Bilinear Multi-Head Attention Graph Neural Network for Traffic Prediction

Haibing Hu[a], Kai Han and Zhizhuo Yin
School of Computer Science and Technology, University of Science and Technology of China, Hefei, China

Keywords: Bilinear Aggregator, Graph Neural Networks, Traffic Forecasting, Multi Attention.

Abstract: Traffic forecasting is an important component of Intelligent Transportation System (ITS) and it has the significance for reducing traffic accidents and improving public safety. Due to the complex spatial-temporal dependencies and the uncertainty of road network, the research on this problem is quite challenging. Some of the latest studies utilize graph convolutional networks (GCNs) to model spatial-temporal relationships. However, these methods are only based on the linear weighted summation of the neighborhood to form the representation of the target node, which cannot capture the signal between pairwise node interactions. In many scenes, adding pairwise node interaction features is an essential way to better represent the target node. Therefore, in this article, we propose an end-to-end novel framework named Bilinear Multi-Head Attention Graph Neural Network (BMHA-GNN) for traffic prediction. We propose a new aggregation operator which utilizes the weighted sum of pairwise interactions of the neighbour nodes and improves the representation ability of GCN based models. We adopt the encoder-decoder framework, the encoder module outputs the representation of traffic data, and the decoder module outputs the prediction results. The multi-head attention mechanism is introduced to aggregate information of different neighbour nodes automatically and stabilize the training process. Extensive experiments are conducted on two real-world datasets (METR-LA, PEMS-BAY) showing that the proposed model BMHA-GNN achieves the state-of-the-art results.

1 INTRODUCTION

Traffic forecasting is a critical factor in the intelligent transportation system (ITS), which is also vital to public safety. The primary task of traffic prediction is to predict the future traffic conditions (e.g., traffic speed or volume) of the road network based on historical data.

Due to the complexity of the spatial-temporal dependencies and uncertainty of road network, this task is highly challenging. In order to overcome these difficulties, a lot of research has been put forward in recent years. Early research is mainly based on classic machine learning methods (D. et al., 2005; S.I.J. and C.M., 2003), which can't express the non-linearity of traffic data exactly. The latest methods based on deep learning can model complex spatial-temporal dependencies and capture higher-order nonlinear features better. Methods based on Convolutional Neural Networks (CNNs) (Yao et al., 2018b; Yao et al., 2018a) and Recurrent Neural Networks (RNNs) (Ma et al., 2017; Xuan et al., 2016) have been proposed recently, but CNNs are better at processing the grid-structure data, such as image and video, etc. In order to utilize convolution operations in non-euclidean scenarios, Graph Convolutional Networks (GCNs) or Graph Neural Networks (GNNs) (Li et al., 2018; Yu et al., 2017) related methods are proposed. Although current GCN-based methods have achieved good performance in this field, the existing GCNs methods only utilize the linear weighted sum of the neighborhood nodes to update the target node when defining graph convolution. It is based on an assumption that the nodes in the neighborhood are mutually independent and the possible feature interactions between them are ignored. However, it is an essential signal to represent the target node. For example, the simultaneous appearance of the time node at the morning peak and the evening peak will affect the target time node representation in the temporal network. Although the use of many powerful feature transformation functions such as multi-layer perceptron (MLP) (Xu et al., 2019; Zhu et al., 2020) can alleviate this problem, this process is ineffective and implicit. An empirical evidence comes from (Beutel et al., 2018), showing that

[a] https://orcid.org/0000-0003-1207-4472

Hu, H., Han, K. and Yin, Z.
Bilinear Multi-Head Attention Graph Neural Network for Traffic Prediction.
DOI: 10.5220/0010763400003116
In *Proceedings of the 14th International Conference on Agents and Artificial Intelligence (ICAART 2022) - Volume 2*, pages 33-43
ISBN: 978-989-758-547-0; ISSN: 2184-433X

MLP is not sufficiently effective to capture multiplication relationship between input features.

In order to address the aforementioned challenges, we propose a new GNNs model called Bilinear Multi-Head Attention Graph Neural Network (BMHA-GNN) for traffic prediction, where we define a new aggregation operator of GNN. We not only use the linear weighted summation of the nodes to represent the target node, but also explicitly adopt the pair-wise node interaction features to represent the target node, which can better model the non-linearity of the spatial-temporal data. The main contributions of our work are summarized as follows:

- We propose an end-to-end BMHA-GNN model to explicitly model the nonlinear interaction features of spatial and temporal nodes, and adopt gated fusion to adaptively utilize spatial and temporal information.
- We propose a new aggregation operator for bilinear graph convolution in the traffic prediction field. To the best of our knowledge, we are the first to explicitly use the pair-wise interaction features between nodes in the traffic prediction research.
- Extensive experiments are carried out on two real-world traffic datasets METR-LA and PEMS-BAY on our work BMHA-GNN, and the results show that our proposed model achieves the state-of-the-art results.

The rest of this article is organized as follows. In section 2, we introduce the background and recent progress of traffic forecasting. In section 3, we define the problem that we need to solve through mathematical formulas. In section 4, we introduce the structure of the proposed model in detail. In section 5, we compare the experimental results with the state-of-the-art methods and do experimental analysis. Finally, we come to the conclusion of this article and look forward to future work.

2 RELATED WORKS

Traffic prediction has been extensively researched in recent years. Some of the earliest methods are based on shallow machine learning methods, such as logistic regression (Nikovski et al., 2005), k-nearest neighbor (KNN) (Zheng and Su, 2014) and support vector regression (SVR) (Chun-Hsin Wu et al., 2004). However, these methods cannot make good use of high-order nonlinear features and can't capture the dependencies of spatial-temporal, which make the prediction effect poor.

