n_k}$, is computed via a $tanh$ function as follows:

$$u_t^{m;k}(i,j) = q^T tanh\Bigg(GC_t^{m;k}(i) W_f^{m;k} GC_t^{m;k}(j) \Bigg); \quad j = 1, 2, \ldots, n_k \tag{4}$$

where $W_f^{m;k} \in \mathbb{R}^{N \times N}$ is a trainable weight matrix and q^T represents the transposition or reshaping operations that are utilized to adjust the dimensions. We then compute the attention weights as probabilities (i.e., $a_t^{m;k}(i,j) \in [0.0, 1.0]$) via a $Softmax$ function given in Eq. (5).

$$\begin{aligned} a_t^{m;k}(i,j) &= Softmax\Bigg(u_t^{m;k}(i,j) \Bigg) \\ &= \frac{exp\Big(u_t^{m;k}(i,j) \Big)}{\sum_{j=1}^{n_k} exp\Big(u_t^{m;k}(i,j) \Big)} \end{aligned} \tag{5}$$

After obtaining the attention weights $a_t^{m;k}(i,j) \in \mathbb{R}^{N \times n_k}$, it is used to map

to fixed spatial traffic feature $GC_t^{m;k}(i)$ for achieving dynamic spatial traffic feature $SGC_t^{m;k} \in \mathbb{R}^N$.

$$SGC_t^{m;k} = \sum_{j=1}^{n_k} a_t^{m;k}(i,j) GC_t^{m;k}(i). \tag{6}$$

Each Self-AGCN layer is followed by a FFNN layer for improving the prediction ability on learned traffic features. This layer consists of a *tanh* layer as given in Eq. (7) and a *dropout* layer as given in Eq. (8).

$$SGC_{tanh;t}^{m;k} = tanh(SGC_t^{m;k}) \tag{7}$$

$$SGC_{drop;t}^{m;k} = dropout(SGC_{tanh;t}^{m;k}) \tag{8}$$

Compared to (Kipf and Welling, 2017) which uses a *ReLU* layer and a *dropout* layer, we propose to use a *tanh* layer in place of the *ReLU* layer. The main reason is that *ReLU* function de-activates negative values and only retains positive values. As such, it may miss some important information hidden behind negative values.

4.4. The Temporal Block

For temporal feature extraction, we propose to use the sequence-to-sequence architecture (Sutskever et al., 2014) which has already been found to offer good performance in the area of natural language processing. The architecture consists of an encoder and a decoder with a context $C \in \mathbb{R}^{B \times N}$ connecting the two (see Figure 1).

The encoder takes the $SAGCN$ produced by the spatial block as the input. It encodes the spatially-fused time series using the following:

$$h_{t-t_e}^e = \begin{cases} f_{encoder}(h_0^e, SAGCN_{t-t_e}), & t_e = T \\ f_{encoder}(h_{t-t_e-1}^e, SAGCN_{t-t_e}), & t_e \in 0, \ldots, T-1 \end{cases} \tag{9}$$

where $h_{t-t_e}^e \in \mathbb{R}^{B \times N}$ is the hidden state in the encoder at $(t-t_e)^{th}$ time step. The initial hidden state is h_0^e. The hidden state $h_{t-t_e-1}^e \in \mathbb{R}^{B \times N}$ at

$(t - t_e - 1)^{th}$ time step and the spatially-fused feature $SAGCN_{t-t_e} \in \mathbb{R}^{N \times k}$ at $(t - t_e)^{th}$ time step are used to calculate the hidden state $h^e_{t-t_e}$ at the $(t - t_e)^{th}$ time step.

The hidden state h^e_t ($t_e = 0$) at the t^{th} time step is the context vector C which encodes all information from the input $SAGCN$ in the encoder.

$$C = h^e_t \tag{10}$$

In the decoder, the context vector C as the initial hidden state $h^d_0 \in \mathbb{R}^{B \times N}$ is decoded to the target sequence. The hidden state $h^d_{t+t_d-1}$ at $(t + t_d - 1)^{th}$ time step and the target traffic speed v_{t+t_d} at $(t + t_d)^{th}$ time step are utilized to calculate the hidden state $h^d_{t+t_d}$ at the $(t + t_d)^{th}$ time step. The hidden state $h^d_{t+t_d}$ at the $(t + t_d)^{th}$ time step in the decoder is the final prediction $\tilde{v}_{t+t_d}$.

The $f_{encoder}$ and $f_{decoder}$ functions are two GRU modules (Chung et al., 2014). While GRU is based on LSTM, it incurs shorter processing time and less Central Processing Unit (CPU) cycles. The reason is that GRU combines LSTM's forget and input gates into a single "update gate", and also merges the memory cell and hidden state. This makes GRU simpler than the standard LSTM but still efficient. Figure 3 depicts the working process and data flow in a recycled unit of the GRU module. GRU consists of three parts: the update gate z_t, the reset gate r_t and the hidden state h^e_t. The update gate, z_t, extracts the long-term dependency of the data. It decides how much information it needs to update from the input $SAGCN_t$ and the hidden state at the previous time step h^e_{t-1} (see Eq. (11)).

$$z_t = \sigma(W_z \times SAGCN_t + U_z \times h^e_{t-1} + b_z) \tag{11}$$

The reset gate, r_t, captures the short-term dependency of traffic features. It decides how much information from the hidden state at the previous time step is retained for updating the current hidden state. It is computed in a similar

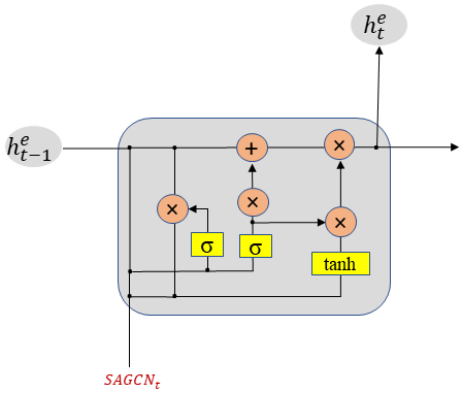


Figure 3: Gated Recurrent Unit

manner as the update gate by Eq. (12).

$$r_t = \sigma(W_r \times SAGCN_t + U_r \times h_{t-1}^e + b_r) \tag{12}$$

Then, the input $SAGCN_t$, the reset gate r_t and the hidden state at the previous time step h_{t-1}^e are used to activate the candidate hidden state $\tilde{h}_t^e$ via Eq. (13).

$$\tilde{h}_t^e = tanh(W_h \times SAGCN_t + U_h \times (r_t \circ h_{t-1}^e) + b_h) \tag{13}$$

where W_z, W_r and W_h are the weights of the update gate, the reset gate and the candidate hidden state respectively while b_z, b_r and b_h are the corresponding bias for each gate and state. Furthermore, U_z, U_r and U_h are the weights of the hidden state at the previous time step h_{t-1}^e in the update gate, the reset gate and the candidate hidden state, respectively. Finally, the current hidden state can be calculated using the update gate z_t, the hidden state at the previous time step h_{t-1}^e and the current candidate hidden state $\tilde{h}_t^e$ using Eq. (14).

$$h_t^e = (1 - z_t) \circ h_{t-1}^e + z_t \circ \tilde{h}_t^e \tag{14}$$

4.5. Loss Function

To train our SAGCN-SST model, the RMSprop optimizer (Tieleman and Hinton, 2012) is used to minimize the error between the real and predicted traffic. It restricts the oscillations in the vertical direction. The learning rate can be adjusted to take larger steps in the horizontal direction for faster convergence compared to Gradient Descent Optimizer (Bottou, 2010).

Mean Square Error (MSE) in Eq. (15) as loss function is adopted to train our model as it learns faster than Mean Absolute Error (MAE).

$$Loss = L(v_t, \tilde{v}_t) = \frac{1}{N} \sum_{i=1}^{N} \left(v_t^i - \tilde{v}_t^i\right)^2 \tag{15}$$

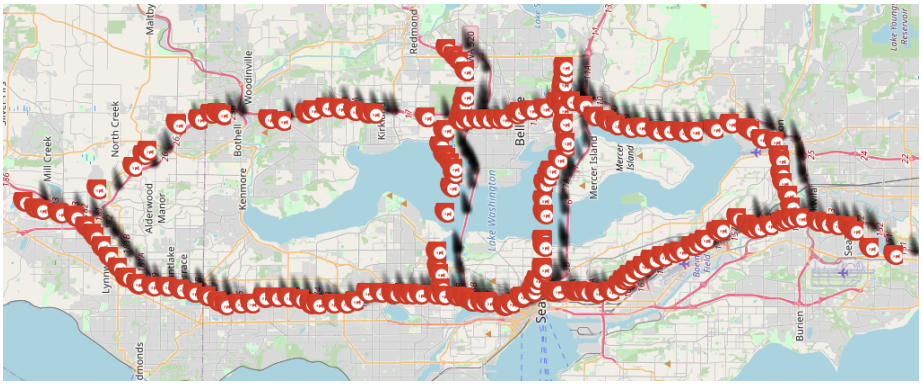
where $L(.)$ is the MSE loss function. It calculates the residual error between the real traffic data v_t^i and the predicted traffic data $\tilde{v}_t^i$.

5. Performance Evaluation

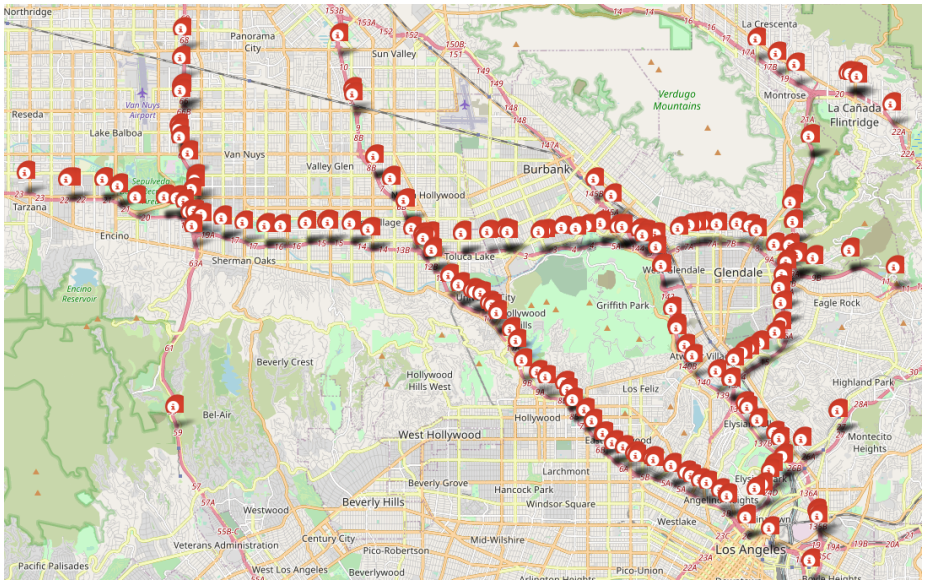
5.1. Data Description

To train and test our proposed framework, two real-world datasets from large-scale road networks are utilized: hereafter labelled as `Loop-Seattle` (Cui et al., 2019) and `METR-LA` (Li et al., 2018). `Loop-Seattle` is collected from inductive loop detectors deployed on four connected freeways (I-5, I-405, I-90 and SR-520) in the Greater Seattle Area, and the locations of the loop detectors are indicated by the red pins in Figure 4 (a). This dataset records traffic information from 323 detectors over the entirety of year 2015 at 5-min time step. We use unweighted undirected graph to represent this network. The second real-world dataset, `METR-LA`, is collected from loop detectors in the highways of Los Angeles County (Jagadish et al., 2014). Similarly, the locations of detectors in this network are shown via the red pins in Figure 4 (b). It includes 207 detectors and covers 4 months of traffic speed data from the 1^{st} of March to the 30^{th} of June in 2012. In this dataset, an undirected graph with edge weights is used to construct the adjacency matrix. The pairwise road

distances between detectors are first computed and then a thresholded Gaussian Kernel (Shuman et al., 2013) is used to build the adjacency matrix. The edge weights are calculated by Eq. (16) below:



(a)



(b)

Figure 4: Locations of loop detectors in (a) `Loop-Seattle` and (b) `METR-LA` datasets.

$$W_{i,j} = \begin{cases} exp(-\frac{dist(i,j)^2}{2\sigma^2}), & if\ dist(i,j) < d_{threshold} \\ 0, & otherwise \end{cases} \tag{16}$$

where $W_{i,j}$ is the edge weight between node i and node j, and $dist(i,j)$ represents the actual physical road distance between node i and node j in the road network. The standard deviation of the distances is denoted by σ and $d_{threshold}$ is the threshold. Table 1 provides the basic statistics of both datasets including maximum (Max), minimum (Min), mean value (Mean), standard deviation (Std) and variance (Var) of traffic speed data as well as the dataset size (in MByte). From the table, we note that `METR-LA` has higher traffic fluctuations with larger standard deviation and variance than `Loop-Seattle`.

Table 1: Characteristics of both traffic speed (miles/hour) datasets

Dataset	Max	Min	Mean	Std	Var	Size
`Loop-Seattle`	158.19	0.74	56.57	11.43	147.25	274.40 MB
`METR-LA`	70.00	0.00	53.72	19.19	374.85	57.00 MB

5.2. Evaluation Metrics

To evaluate our model, we follow the evaluation metrics used in (Huang et al., 2014; Lv et al., 2015) and define the accuracy of the multi-step traffic prediction as $(100\% - MAPE)$ where Mean Absolute Percentage Error (MAPE) is given as

$$MAPE = \frac{1}{N \times T'} \cdot \sum_{i=1}^{N} \sum_{t=1}^{T'} \frac{|v_t^i - \tilde{v}_t^i|}{v_t^i} \times 100\%. \tag{17}$$

MAPE is the absolute difference between the real and predicted traffic data and is utilized to measure the prediction error.

Furthermore, we complement this with two other conventional performance metrics commonly used in the literature (Tan et al., 2009; Lv et al., 2015) namely Mean Absolute Error (MAE) and Root-Mean Square Error (RMSE) which are computed as follows:

$$MAE = \frac{1}{N \times T'} \cdot \sum_{i=1}^{N} \sum_{t=1}^{T'} |v_t^i - \tilde{v}_t^i| \tag{18}$$

$$RMSE = \left[\frac{1}{N \times T'} \cdot \sum_{i=1}^{N} \sum_{t=1}^{T'} (v_t^i - \tilde{v}_t^i)^2 \right]^{\frac{1}{2}}. \tag{19}$$

MAE presents the average absolute difference between the real and predicted traffic data. It is used to measure absolute prediction error. RMSE is the standard deviation of the residuals, which is the difference between the real and predicted traffic data.

5.3. Parameter settings

In our SAGCN-SST model, there are several parameters that need to be set for achieving accurate predictions. The parameters mainly relate to the training process. Specifically, the parameters are the learning rate r, batch size B, observed time steps T, targeted time steps T' and the number of epochs. We follow (Zang et al., 2018; Cui et al., 2019) and set the learning rate r, batch size B and observed time steps T as 10^{-3}, 40 and 10 respectively. As prior mentioned, since we are addressing multi-step traffic prediction problem, we set the targeted time steps T' as 1, 3, 6, 9 and 12, corresponding to 5, 15, 30, 45 and 60 mins as prediction horizons respectively (Li et al., 2018; Yu et al., 2018). To find the number of epochs, we use *stop early* strategy in which the training process will be stopped when the training loss continues to decrease in 10 epochs while the validation loss increases. This avoids the problem of over-fitting.

Finally, we follow the convention and use 70% of the data for training, 20% for validation and 10% for testing. All experiments are conducted on a GeForce GTX 1080 Ti GPU with 11 GB physical memory, and Pytorch is used to program this work.

5.4. Performance Evaluation of SAGCN-SST

To evaluate our proposed SAGCN-SST, we first conduct experiments with different number of GCN blocks (within the $m = [1..6]$ interval) for $T' = 1$ (i.e., prediction for 5 mins in advance) for the $1 - hop$ neighborhood (i.e., $k = 1$).

Table 2 presents the prediction results achieved by our SAGCN-SST model for both `Loop-Seattle` and `METRA-LA` datasets.