In order to better model the spatial relationship, researchers use convolutional neural networks (CNNs) (Yao et al., 2018b; Yao et al., 2018a) to model the spatial dependencies. However, the data processed by CNNs need to be in the euclidean space, these methods are not good at processing non-euclidean road network data. Therefore, Graph neural networks (GNNs) methods (Li et al., 2018; Yu et al., 2017) are proposed to deal with non-euclidean traffic data. There are two main categories of existing GNN models: spectral GNNs (Bruna et al., 2014) and spatial GNNs (Atwood and Towsley, 2015). Spectral GNN are defined as conducting convolution operations in the fourier domain with spectral node representations. By aggregating the characteristics of a target node from spatially related neighbors, Spatial GNNs perform convolution operations directly over the structure of the graph.

Recent years, there have been extensive GNN-based models proposed to model non-euclidean traffic network data. Li et al (Li et al., 2018) proposed Diffusion Convolutional Recurrent Neural Network (DCRNN) which uses the diffusion graph convolution operator to replace the fullly-connected layers in Gated Recurrent Units (GRU) (Chung et al., 2014). Zhang et al (Q. et al., 2020) proposed Spatial-Temporal Graph Structure Learning (SLCNN) which enables to extend the traditional convolution neural network (CNN) to graph domains and learns the graph structure for traffic forecasting. Yu et al (Yu et al., 2017) proposed Spatial-Temporal Graph Convolutional Networks (STGCN) to tackle the time series prediction problem in traffic domain. They formulated the problem on graphs and built the model with complete convolutional structures. Song et al (Song et al., 2020) proposed Spatial-Temporal Synchronous Graph Convolutional Networks (STSGCN), through an alaborately designed spatial-temporal synchronous modeling mechanism, the model is able to effectively capture the complex localized spatial-temporal correlations. Guo et al (S. et al., 2019) proposed Attention based Spatial-Temporal Graph Convolution Network (ASTGCN), which uses a spatial-temporal attention mechanism to learn the dynamic spatial-temporal correlations of traffic data. Zheng et al (Zheng et al., 2020) proposed a Graph multi-attention network (GAMN) to predict traffic conditions for time steps, which adapts an encoder-decoder architecture. Chen et al (Chen et al., 2019) proposed a Multi-Range Attentive Bicomponent GCN (MRA-BGCN), which implements the interactions of both nodes and edges using bicomponent graph convolution. Wu et al (Wu et al., 2019) proposed a Graph WaveNet for Deep Spatial-Temporal Graph Modeling, which can handle very long sequences with a stacked dilated 1D convolution component. However, they do not use the interaction features between nodes, which can express non-linearity better.

3 PRELIMINARIES

First of all, we use $\mathcal{G} = (V, E, A)$ to represent the spatial network of traffic data, where V is the set of vertices, $|V| = N$ is the number of network vertices, E is the set of edges and A is the adjacency matrix of network $\mathcal{G}$;

The traffic condition at time step t is represented as a graph signal matrix $X^t_{\mathcal{G}} \in R^{N \times C}$, where C is the number of attribute features.

Therefore the problem of spatial-temporal network can be defined as follows: Given the historical spatial-temporal network series data $[X^{t-P+1}_{\mathcal{G}}, X^{t-P+2}_{\mathcal{G}}, ..., X^t_{\mathcal{G}}]$, we need to learn a function f, which can map the historical data into the future observations $[X^{t+1}_{\mathcal{G}}, X^{t+2}_{\mathcal{G}}, ..., X^{t+Q}_{\mathcal{G}}]$, that is,

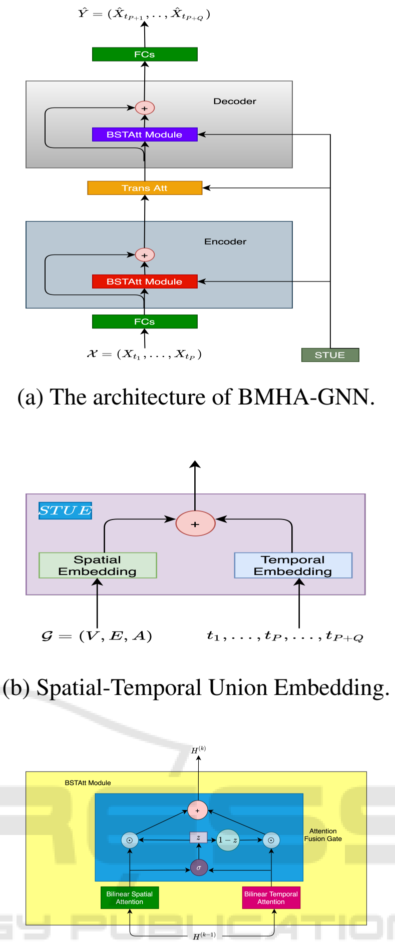
$$[X^{t-P+1}_{\mathcal{G}}, X^{t-P+2}_{\mathcal{G}}, ..., X^t_{\mathcal{G}}] \xrightarrow{f} [X^{t+1}_{\mathcal{G}}, X^{t+2}_{\mathcal{G}}, ..., X^{t+Q}_{\mathcal{G}}], \tag{1}$$

where P represents time steps of historical data, and Q represents time steps of future predicted data.

4 BILINEAR MULTI-HEAD ATTENTION GRAPH NEURAL NETWORK

4.1 Model Overview

We show our model architecture comprehensively in Figure 1. We adopt an end-to-end encoder-decoder framework, for spatial embedding and temporal embedding, we not only use first-order linearly weighted features, but also use the pair-wise second-order feature interactions (Zhu et al., 2020), which can better capture the non-linearity relationship in spatial and temporal nodes. Both encoder and decoder contain K Bilinear Spatial-Temporal Attentional blocks (BSTAtt), each block contains three components, bilinear spatial attention, bilinear temporal attention and an attention fusion gate. A transform layer is designed between encoder and decoder layer to convert the output of encoder feature to decoder. By a spatial-temporal union embedding (STUE), we incorporate the graph structure and time information into the multi-head attention mechanisms. We introduce each module as follows in detail.


$\hat{Y} = (\hat{X}_{t_{P+1}}, .., \hat{X}_{t_{P+Q}})$
FCs
Decoder
+
BSTAtt Module
Trans Att
Encoder
+
BSTAtt Module
FCs
$\mathcal{X} = (X_{t_1}, ..., X_{t_P})$
STUE

(a) The architecture of BMHA-GNN.

STUE
+
Spatial Embedding
Temporal Embedding
$\mathcal{G} = (V, E, A)$
$t_1, ..., t_P, ..., t_{P+Q}$

(b) Spatial-Temporal Union Embedding.

$H^{(k)}$
BSTAtt Module
+
⊙
z
$1 - z$
⊙
Attention Fusion Gate
σ
Bilinear Spatial Attention
Bilinear Temporal Attention
$H^{(k-1)}$

(c) BSTAtt Module.

Figure 1: The framework of Bilinear Multi-Head Attention Graph Neural Network. (a) The architecture of BMHA-GNN. (b) Spatial-Temporal Union Embedding. (c) BSTAtt Module.

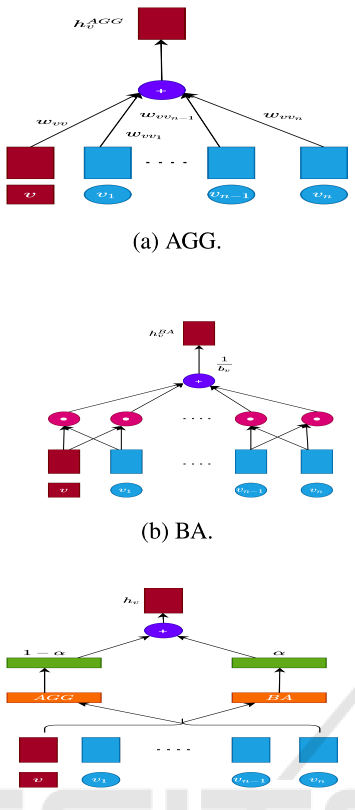
4.2 Bilinear Aggregator

In this part, we will introduce the aggregation operators of GNN. Let $\mathcal{G} = (V, E, A)$ to represent the spatial network, and A is the adjacency matrix, $A \in \{0, 1\}^{N \times N}$, $A_{ij} = 1$ means that an edge exists between node i and node j. $\mathcal{N}(v) = \{i | A_{vi} = 1\}$, is a set of all nodes which has an edge with node v. $\widetilde{\mathcal{N}}(v) = v \cup \mathcal{N}(v)$. We use $d_v = |\mathcal{N}(v)|$ to denote the degree of node v, $\widetilde{d_v} = d_v + 1$.

By recursively aggregating the features from neighbors, the spatial GNN can achieves the goal to learn a representation vector $h_v \in R^D$ for each node v.

$$h^{(k)}_v = AGG(\{h^{(k-1)}_i\}_{i \in \widetilde{\mathcal{N}}(v)}), \tag{2}$$

where AGG represents a linear weighted sum function



(a) AGG.

(b) BA.

(c) BGNN.

Figure 2: Aggregators in GNN; (a) is linear aggregator; (b) is bilinear aggregator; (c) is BGNN aggregator.

of the neighborhood, $h_i^{(k-1)}, h_v^{(k)}$ denotes the representation of node i, v at the $k-1$-th and k-th iteration, respectively.

However, this only uses first-order nonlinear features and does not use high-order feature combinations that can better represent the target node. Although a method similar to MLP can alleviate this problem, it is implicit and inefficient (Beutel et al., 2018).

Inspired by factorization machines (FMs) (Rendle, 2010), which have been intensively used to learn the interactions among categorical variables and is an effective manner to model the interaction, we can define a bilinear aggregator (Zhu et al., 2020) for GNN to model the neighbor node interactions in local structure.

$$BA(\{h_i\}_{i\in\widetilde{\mathcal{N}}(v)}) = \frac{1}{b_v}\sum_{i\in\widetilde{\mathcal{N}}(v)}\sum_{j\in\widetilde{\mathcal{N}}(v)\&i<j} h_iW \odot h_jW, \tag{3}$$

where $\odot$ is element-wise product, v is the target node, i and j are node index from $\widetilde{\mathcal{N}}(v)$, $b_v = \frac{1}{2}\tilde{d}_v(\tilde{d}_v - 1)$ denotes the total number of interactions for node v, W is the model parameter.