Table 2: Results of SAGCN-SST with $m = [1..6]$ GCN blocks for `Loop-Seattle` and `METR-LA`

	`Loop-Seattle` ($T' = 1$)			`METR-LA` ($T' = 1$)		
Layers	MAE	MAPE	RMSE	MAE	MAPE	RMSE
$m = 1$	0.8435	1.98	1.1175	0.9858	2.05	1.4051
$m = 2$	0.8186	1.91	1.0851	0.9672	2.01	1.3638
$m = 3$	0.8031	1.89	1.0686	0.9134	1.90	1.2894
$m = 4$	0.8181	1.92	1.0872	0.9499	1.98	1.3471
$m = 5$	0.8219	1.96	1.2140	0.9353	1.93	1.3209
$m = 6$	0.8102	1.89	1.0759	0.9340	1.93	1.3144

For `Loop-Seattle`, our SAGCN-SST model achieves MAPE below 2% for m between 1 and 6. The lowest MAPE is achieved (i.e., 1.89%) when SAGCN-SST has 3 and 6 GCN blocks. In these two cases (i.e., $m = 3$ and $m = 6$), the errors in MAE and RMSE for $m = 3$ are lower. Furthermore, considering that having a deeper model with more GCN blocks costs more in terms of physical memory and GPU cycles, incurs longer running time, and may even result in over-fitting, we set $m = 3$ in the ensuing experiments for `Loop-Seattle`. For `METR-LA`, the best result (i.e., MAPE = 1.90%) is achieved when m is also set as 3. In addition, the average MAE and RMSE for both datasets are respectively below 1.00 and 1.50. Compared to results achieved from `Loop-Seattle`, these three types of errors obtained from `METR-LA` are larger. This indicates that traffic pattern in `METR-LA` is more complex.

Next, we proceed with experiments for multi-step traffic speed prediction with prediction horizon, $T' = \{1, 3, 6, 9, 12\}$ with different neighborhoods (i.e., $k = [1..5]$) on both datasets. The results are summarized in Table 3. For `Loop-Seattle`, our SAGCN-SST consistently achieves MAPE less than 2% for the different $k-hop$ neighborhoods across all prediction horizons. This indicates that longer prediction horizons (i.e., $T' = \{6, 9, 12\}$) does not affect SAGCN-

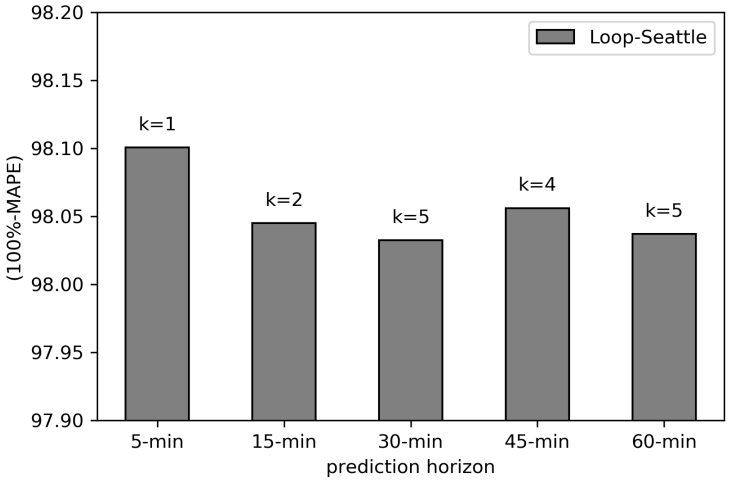
SST's prediction performance. This is mainly due to our design which integrated the sequence-to-sequence architecture in our temporal block that effectively captures long-temporal dependency of the data. Figure 5 (a) presents the best prediction results achieved on `Loop-Seattle` for different neighborhoods (i.e., $k = \{1, 2, 3, 4, 5\}$) over different prediction horizons ($T' = \{1, 3, 6, 9, 12\}$). It is clear that short-term predictions mainly depend on adjacent neighbors while long-term predictions need to take into account the influence on a wider neighborhood. Specifically, for $T' = 1$ and 3, the best results are obtained when k is equal to 1 and 2, respectively. In contrast, for $T' = \{6, 9, 12\}$, the best performances are achieved when k is higher (i.e., when $k = 5$, 4 and 5, respectively).

The observations are quite different for the `METR-LA` dataset. The best predictions are always obtained when $k = 1$ (see Figure 5(b)). This is due to the different nature of the traffic patterns in the two datasets. Specifically, we observe that congestion duration in `Loop-Seattle` tends to be significantly longer than that recorded in `METR-LA`. We use Figure 6 to exemplify this. The figure shows the real (black solid line) and predicted (red dashed line) traffic speed in a day with 288 time steps from two randomly selected detectors retrieved from `Loop-Seattle` (left column) and `METR-LA` (right column), respectively. The mean and variance of real traffic speed from this day are 50.92 and 203.19 on `Loop-Seattle` and 65.87 and 8.44 on `METR-LA`, respectively. Its predicted mean and variance are 50.60 and 210.16 on `Loop-Seattle` and 66.34 and 2.80 on `METR-LA`, respectively. The x-axis represents the time step and the y-axis is traffic speed. The coordinates indicated by the arrows show the start and end of a congestion incident. From the figure, we see that the traffic congestion duration is 120 mins ($= (150 - 126) \times 5$) on `METR-LA` while for `Loop-Seattle`, the congestion lasts for 330 mins ($= (181 - 115) \times 5$). From these observations from the two datasets, we can see that higher k (i.e., considering wider neighborhood) offers better prediction accuracy for traffic congestion that tends to last longer (i.e., long-term prediction) and vice versa.

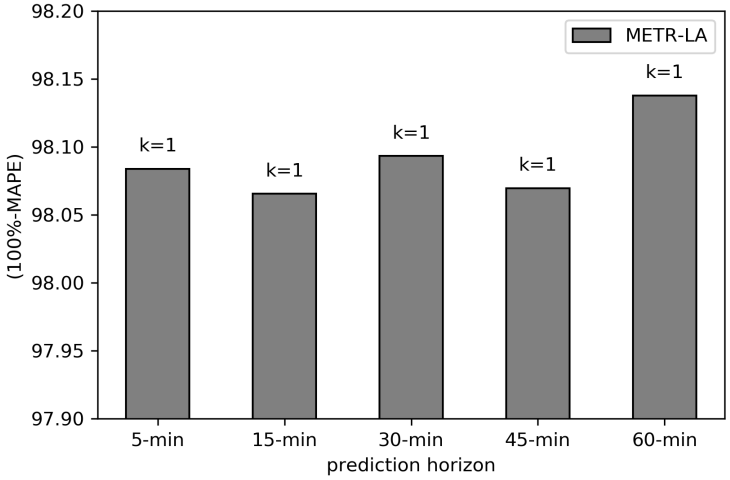
In Figure 6, prediction horizon T' increases in the range $\{1, 3, 6, 9, 12\}$ from top to bottom. From the figure, it is clear that our model is able to accurately

Table 3: Results of SAGCN-SST on the $k-hop$ neighbourhoods for both datasets

Model	Loop-Seattle			METR-LA		
Name	MAE	MAPE	RMSE	MAE	MAPE	RMSE
(a) 5-min future prediction (T'=1)						
k=1	0.8156	1.90	1.0802	0.9256	1.92	1.3090
k=2	1.0562	2.55	1.4301	1.0494	2.18	1.5103
k=3	0.8320	1.96	1.1041	1.0091	2.11	1.4417
k=4	0.8021	1.90	1.0735	1.0706	2.23	1.5290
k=5	0.8277	1.94	1.1012	1.0375	2.14	1.4714
(b) 15-min future prediction (T'=3)						
k=1	0.8627	2.03	1.1528	0.9392	1.93	1.3260
k=2	0.8280	1.96	1.0982	1.0085	2.12	1.4313
k=3	0.8702	2.08	1.1692	1.0619	2.21	1.5153
k=4	0.8453	2.02	1.1300	1.0295	2.12	1.4467
k=5	0.8643	2.05	1.1540	1.0468	2.16	1.4894
(c) 30-min future prediction (T'=6)						
k=1	0.8500	2.02	1.1310	0.9242	1.91	1.3049
k=2	0.8767	2.08	1.1585	1.0471	2.19	1.5045
k=3	0.8745	2.07	1.1618	1.0023	2.08	1.4239
k=4	0.8651	2.04	1.1469	1.0488	2.17	1.4849
k=5	0.8332	1.97	1.1147	1.0577	2.21	1.5047
(d) 45-min future prediction (T'=9)						
k=1	0.8358	1.99	1.1197	0.9198	1.93	1.2976
k=2	0.8432	2.01	1.1287	1.0640	2.20	1.5159
k=3	0.8507	2.01	1.1332	1.0569	2.20	1.5159
k=4	0.8221	1.94	1.0921	1.0256	2.12	1.4603
k=5	0.8344	1.96	1.1082	1.0671	2.25	1.5244
(e) 60-min future prediction (T'=12)						
k=1	0.8613	2.03	1.1459	0.9033	1.86	1.2701
k=2	0.8577	2.03	1.1422	1.0399	2.19	1.4726
k=3	0.8382	2.00	1.1129	1.0510	2.19	1.4896
k=4	0.8271	1.97	1.1065	1.0671	2.25	1.5182
k=5	0.8266	1.96	1.0985	0.9890	2.04	1.3956