Similar to the mathematical re-formulation process in FM (Rendle, 2010), we can transform the formula (3) into the following equation:

$$\begin{aligned} BA(\{h_i\}_{i\in\widetilde{\mathcal{N}}(v)}) &= \frac{1}{2b_v}\Bigg(\sum_{i\in\widetilde{\mathcal{N}}(v)}\sum_{j\in\widetilde{\mathcal{N}}(v)} s_i \odot s_j \\ &\quad - \sum_{i\in\widetilde{\mathcal{N}}(v)} s_i \odot s_i\Bigg) \\ &= \frac{1}{2b_v}\Bigg(\Big(\sum_{i\in\widetilde{\mathcal{N}}(v)} s_i\Big)^2 - \sum_{i\in\widetilde{\mathcal{N}}(v)} s_i^2\Bigg), \end{aligned} \tag{4}$$

where $s_i = h_iW \in R^D$. From (Zhu et al., 2020), the bilinear aggregator is permutation invariant and the time complexity is $O(|\widetilde{\mathcal{N}}(v)|)$.

Then, we can define a new graph convolution operator as follows:

$$\begin{aligned} H^{(k)} &= BGNN(H^{(k-1)}, A) \\ &= (1-\alpha)\cdot AGG(H^{(k-1)}, A) + \alpha \cdot BA(H^{(k-1)}, A), \end{aligned} \tag{5}$$

where α is a hyperparameter to adjust the weight of the traditional GNN aggregator and bilinear aggregator, and $H^{(k)}$ is the node representation at the k-th layer. Figure 2 illustrates three different GNN aggregators.

We can also define the 2-layer $BGNN$ model as follows:

$$\begin{aligned} BGNN_2(X, A) = (1-\alpha)\cdot GNN_2(X, A) \\ + \alpha[(1-\beta)\cdot BA(X, A) \\ + \beta \cdot BA(X, A^{(2)})], \end{aligned} \tag{6}$$

where

$$GNN_2(H^{(k)}, A) = AGG(\sigma(AGG(H^{(k-1)}, A)), A)$$

represents the 2-layer GNN, σ is a non-linear activation function, β represents the strengths of bilinear interaction within 1-hop neighbors and 2-hop neighbors and $A^{(2)} = binarize(AA)$ stores the 2-hop connectivities of the graph. *binarize* is the operation of specializing non-zero elements into 1.

4.3 Spatial-temporal Union Embedding

We introduce the spatial-temporal union embedding ($STUE$) in this part. Follow the node2vec approach (Grover and Leskovec, 2016), a spatial embedding was proposed (Zheng et al., 2020) to encode vertices into vectors, which can preserve the graph structure information. By co-train the pre-learned vectors with the whole model and feed these vectors into a fully-connected neural network, we can obtain the spatial embedding $e^S_{v_i} \in R^D$, where $v_i \in V$. However, we can only get the static spatial embedding, which cannot represent the road network that changes according to time. For this reason, we also propose time embedding similar to (Zheng et al., 2020), we encode the time-of-day and day-of-week of each time step into R^T and R^7 by one-hot encoding, concatenate these two into a vector of R^{T+7}, and feed it into a fully-connected neural network, we can get a vector of R^D, which is represented as $e^T_{t_j} \in R^D$, where $1 \leq j \leq P+Q$, P stands for the historical time steps, and Q stands for the future time steps. For vertex v_i at time step t_j, we can obtain the $STUE$ embedding by spatial embedding and time embedding, that is, $e_{v_i,t_j} = f(e^S_{v_i}, e^S_{t_j})$, f is a function. For simplicity, f can be defined as the summation of two vectors. Thus, $STUE$ contains both spatial information and temporal information.

4.4 Bilinear Spatial-temporal Attention Block

As shown in Figure 1 (c), there are three components in the BSTAtt Block, bilinear spatial attention, bilinear temporal attention and an attention fusion gate. We represent the input of k^{th} block as $H^{(k-1)}$, $h^{(k-1)}_{v_i,t_j}$ as the representation of the hidden state of vertex v_i at time step t_j. We denote $H^{(k)}_S$, $H^{(k)}_T$ as the representation of the output of bilinear spatial and bilinear temporal attention in the k^{th} block, where $hs^{(k)}_{v_i,t_j}$ and $ht^{(k)}_{v_i,t_j}$ represents the hidden state of vertex v_i at time step t_j, respectively. We obtain the output of k^{th} block after the attention fusion gate, denoted as $H^{(k)}$.

4.4.1 Bilinear Spatial Attention

In order to adaptively caputure the correlations between sensors in the road network, we design a bilinear spatial attention mechanism to represent the target node embedding. For node v_i at time step t_j, we can obtain a first order weighted sum from all vertices.

$$\begin{aligned} hs^{(k)}_{v_i,t_j} &= AGG(h^{(k-1)}_{v,t_j}) \\ &= \sum_{v \in V} \alpha_{v_i,v} \cdot h^{(k-1)}_{v,t_j} \end{aligned} \tag{7}$$

We compute the relevance between vertex v_i and v by concatenate the hidden state with spatial-temporal union embedding and adopt the scaled dot-product approach (Vaswani et al., 2017).

$$s_{v_i,v} = \frac{\left\langle h^{(k-1)}_{v_i,t_j} \| e_{v_i,t_j}, h^{(k-1)}_{v,t_j} \| e_{v,t_j} \right\rangle}{\sqrt{2D}}, \tag{8}$$

where $\|$ represents the concatenation operation, $\langle \bullet, \bullet \rangle$ represents the inner product operator. Via softmax, $s_{v_i,v}$ is normalized as:

$$\alpha_{v_i,v} = \frac{exp(s_{v_i,v})}{\sum_{v_r \in V} exp(s_{v_i,v_r})}. \tag{9}$$

Similar to (Zheng et al., 2020), we also extend the spatial embedding mechanism to multi-head ones to stabilize the learning process. L denotes the total number of parallel attention.

$$s^{(l)}_{v_i,v} = \frac{\left\langle f^{(l)}_{s,2}(h^{(k-1)}_{v_i,t_j} \| e_{v_i,t_j}), f^{(l)}_{s,3}(h^{(k-1)}_{v,t_j} \| e_{v,t_j}) \right\rangle}{\sqrt{d}}, \tag{10}$$

$$\alpha^{(l)}_{v_i,v} = \frac{exp(s^{(l)}_{v_i,v})}{\sum_{v_r \in V} exp(s^{(l)}_{v_i,v_r})}, \tag{11}$$

$$\begin{aligned} hs^{(k)}_{v_i,t_j} &= AGG(h^{(k-1)}_{v_i,t_j}) \\ &= \|_{l=1}^{L} \left\{ \sum_{v \in V} \alpha^{(l)}_{v_i,v} \cdot f^{(l)}_{s,1}(h^{(k-1)}_{v,t_j}) \right\}, \end{aligned} \tag{12}$$

where $f^{(l)}_{s,1}(\bullet), f^{(l)}_{s,2}(\bullet)$, and $f^{(l)}_{s,3}(\bullet)$ denote different non-linear projections in the l^{th} attention head, and $d = D/L$.

We define the spatial second-order interactions weighted sum as follows:

$$\begin{aligned} hs^{(k)}_{v_i,t_j} &= BA(\{h^{(k-1)}_{v_m,t_j}\}_{v_m \in \widetilde{\mathcal{N}}(v_i)}) \\ &= \frac{1}{2b_v} \left(\sum_{v_m \in \widetilde{\mathcal{N}}(v_i)} \sum_{v_n \in \widetilde{\mathcal{N}}(v_i)} s_{v_m} \odot s_{v_n} \right. \\ &\quad \left. - \sum_{v_m \in \widetilde{\mathcal{N}}(v_i)} s_{v_m} \odot s_{v_m} \right) \\ &= \frac{1}{2b_v} \left(\left(\sum_{v_m \in \widetilde{\mathcal{N}}(v_i)} s_{v_m} \right)^2 - \sum_{v_m \in \widetilde{\mathcal{N}}(v_i)} s^2_{v_m} \right), \end{aligned} \tag{13}$$

where $s_{v_m} = h^{(k-1)}_{v_m,t_j} W \in R^D$, $s_{v_n} = h^{(k-1)}_{v_n,t_j} W \in R^D$ and $b_v = \frac{1}{2} \widetilde{d}_v(\widetilde{d}_v - 1)$ denotes the total number of interactions for node v_i.

Thus, the bilinear aggregator is defined beblow:

$$\begin{aligned} H^{(k)}_S &= BGNN(H^{(k-1)}, A) \\ &= (1-\alpha) \cdot AGG(H^{(k-1)}, A) + \alpha \cdot BA(H^{(k-1)}, A), \end{aligned} \tag{14}$$

where $H_S^{(k)} \in R^{T\times N\times D}$ stores the node representations at the k-th layer, $T = P$ in encoder module and $T = Q$ in decoder module. And α is a hyper-parameter to adjust the traditional GNN aggregator and bilinear aggregator. We can also define the 2-layer GNN model

$$GNN_2(H^{(k-1)},A) = AGG(\sigma(AGG(H^{(k-1)},A)),A), \tag{15}$$

$$\begin{aligned} H_S^{(k)} &= BGNN_2(H^{(k-1)},A) \\ &= (1-\alpha)\cdot GNN_2(H^{(k-1)},A) \\ &+ \alpha[(1-\beta)\cdot BA(H^{(k-1)},A) \\ &+ \beta\cdot BA(H^{k-1},A^{(2)})], \end{aligned} \tag{16}$$

where $A^{(2)} = binarize(AA)$ stores the 2-hop connectivities of the graph, β represents the strengths of bilinear interaction within 1-hop neighbors and 2-hop neighbors and σ is a non-linear activation function.

4.4.2 Bilinear Temporal Attention

As for vertex v_i, we define the correlation between time step t_j and t as follows, the process is similar to bilinear spatial attention. The difference is that, we only consider the time information earlier than the target step.

$$u_{t_j,t}^{(l)} = \frac{\left\langle f_{t,2}^{(l)}(h_{v_i,t_j}^{(k-1)}\|e_{v_i,t_j}), f_{t,3}^{(l)}(h_{v_i,t}^{(k-1)}\|e_{v_i,t})\right\rangle}{\sqrt{d}}, \tag{17}$$

$$\beta_{t_j,t}^{(l)} = \frac{exp(u_{t_j,t}^{(l)})}{\sum_{t_r\in\mathcal{N}_{t_j}} exp(u_{t_j,t_r}^{(l)})}, \tag{18}$$

$$\begin{aligned} ht_{v_i,t_j}^{(k)} &= AGG(h_{v_i,t}^{(k-1)}) \\ &= \|_{l=1}^{L}\left\{\sum_{t\in\mathcal{N}_{t_j}} \beta_{t_j,t}^{(l)}\cdot f_{t,1}^{(l)}(h_{v_i,t}^{(k-1)})\right\}, \end{aligned} \tag{19}$$

where N_{t_j} stands for a set of time steps before t_j.