(a)



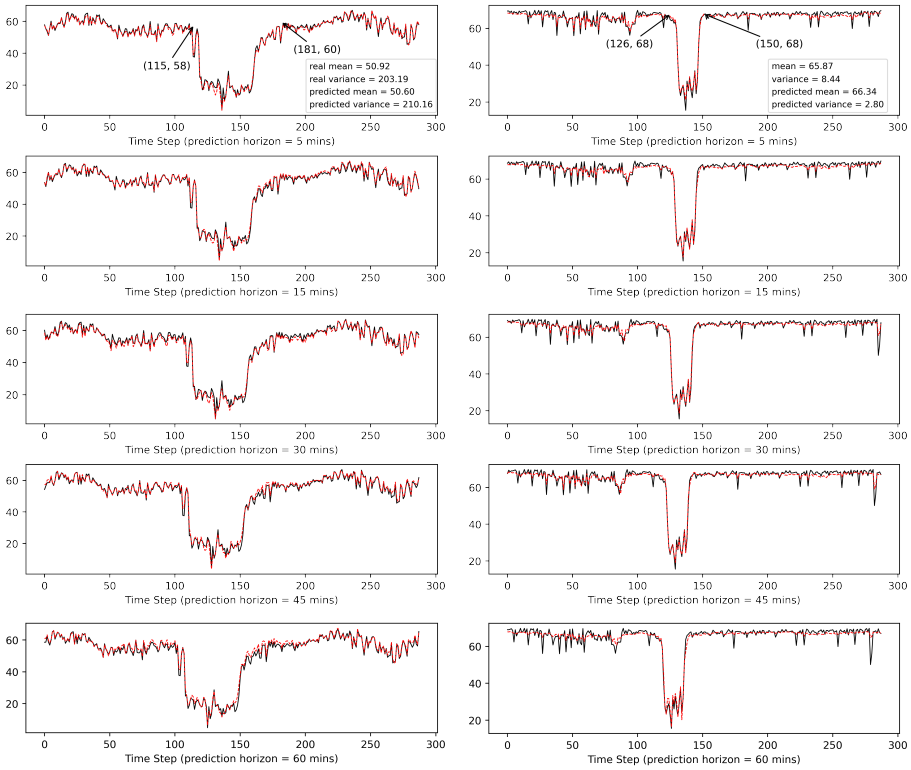
(b)

Figure 5: Relationship of the $k-hop$ neighbourhoods and the prediction horizon with accuracy from SAGCN-SST model on `Loop-Seattle` (a) and `METR-LA` (b). The x-axis is the prediction horizon and the y-axis is the prediction accuracy.

predict traffic speed across the entire duration including during peak hours (traffic speed is lower) and off-peak hours (traffic speed is higher). Specifically for more challenging tasks $T' = \{6, 9, 12\}$ compared to $T' = \{1, 3\}$, SAGCN-SST can not only accurately follow the overall trends but also capture the details of rapid fluctuations.

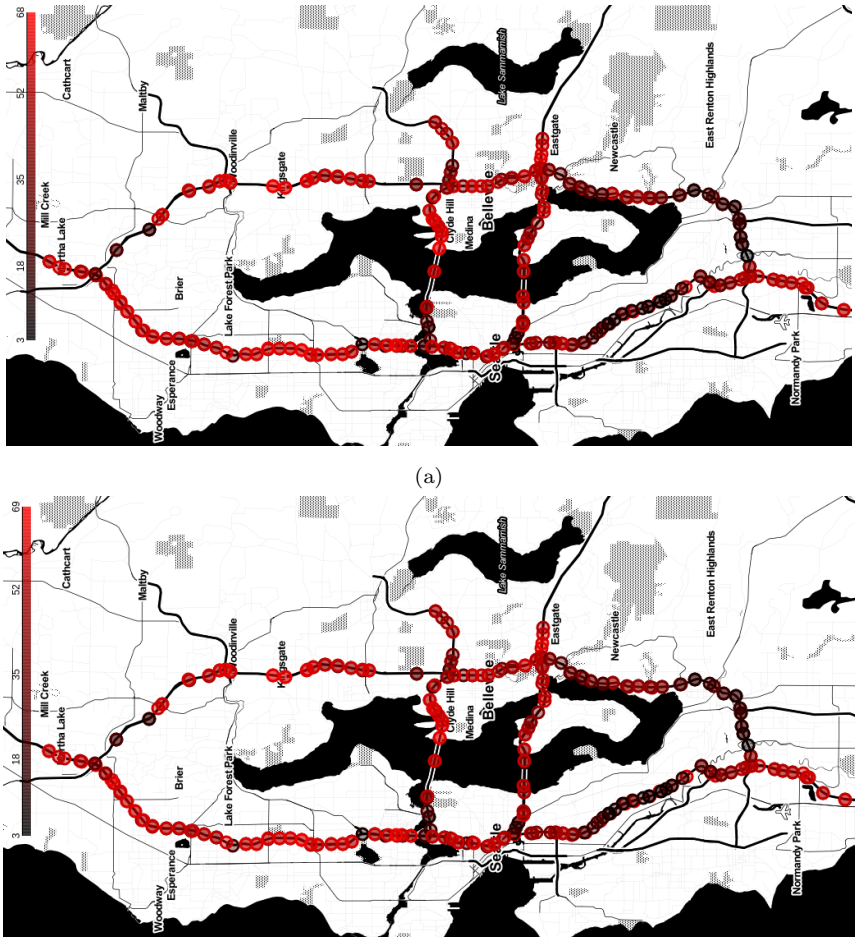
Figure 7 and Figure 8 visualize the real (sub-figure (a)) and predicted (sub-figure (b)) traffic speed at a randomly selected time step from our SAGCN-SST model on the road network of `Loop-Seattle` and `METR-LA`, respectively. From the figures, it can be observed that the real traffic on the road network are closely predicted across the entire map and thus, further validating the capability of our SAGCN-SST model in computing accurate predictions on large-scale road networks. Furthermore, both Figure 7 and Figure 8 also show that very low or high traffic speed are recorded on several continuous detectors for both road networks. It indicates that traffic state recorded by a detector is influenced by its neighbours. This information should be considered in traffic prediction models. Our SAGCN-SST, that defines $k - hop$ neighborhoods for each detector and analyzes spatial features from its neighborhood, captures exactly this information to achieve accurate predictions.

Let the residuals of traffic speed predictions be defined as $(v_t^i - \tilde{v}_t^i)$. Considering that the residual as an indicator on whether the results of a model are statistically correct, we show in Figure 9 the residuals of traffic speed predictions by our proposed model on both datasets. From top to bottom, the prediction horizons are 5, 30 and 60 mins, respectively. The x-axis represents the residuals (i.e., $v_t^i - \tilde{v}_t^i$) and the y-axis represents the probability density of the residuals. For both datasets with 5 mins as prediction horizon (top row of the figure), we see that the residual distributions follow normal distributions with zero means. For longer prediction horizons (i.e., 30 and 60 mins), while the residual distributions still resemble that of a normal distribution, the means shift away from zero. This is because longer prediction horizons are less impacted by historical traffic data compared to short prediction horizons. The residual's normal distributions in Figure 9 again suggests that our proposed model is capable of capturing dynamic spatial-temporal features and provide more accurate predictions.