We define the temporal second-order interactions weighted sum as follows:

$$\begin{aligned} ht_{v_i,t_j}^{(k)} &= BA(\{h_{v_i,t_m}^{(k-1)}\}_{t_m\in N_{t_j}}) \\ &= \frac{1}{2b_t}\left(\sum_{t_m\in N_{t_j}}\sum_{t_n\in N_{t_j}} s_{t_m}\odot s_{t_n} - \sum_{t_m\in N_{t_j}} s_{t_m}\odot s_{t_m}\right) \\ &= \frac{1}{2b_t}\left(\left(\sum_{t_m\in N_{t_j}} s_{t_m}\right)^2 - \sum_{t_m\in N_{t_j}} s_{t_m}^2\right), \end{aligned} \tag{20}$$

where $s_{t_m} = h_{v_i,t_m}^{(k-1)}W \in R^D$, $s_{t_n} = h_{v_i,t_n}^{(k-1)}W \in R^D$, $b_t = \frac{1}{2}|N_{t_j}|(|N_{t_j}|-1)$ denotes the total number of interactions for node t_j and $|N_{t_j}|$ denotes the number of time steps before t_j.

Similar to bilinear spatial embedding, we can also define bilinear temporal embedding.

$$\begin{aligned} H_T^{(k)} &= BGNN_2(H^{(k-1)},A) \\ &= (1-\alpha)\cdot GNN_2(H^{(k-1)},A) \\ &+ \alpha[(1-\beta)\cdot BA(H^{(k-1)},A) \\ &+ \beta\cdot BA(H^{(k-1)},A^{(2)})], \end{aligned} \tag{21}$$

where $A^{(2)} = binarize(AA)$ stores the 2-hop connectivities of the graph, $H_T^{(k)} \in R^{T\times N\times D}$, $T = P$ in encoder module and $T = Q$ in decoder module. α, β has same effect as the previous section. GNN_2 has the same definition as equation 15.

4.4.3 Attention Fusion Gate

In this part, we design an attention fusion gate (AFG) to adatively use bilinear spatial embedding and bilinear temporal embedding representations. In the k^{th} block, $H_S^{(k)}$, $H_T^{(k)}$ denotes the output of bilinear spatial attention and bilinear temporal attention, respectively. They both have the shapes of $R^{P\times N\times D}$ in the encoder or $R^{Q\times N\times D}$ in the decoder. We define the attention fusion gate as follows:

$$H^{(k)} = \eta\odot H_S^{(k)} + (1-\eta)\odot H_T^{(k)} \tag{22}$$

with

$$\eta = \sigma(H_S^{(k)}W_{\eta,1} + H_T^{(k)}W_{\eta,2} + b_\eta), \tag{23}$$

where $W_{\eta,1}, W_{\eta,2} \in R^{D\times D}$, $b_\eta \in R^D$ are learnable parameters, $\odot$ means the element-wise product, $\sigma(\bullet)$ represents the sigmoid function and η denotes the gate.

4.5 Transform Attention

Between the encoder and decoder module, we design a transform attention layer, which can ease the error propagation effect between different time steps in the long time horizon (Zheng et al., 2020). By Spatial and Temporal Union Embedding($STUE$), we can define the relevance between the historical time step $t(t = t_1,\ldots,t_P)$ and the prediction time step $t_j(t_j = t_{P+1},\ldots,t_{P+Q})$ as follows:

$$\lambda_{t_j,t}^{(l)} = \frac{\left\langle f_{t_s,1}^{(l)}(e_{v_i},t_j), f_{t_s,2}^{(l)}(e_{v_i},t)\right\rangle}{\sqrt{d}} \tag{24}$$

$$\eta_{t_j,t}^{(l)} = \frac{exp(\lambda_{t_j,t}^{(l)})}{\sum_{t_s=t_1}^{t_P} exp(\lambda_{t_j,t_s}^{(l)})} \tag{25}$$

by adaptively selecting relevant features from all historical P time steps, the encoded traffic feature is transformed to the decoder with the attention score $\eta_{t_j,t}^{(l)}$.

$$h_{v_i,t_j}^{(k)} = \|_{l=1}^{L} \left\{ \sum_{t=t_1}^{t_P} \eta_{t_j,t}^{(l)} \cdot f_{t_s,3}^{(l)}(h_{v_i,t}^{(k-1)}) \right\}, \tag{26}$$

where $f_{t_s,1}^{(l)}, f_{t_s,2}^{(l)}, f_{t_s,3}^{(l)}$ are shared learnable parameters by all vertices and time steps.

4.6 Encoder-decoder Framework

In Figure 1, we fully demonstrate the architecture of our model, which uses an end-to-end encoder-decoder structure. We summarize the pipeline and tensor dimensions of our model in Figure 3. Firstly, we obtain the historical data $X \in R^{P \times N \times C}$. After a two-layer fully connected network, we obtain $H^{(0)} \in R^{P \times N \times D}$ as the input of the encoder. After K *BSTAtt* blocks, we obtain the output of the encoder $H^{(K)} \in R^{P \times N \times D}$. We obtain $H^{(K+1)} \in R^{Q \times N \times D}$ after a transform module. We obtain $H^{(2K+1)}$ after K *BSTAtt* decoder blocks and feed it into two fully connected network, we obtain the final predict value $\hat{Y} \in R^{Q \times N \times C}$.

4.7 Loss Function

We select mean absolute error (MAE) as our loss function.

$$\mathcal{L}(\Theta) = \frac{1}{Q} \sum_{t=t_{P+1}}^{t_{P+Q}} |Y_t - \hat{Y}_t|, \tag{27}$$

where Θ represents all learnable parameters in BMHA-GNN, Y_t and $\hat{Y}_t$ denote the ground truth and predict value at time step t, respectively.