`Loop-Seattle` (left) *vs* `METR-LA` (right)

Figure 6: (Color Online) Real and predicted traffic speed ($miles/hour$) in a day with 288 ($= \frac{24h*60mins}{5mins}$) time steps from SAGCN-SST on `Loop-Seattle` (left) and `METR-LA` (right) with a time step = 5 mins.



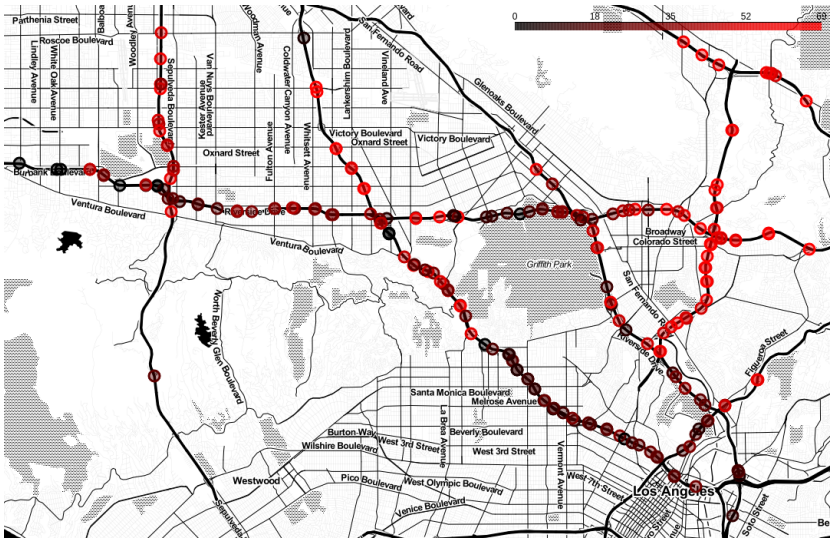
(a)

(b)

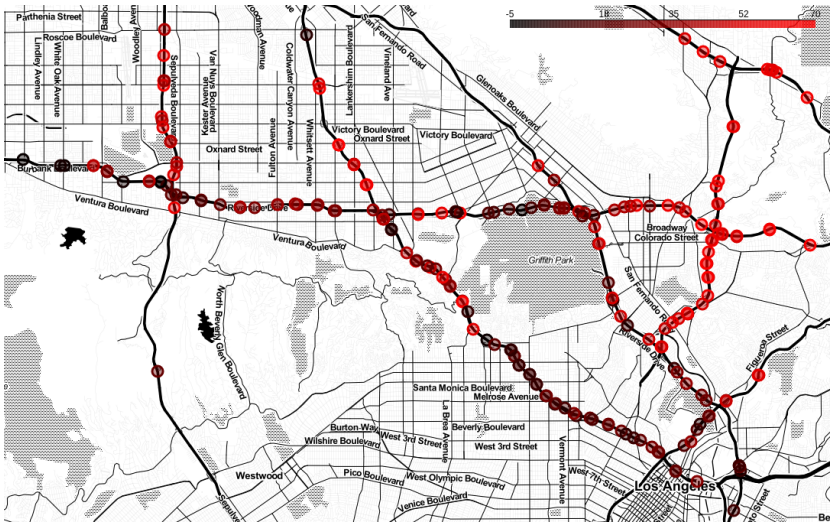
Figure 7: (Color Online) Visualization of real (a) and predicted (b) traffic speed at a randomly selected time step on the road network of `Loop-Seattle`. Darker color represent lower traffic speed.

5.5. Comparison Study

We compare our proposed model, SAGCN-SST, to well-known existing models in recent literature. The chosen representative models from the state-of-the-art adopt one of four different approach to treat traffic prediction problem. These four approaches respectively consider traffic prediction as 1) a temporal, 2) a spatial, 3) a spatial-temporal and 4) a fixed spatial dynamic-temporal process. The seven models that are used for our comparison study based on the



(a)

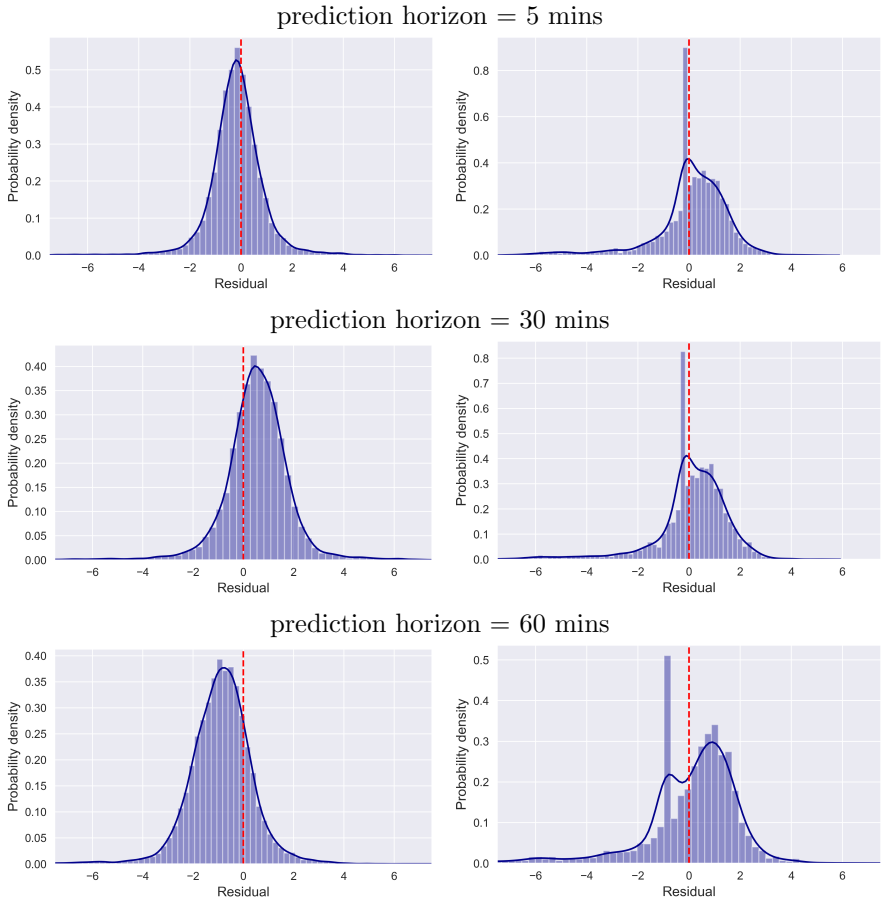


(b)

Figure 8: (Color Online) Visualization of real (a) and predicted (b) traffic speed at a randomly selected time step on the road network of `METR-LA`. Darker color represent lower traffic speed.

abovementioned approaches are the following:

1) Gate Recurrent Unit (GRU) (Chung et al., 2014): See Section 4.4 for details.



`Loop-Seattle` (left) VS `METR-LA` (right)

Figure 9: The prediction residuals of our proposed model on two datasets: `Loop-Seattle` (left column) and `METR-LA` (right column).

2) Graph Convolutional Network (GCN) reported in (Zhang et al., 2019) consists of two graph convolutional layers, and each layer is followed by a *ReLU* and a *dropout* layer. Another work following this approach is Graph Convolutional Network with Self-Attention mechanism (namely SAGCN) which is treated as a sub-spatial block in our SAGCN-SST model.

3) CNN-LSTM (also known as SRCNs in (Yu et al., 2017)) consists of Deep Convolutional Neural Networks (DCNNs) and LSTMs. DCNNs are used to capture the spatial dependency of network-wide traffic and LSTMs are utilized to learn the temporal dynamics. Temporal Graph Convolutional Network (T-GCN) (Zhao et al., 2019) combines GCN and GRU. GCN captures the spatial dependency and GRU focuses on learning the temporal dependency. Traffic Graph Convolutional Long Short-Term (TGC-LSTM) model (Cui et al., 2019) consists of GCN and LSTM corresponding to capture spatial and temporal features, respectively. An L1-norm on the graph convolution weights and an L2-norm on the graph convolution features are added to the loss function for enhancing the interpretability of the model.

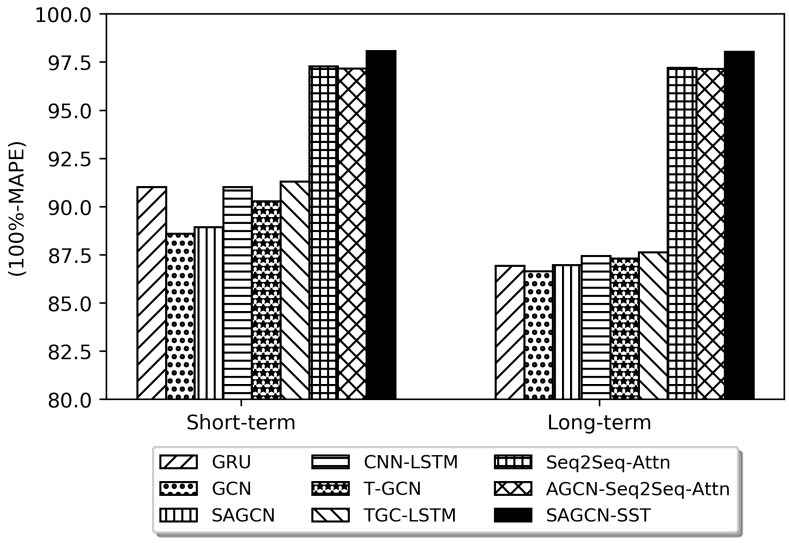
4) AGC-Seq2Seq-Attn model in (Zhang et al., 2019) includes two parts: the Graph Convolutional network and the Sequence-to-Sequence architecture consisting of an encoder and a decoder with the Attention mechanism. Two GRUs are used to build the encoder and the decoder. The graph convolution operation is firstly utilized to capture the spatial characteristics based on the topology of the underlying road network, and then its output is treated as the input of the encoder that encodes the spatially-fused time series to a context vector. After that, the context vector is decoded to the target multi-step outputs in the decoder with the attention mechanism. Another work following this approach is the Seq2Seq-Attn model reported in (Zhang et al., 2019), and the main difference between Seq2Seq-Attn and AGCN-Seq2Seq-Attn (Zhang et al., 2019) is the graph convolution layer.

Table 4: Comparison of all models for both datasets

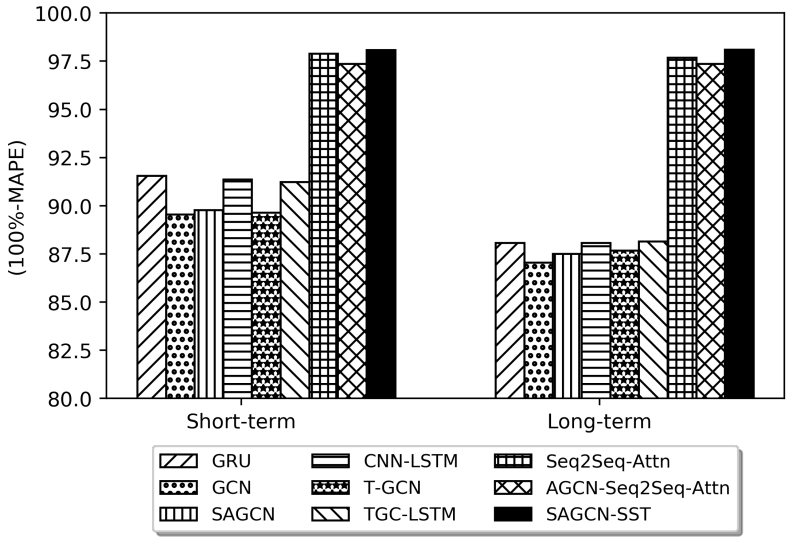
Model Name	Loop-Seattle			METR-LA		
	MAE	MAPE	RMSE	MAE	MAPE	RMSE
(a) 5-min future prediction (T'=1)						
GRU	3.0796	8.10	4.5349	3.3181	7.87	5.2300
GCN	3.7696	11.00	5.8426	4.1113	10.00	6.4648
SAGCN	3.6297	10.38	5.4396	4.1258	9.86	6.3436
CNN-LSTM	3.0753	8.19	4.5690	3.2930	7.88	5.3107
T-GCN	3.3568	9.07	4.9371	3.8915	9.77	6.2140
TGC-LSTM	3.0007	7.90	4.4650	3.5857	8.31	5.5387
Seq2Seq-Attn	1.1724	2.75	1.5625	1.0344	2.10	1.4552
AGCN-Seq2Seq-Attn	1.2167	2.86	1.6378	1.2690	2.63	1.8230
SAGCN-SST	0.8156	1.90	1.0802	0.9256	1.92	1.3090
(b) 15-min future prediction (T'=3)						
GRU	3.5166	9.86	5.3336	3.8002	9.28	5.9989
GCN	3.9293	11.82	6.1877	4.4120	10.92	6.8530
SAGCN	3.8825	11.74	5.9793	4.3333	10.59	6.7672
CNN-LSTM	3.4752	9.77	5.3258	3.8441	9.42	6.1117
T-GCN	3.6482	10.36	5.5533	4.3111	10.97	6.7720
TGC-LSTM	3.4051	9.51	5.2335	3.7832	9.25	6.0438
Seq2Seq-Attn	1.1577	2.69	1.5352	1.0419	2.13	1.4614
AGCN-Seq2Seq-Attn	1.2134	2.82	1.6206	1.2925	2.67	1.8408
SAGCN-SST	0.8280	1.96	1.0982	0.9392	1.93	1.3260
(c) 30-min future prediction (T'=6)						
GRU	3.9625	11.73	6.0602	4.4496	10.92	6.8299
GCN	4.1249	12.62	6.5210	4.9551	12.24	7.4193
SAGCN	4.1371	12.63	6.4485	4.6797	11.63	7.1876
CNN-LSTM	3.8874	11.55	6.0237	4.3388	10.90	6.8113
T-GCN	3.9805	11.54	6.1095	4.6214	11.44	7.1553
TGC-LSTM	3.8570	11.17	5.9748	4.4771	11.03	6.9709

Seq2Seq-Attn	1.2281	2.87	1.6327	0.9741	1.97	1.5757
AGCN-Seq2Seq-Attn	1.2438	2.89	1.6425	1.2431	2.57	1.7826
SAGCN-SST	0.8332	1.97	1.1147	0.9242	1.91	1.3049
(d) 45-min future prediction (T'=9)						
GRU	4.2900	13.04	6.5693	4.9017	12.03	7.4453
GCN	4.2786	13.00	6.7792	5.2008	12.94	7.7243
SAGCN	4.2671	12.80	6.6604	5.1772	12.75	7.7956
CNN-LSTM	4.0928	12.73	6.4350	4.7360	12.25	7.3165
T-GCN	4.2564	12.67	6.5664	4.9924	12.53	7.5092
TGC-LSTM	4.0914	12.51	6.4900	4.6058	11.84	7.2901
Seq2Seq-Attn	1.2129	2.83	1.6082	1.5080	3.19	2.1990
AGCN-Seq2Seq-Attn	1.2002	2.82	1.6132	1.2759	2.61	1.8303
SAGCN-SST	0.8221	1.94	1.0921	0.9198	1.93	1.2976
(e) 60-min future prediction (T'=12)						
GRU	4.5500	14.42	6.9905	5.1421	12.86	7.6679
GCN	4.4920	14.44	7.1989	5.4567	13.69	8.1325
SAGCN	4.4873	13.67	7.0132	5.2514	13.11	8.0610
CNN-LSTM	4.3054	13.38	6.7549	5.0391	12.67	7.7585
T-GCN	4.5414	13.90	7.0176	5.1833	13.02	7.7762
TGC-LSTM	4.3459	13.44	6.9132	5.0078	12.73	7.7247
Seq2Seq-Attn	1.1607	2.70	1.5469	0.9079	1.82	1.2839
AGCN-Seq2Seq-Attn	1.2172	2.85	1.6270	1.3181	2.73	1.8674
SAGCN-SST	0.8266	1.96	1.0985	0.9033	1.86	1.2701

Table 4 presents achieved results of all chosen models for $T' = \{1, 3, 6, 9, 12\}$ on both datasets while Figure 10 shows the corresponding accuracy for $T' = \{1, 3\}$ and $T' = \{6, 9, 12\}$. From Table 4 and Figure 10, the following attributes can be observed with regards to the different approaches to resolve the traffic prediction problem.