5 EXPERIMENTS

5.1 Datasets

We evaluate BMHA-GNN on two different public traffic network datasets, METR-LA and PEMS-BAY (Li et al., 2018). METR-LA estimates four months of traffic velocity figures, spanning from March 1st 2012 to June 30th 2012, including 207 sensors on Los Angeles County highways. PEMS-BAY provides five months of traffic speed figures, spanning from January 1st 2017 to May 31th 2017, with 325 sensors in the BAY area. We follow the same protocols for data pre-processing as Li et al (Li et al., 2018). Sensors' observations are aggregated into 5-minute windows and the data is normalized via the *Z-Score* method. The details of the dataset are listed in Table 1 and the distribution of sensors are visualized in Figure 4. According to some previous practices (Li et al., 2018), we divide the dataset into training set, validation set, and test set, with a ratio of 7:1:2. Each traffic sensor is considered as a vertex and the node-wise graph's adjacency matrix is constructed by the road network distance with the Gaussian kernel threshold (Shuman et al., 2012). We define the adjacency matrix A similar to (Zheng et al., 2020) as follows:

Table 1: Details of PEMS-BAY and METR-LA.

Dataset	♯ Nodes	♯ Edges	♯ Time-Steps
PEMS-BAY	325	2369	52116
METR-LA	207	1515	34272

$$A_{v_x,v_y} = \begin{cases} exp(-\frac{d_{v_x,v_y}^2}{\sigma^2}), if \quad exp(-\frac{d_{v_x,v_y}^2}{\sigma^2}) \geq \varepsilon, \\ \quad 0 \qquad ,otherwise \end{cases} \tag{28}$$

where d_{v_x,v_y} is the road network distance from sensor v_x to v_y, σ and ε are thresholds to control the distribution and sparsity of matrix A. We set $\varepsilon = 0.1$ and $\sigma = 10$ for default.

5.2 Baselines

We compare our BMHA-GNN with the following models:

- HA: Historical Average, We use the average of historical data in the same time period as the prediction result.
- VAR (J.D., 1994): A classic time series prediction model, which utilize vector auto-regression method.
- FC-LSTM: Fully connected long short term memory network (Sutskever et al., 2014) for predictions of time series.
- DCRNN: Diffusion Convolutional Recurrent Neural Network (Li et al., 2018), which captures both spatial and temporal dependencies among time series using diffusion convolution and the sequence to sequence learning framework together with scheduled sampling.
- ST-GCN: Spatial-Temporal Graph Convolution Network (Yu et al., 2017), which applies

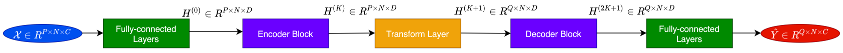

$\mathcal{X} \in R^{P\times N\times C}$
Fully-connected Layers
$H^{(0)} \in R^{P\times N\times D}$
Encoder Block
$H^{(K)} \in R^{P\times N\times D}$
Transform Layer
$H^{(K+1)} \in R^{Q\times N\times D}$
Decoder Block
$H^{(2K+1)} \in R^{Q\times N\times D}$
Fully-connected Layers
$\hat{Y} \in R^{Q\times N\times C}$

Figure 3: Summary of model pipeline and tensor dimensions.

Table 2: The performance of our model and baselines on different predicting intervals.

Dataset	Models	15min			30min			60min		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
METR-LA	HA	4.16	7.80	13.00%	4.16	7.80	13.00%	4.16	7.80	13.00%
	VAR	4.43	7.89	10.20%	5.42	9.14	12.70%	6.52	10.12	15.80%
	FC-LSTM	3.44	6.31	9.60%	3.78	7.23	10.89%	4.37	8.69	13.20%
	DCRNN	2.75	5.37	7.30%	3.15	6.44	8.80%	3.6	7.58	10.50%
	ST-GCN	2.88	5.74	7.60%	3.46	7.24	9.60%	4.59	9.40	12.70%
	Graph WaveNet	2.69	5.15	6.90%	3.07	6.22	8.40%	3.53	7.37	10.00%
	MRA-BGCN	2.67	5.12	6.80%	3.06	6.17	8.30%	3.49	7.30	10.00%
	GMAN	2.81	5.37	7.66%	3.10	6.30	8.48%	3.45	7.36	10.01%
	FC-GAGA	2.70	5.24	7.01%	3.04	6.19	8.31%	3.45	7.19	9.88%
	BMHA-GNN	**2.66**	**5.10**	**6.78%**	**3.04**	**6.16**	**8.30%**	**3.43**	**7.10**	**9.85%**
PEMS-BAY	HA	2.88	5.59	6.80%	2.88	5.59	6.80%	2.88	5.59	6.80%
	VAR	1.74	3.16	3.60%	2.32	4.25	5.00%	2.92	5.43	6.49%
	FC-LSTM	2.04	4.18	4.80%	2.20	4.54	5.20%	2.38	4.96	5.70%
	DCRNN	1.38	2.95	2.90%	1.74	3.97	3.90%	2.06	4.74	4.90%
	ST-GCN	1.36	2.95	2.90%	1.81	4.27	4.20%	2.48	5.68	5.80%
	Graph WaveNet	1.30	2.74	2.70%	1.63	3.70	3.70%	1.95	4.52	4.60%
	MRA-BGCN	1.29	2.72	2.90%	1.61	3.67	3.80%	1.91	4.46	4.60%
	GMAN	1.34	2.82	2.81%	1.62	3.72	3.63%	1.86	4.32	4.31%
	FC-GAGA	1.34	2.82	2.82%	1.66	3.75	3.71%	1.93	4.40	4.48%
	BMHA-GNN	**1.28**	**2.71**	**2.70%**	**1.60**	**3.67**	**3.60%**	**1.82**	**4.30**	**4.28%**