(a)



(b)

Figure 10: Comparison of prediction accuracy (100%-MAPE) on short- and long-term prediction tasks for all models on `Loop-Seattle` (a) and `METR-LA` (b).

- The three types of error achieved by GRU increase when the prediction horizon is longer, as shown in Table 4. For example, from $T' = 1$ to $T' = 12$, the MAE, MAPE and RMSE of GRU on `Loop-Seattle` increase

from 3.0796, 8.10% and 4.5349 to 4.5500, 14.42% and 6.9905 respectively. The same applies to the `METR-LA` dataset. This indicates that GRU, which mainly works for the temporal feature extraction, can offer higher prediction accuracy for short-term prediction while the performance deteriorates when the prediction horizon is longer.

- From Table 4, GCN and SAGCN are the two worst performing models. This is due to the fact that GCN and SAGCN are generally used to capture spatial features. They are unable to capture temporal features. Compared to GRU that presents a large increase of MAE, MAPE and RMSE from $T' = 1$ to $T' = 12$ on both datasets, GCN and SAGCN models only present a small increase. For example, on `Loop-Seattle`, the MAPEs of GCN and SAGCN increase by 3.44% and 2.29%, respectively, while the MAPE of GRU grows by 6.32%. The reason is that the spatial feature starts to play an increasingly more important role when the prediction horizon is longer. In addition, on both datasets, SAGCN performs slightly better than GCN for the same prediction horizon. This is because the self-attention mechanism in SAGCN is able to derive the different contributions of neighboring nodes via distribution of different weights to each of them.

- Spatial-temporal models (including CNN-LSTM, T-GCN and TGC-LSTM) achieve better performances on both datasets compared to spatial models (i.e., GCN and SAGCN). For example, for $T' = 1$, the average MAE, MAPE and RMSE of spatial-temporal models on `Loop-Seattle` are 3.1443, 8.39% and 4.6570, respectively, while the average MAE, MAPE and RMSE achieved by spatial models are 3.6997, 10.69% and 5.6411, respectively. This is due to the fact that CNN-LSTM, T-GCN and TGC-LSTM are able to capture both spatial and temporal features for their final prediction while GCN and SAGCN only rely on extracting spatial features. This phenomenon is more obvious for short-term prediction. For instance, for $T' = 1$, the average MAE, MAPE and RMSE of CNN-LSTM, T-GCN and TGC-LSTM decrease by 0.5554, 2.30% and 0.9841, respectively, com-

pared to the average MAE (3.6997), MAPE (10.69%) and RMSE (5.6411) of GCN and SAGCN. For $T' = 12$, the three types of error only decrease by 0.0921, 0.48% and 0.2109, respectively, compared to GCN and SAGCN with 4.4897, 14.06% and 7.1061 of these errors. This is because the spatial feature starts to play an increasingly more important role when the prediction horizon is longer. Similar observations can also be found in `METR-LA`.

- Seq2Seq-Attn and AGCN-Seq2Seq-Attn perform better than CNN-LSTM, T-GCN and TGC-LSTM. Their MAE, MAPE and RMSE over all different prediction horizons on two datasets are less than 1.5080, 3.19% and 2.1990, respectively. The accuracy for short-term prediction by Seq2Seq-Attn and AGCN-Seq2Seq-Attn are similar for long-term prediction. Between Seq2Seq-Attn and AGCN-Seq2Seq-Attn, the results achieved by Seq2Seq-Attn are slightly better. This observation does not agree with the results reported in (Zhang et al., 2019) where the reversed is observed. The main reason for this phenomenon is that AGCN-Seq2Seq-Attn is a more complex model that requires large datasets and higher number of features for achieving better results. In (Zhang et al., 2019), AGCN-Seq2Seq-Attn is tested utilizing several features including maximum, minimum and median of traffic speed as opposed to our experiments here where despite similar size datasets, only use the original traffic speed as the sole feature. Therefore, Seq2Seq-Attn performs slightly better than AGCN-Seq2Seq-Attn in our work.

- Our SAGCN-SST model achieves the best results on both datasets over all different prediction horizons with an average MAPE less than 2% (i.e., prediction accuracy $> 98\%$). In addition, the MAE and RMSE are 0.8156 and 1.0802, respectively. The closest rivals are Seq2Seq-Attn and AGCN-Seq2Seq-Attn. These models consider the dynamics on temporal feature extraction by taking attention mechanism in the decoder of the sequence-to-sequence architecture. On the other hand, our SAGCN-SST

considers the dynamic process on spatial feature extraction by adopting self-attention mechanism on the graph convolutional layer that can more effectively capture the dynamic spatial dependency between the targeted node and their neighbors in different neighborhoods. The experimental results in this paper also indicate that adopting self-attention mechanism on the graph convolutional layer is more efficient than including it in the decoder of the sequence-to-sequence architecture for traffic prediction. For example, comparing with less than 2% of MAPE achieved by SAGCN-SST, Seq2Seq-Attn and AGCN-Seq2Seq-Attn only obtain less than 3% of MAPE.

In addition, our SAGCN-SST can obtain accurate predictions with small number of features. Between AGCN-Seq2Seq-Attn and SAGCN-SST, our SAGCN-SST model obtains the higher prediction accuracy ($> 98\%$) compared to the 97% accuracy achieved by the AGCN-Seq2Seq-Attn when the experiments are conducted on the same datasets. The main reason is that sub-spatial blocks in SAGCN-SST are paralleled, rather than stacked. This avoids increasing the depth of our model so as to improves scalability and avoid over-fitting at the early stage of training.

Overall, models can achieve more accurate predictions when the traffic prediction problem is treated as a dynamic spatial-temporal process as opposed to considering the problem as 1) a temporal, 2) a spatial or 3) a spatial-temporal and 4) a fixed spatial dynamic-temporal process. This can be clearly observed in Figure 10 where SAGCN-SST, Seq2Seq-Attn and AGCN-Seq2Seq-Attn perform significantly better than the other six models. From the results, we also see that our SAGCN-SST, that is able to capture dynamic spatial features, achieves the best results for both short- and long-term predictions. For $T' = \{1, 3\}$, the next best batch of models are GRU, CNN-LSTM, T-GCN and TGC-LSTM, followed by GCN and SAGCN. This is because GRU, CNN-LSTM, T-GCN and TGC-LSTM are able to capture temporal features, which play a more important role for short-term prediction. For $T' = \{6, 9, 12\}$, CNN-LSTM, T-GCN and TGC-

LSTM still forms the group of models offering the next best results, followed by GRU that has performance similar to GCN and SAGCN, but the difference between all models is smaller than for $T' = \{1, 3\}$. This can be attributed to the increased importance of spatial features in longer prediction horizons.

6. Conclusion

In this paper, we present a novel deep learning model, namely SAGCN-SST, for addressing the multi-step traffic speed prediction problem on large-scale road networks. We claim that the influence of different neighboring road segments towards the future traffic state of a specific road segment of interest are unique and should be considered into the prediction process. Considering traffic speed prediction as a dynamic spatial-temporal process, our model takes advantage of graph convolutional networks, the graph convolutional layer with self-attention mechanism and the sequence-to-sequence architecture, respectively for the fixed spatial, dynamic spatial and long-temporal feature extractions, to make our predictions. We examine our proposed model, SAGCN-SST, on two real-world large-scale road networks: `Loop-Seattle` and `METR-LA`. Traffic congestion frequently occurs for a short duration in `METR-LA` and for longer duration in `Loop-Seattle`. We compare our SAGCN-SST against well-known models in recent literature including: 1) one model treating traffic prediction as a temporal process (i.e., GRU), 2) two models treating traffic prediction as a spatial process (i.e., GCN and SAGCN), 3) three models treating treating traffic prediction as a spatial-temporal process (i.e., CNN-LSTM, T-GCN and TGC-LSTM), and 4) two models treating traffic prediction as fixed spatial dynamic temporal process (i.e., Seq2Seq-Attn and AGCN-Seq2Seq-Attn). Our SAGCN-SST model achieves the highest accuracy among all competing models for both short- and long-term predictions. The average MAE, MAPE and RMSE on both datasets with frequent traffic congestion and accidents are less than 1, 2% and 1.4, respectively, which translate to prediction accuracy being higher than 98%. This results show our SAGCN-SST model is not only accurate but

also robust, recording similar accuracy when making predictions for different prediction horizons including the more challenging long-term prediction.

References

Adolf, D.M., 1990. Traffic flow fundamentals. Englewood Cliffs, NJ: Pretice-Hall.

Ahmed, E., Saint, A., Shabayek, A.E.R., Cherenkova, K., Das, R., Gusev, G., Aouada, D., Ottersten, B., 2019. A survey on deep learning advances on different 3d data representations. Computer Vision and Pattern Recognition 01.

Ahmed, M.S., Cook, A.R., 1979. Analysis of freeway traffic time-series data by using box-jenkins techniques. Transportation Research Record , 1–9.

Bickel, P.J., Chen, C., Kwon, J., Rice, J., Van Zwet, E., Varaiya, P., 2007. Measuring traffic. Statistical Science , 581–597.

Bottou, L., 2010. Large-scale machine learning with stochastic gradient descent, in: Proceedings of COMPSTAT'2010. Springer, pp. 177–186.

Castro-Neto, M., Jeong, Y.S., Jeong, M.K., Han, L.D., 2009. Online-svr for short-term traffic flow prediction under typical and atypical traffic conditions. Expert systems with applications 36, 6164–6173.

Chung, J., Gulcehre, C., Cho, K., Bengio, Y., 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling, in: NIPS 2014 Workshop on Deep Learning, December 2014.

Cui, Z., Henrickson, K., Ke, R., Wang, Y., 2019. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. IEEE Transactions on Intelligent Transportation Systems 21, 4883–4894.

Fu, R., Zhang, Z., Li, L., 2016. Using lstm and gru neural network methods for traffic flow prediction, in: 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), IEEE. pp. 324–328.

Gowrishankar, Satyanarayana, P., 2008. Neural network based traffic prediction for wireless data networks. International Journal of Computational Intelligence Systems 1, 379–389.

Grover, A., Leskovec, J., 2016. node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855–864.

Guo, K., Hu, Y., Qian, Z., Liu, H., Zhang, K., Sun, Y., Gao, J., Yin, B., 2020. Optimized graph convolution recurrent neural network for traffic prediction. IEEE Transactions on Intelligent Transportation Systems 22, 1138–1149.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural computation 9, 1735–1780.

Huang, W., Song, G., Hong, H., Xie, K., 2014. Deep architecture for traffic flow prediction: deep belief networks with multitask learning. IEEE Transactions on Intelligent Transportation Systems 15, 2191–2201.

Jagadish, H., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J.M., Ramakrishnan, R., Shahabi, C., 2014. Big data and its technical challenges. Communications of the ACM 57, 86–94.

Jia, Y., Wu, J., Du, Y., 2016. Traffic speed prediction using deep learning method, in: IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), IEEE. pp. 1217–1222.

Kang, D., Lv, Y., Chen, Y.y., 2017. Short-term traffic flow prediction with lstm recurrent neural network, in: IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), IEEE. pp. 1–6.

Ke, J., Zheng, H., Yang, H., Chen, X.M., 2017. Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach. Transportation Research Part C: Emerging Technologies 85, 591–608.

Kipf, T.N., Welling, M., 2017. Semi-supervised classification with graph convolutional networks, in: Proceedings of the International Conference on Learning Representations (ICLR).

Lee, S., Fambro, D.B., 1999. Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting. Transportation Research Record 1678, 179–188.

Li, Y., Yu, R., Shahabi, C., Liu, Y., 2018. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, in: Proceedings of the International Conference on Learning Representations (ICLR).

Long, J., Gao, Z., Ren, H., Lian, A., 2008. Urban traffic congestion propagation and bottleneck identification. Science in China Series F: Information Sciences 51, 948.

Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F.Y., 2014. Traffic flow prediction with big data: a deep learning approach. IEEE Transactions on Intelligent Transportation Systems 16, 865–873.

Lv, Y., et al., 2015. Traffic flow prediction with big data: A deep learning approach. IEEE Transactions on Intelligent Transportation Systems 16, 865–873.

Lv, Z., Xu, J., Zheng, K., Yin, H., Zhao, P., Zhou, X., 2018. Lc-rnn: A deep learning model for traffic speed prediction, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI), pp. 3470–3476.

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

Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y., 2015. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transportation Research Part C: Emerging Technologies 54, 187–197.

Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., Khudanpur, S., 2010. Recurrent neural network based language model, in: the 11th annual conference of the international speech communication association, pp. 1045–1048.

Min, W., Wynter, L., 2011. Real-time road traffic prediction with spatio-temporal correlations. Transportation Research Part C: Emerging Technologies 19, 606–616.

Nguyen, H., Liu, W., Chen, F., 2016. Discovering congestion propagation patterns in spatio-temporal traffic data. IEEE Transactions on Big Data 3, 169–180.

Shi, X., Qi, H., Shen, Y., Wu, G., Yin, B., 2020. A spatial-temporal attention approach for traffic prediction. IEEE Transactions on Intelligent Transportation Systems .

Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A., Vandergheynst, P., 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE signal processing magazine 30, 83–98.

Sutskever, I., Vinyals, O., Le, Q.V., 2014. Sequence to sequence learning with neural networks, in: Advances in neural information processing systems, pp. 3104–3112.

Tan, M.C., Wong, S.C., Xu, J.M., Guan, Z.R., Zhang, P., 2009. An aggregation approach to short-term traffic flow prediction. IEEE Transactions on Intelligent Transportation Systems 10, 60–69.

Tian, Y., Pan, L., 2015. Predicting short-term traffic flow by long short-term memory recurrent neural network, in: 2015 IEEE international conference on smart city/SocialCom/SustainCom (SmartCity), IEEE. pp. 153–158.

Tieleman, T., Hinton, G., 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning 4, 26–31.

Van Der Voort, M., Dougherty, M., Watson, S., 1996. Combining kohonen maps with arima time series models to forecast traffic flow. Transportation Research Part C: Emerging Technologies 4, 307–318.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 6000–6010.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., 2018. Graph attention networks, in: Proceedings of the International Conference on Learning Representations (ICLR).

Wang, F.Y., 2010. Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications. IEEE Transactions on Intelligent Transportation Systems 11, 630–638.

Williams, B.M., Hoel, L.A., 2003. Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results. Journal of Transportation Engineering 129, 664–672.

Xu, M., Dai, W., Liu, C., Gao, X., Lin, W., Qi, G.J., Xiong, H., 2020. Spatial-temporal transformer networks for traffic flow forecasting. Electrical Engineering and Systems Science - Signal Processing .

Yu, B., Yin, H., Zhu, Z., 2018. Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, pp. 3634–3640.

Yu, H., Wu, Z., Wang, S., Wang, Y., Ma, X., 2017. Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. Sensors 17, 1501.

Zang, D., Ling, J., Wei, Z., Tang, K., Cheng, J., 2018. Long-term traffic speed prediction based on multiscale spatio-temporal feature learning network. IEEE Transactions on Intelligent Transportation Systems 20, 3700–3709.

Zhang, G.P., 2003. Time series forecasting using a hybrid arima and neural network model. Neurocomputing 50, 159–175.

Zhang, Z., Li, M., Lin, X., Wang, Y., He, F., 2019. Multistep speed prediction on traffic networks: A deep learning approach considering spatio-temporal dependencies. Transportation Research Part C: Emerging Technologies 105, 297–322.

Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., Deng, M., Li, H., 2019. T-gcn: A temporal graph convolutional network for traffic prediction. IEEE Transactions on Intelligent Transportation Systems .

Zheng, C., Fan, X., Wang, C., Qi, J., 2020. Gman: A graph multi-attention network for traffic prediction, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 1234–1241.

Zheng, G., Chai, W.K., Katos, V., 2019. An ensemble model for short-term traffic prediction in smart city transportation system, in: IEEE Global Communications Conference (GLOBECOM), pp. 1–6. doi:10.1109/GLOBECOM38437.2019.9014061.

Zheng, G., Chai, W.K., Katos, V., 2021a. The sequence-to-sequence architecture with an embedded module for long-term traffic speed forecasting with missing data, in: Proceedings of the 26th International Conference on Automation and Computing.

Zheng, G., Chai, W.K., Katos, V., Walton, M., 2021b. A joint temporal-spatial ensemble model for short-term traffic prediction. Neurocomputing 457, 26–39.