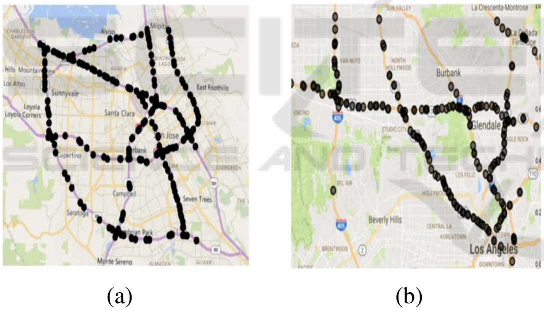


Figure 4: Sensor distribution of the PEMS-BAY and METR-LA dataset. (a) PEMS-BAY, (b) METR-LA.

purely convolutional structures to extract spatial-temporal features simultaneously.

- Graph WaveNet: Graph WaveNet for Deep Spatial-Temporal Graph Modeling (Wu et al., 2019), which constructs a self-adaptive adjacency matrix to capture the hidden spatial dependencies and proposes a new graph convolution with dilated casual convolution.
- MRA-BGCN: Multi-Range Attentive Bicomponent Graph Convolution Network (Chen et al., 2019), which proposes the bicomponent graph convolution to explicitly model the corrections of both nodes and edges.
- GMAN: A Graph Multi-Attention Network (Zheng et al., 2020), which proposes spatial-temporal attention mechanisms to model the dynamic spatial and non-linear temporal correlations.
- FC-GAGA: Fully Connected Gated Graph Architecture (Oreshkin et al., 2020), which uses hard graph gating mechanism and fully connected time-series forecasting architecture.

5.3 Experimental Results

5.3.1 Experiments Settings

Firstly, we recall the definition of our task, $f : R^{P\times N\times C} \rightarrow R^{Q\times N\times C}$ We are given the historical traffic data of the past hour and predict the traffic data of the next hour, i.e., $P = Q = 12$.

In our experiment, we set the number of BSTAtt Module K to 3, and the dimension of vertex D to 64. We set the number of multi-head L to 8 and the output dimension of each attention head d is 8. We set the Bilinear aggregator parameters $\alpha = 0.3$ for bilinear spatial attention, and $\alpha = 0.2$ for bilinear temporal attention. We set $\beta = 0.7$ to control the one-hop and two-hop neighbors. We set the max epoch to 500, if the validation loss does not decrease in the last 20 epochs, it will be terminated early. We use the *AdamOptimizer* (Kingma and Ba, 2014) to minimize loss, and the learning rate is set to 0.001.

In order to show our experimental results more objectively, we ran each experiment 10 times through different initialization seeds and took the average value to represent the final result. At the same time, the confidence p_value is 0, which objectively shows the superiority of our experimental results. We will also open source code in the future for everyone to reproduce.

5.3.2 Evaluate Metrics

We adopt the three most common traffic prediction indicators to evaluate our model: (1) Mean Absolute Error (MAE), (2) Root Mean Squared Error (RMSE), and (3) Mean Absolute Percentage Error (MAPE).

5.3.3 Performance Comparison

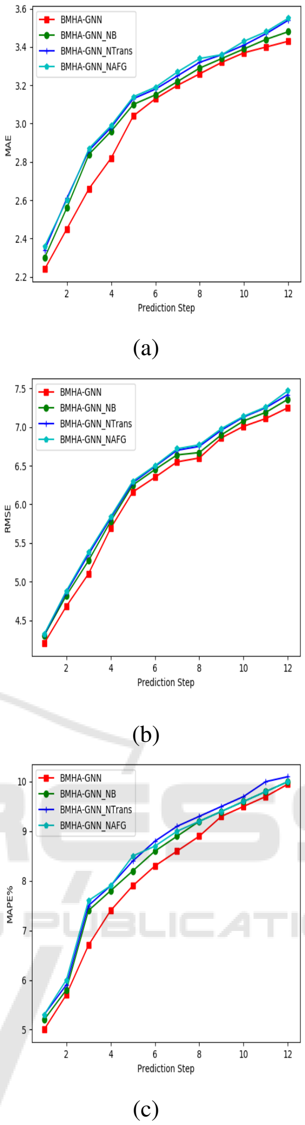
Table 2 shows the performance of our BMHA-GNN model and nine baseline models on two datasets, METR-LA, PEMS-BAY (Li et al., 2018). We divided the 1h prediction results into short time (15min), medium time (30min), and long time (1h). We can see from Table 2 that our model achieves the state-of-the-art in most scenarios.

Compare with these baseline models, we observe the following phenomena: (1) The performance of GNN-based models will be better than other models, because GNNs can better capture the dependency between spatial and temporal. (2) The use of bilinear aggregator in spatial embedding and temporal embedding can learn higher-order information and combination features. We believe that if two nodes appear at the same time, it will be a very strong signal for the current target node.

5.3.4 Ablation Study

To verify the effect of each module, we make several variants for BMHA-GNN, BMHA-GNN_NB (without Bilinear aggregator), BMHA-GNN_NTrans (without Transform Attention), BMHA-GNN_NAFG (without Attention Fusion Gate). As we all know, the dataset of METR-LA is more complicated and it is more difficult to estimate. Therefore the ablation study experiments will use this dataset.

Figure 5 shows the three indicators on the METR-LA dataset. From it, we can see that the performance of BMHA-GNN is better than BMHA-GNN_NB significantly, which proves that the bilinear aggregator we propose is effective.



(a)

(b)

(c)

Figure 5: Performance of each prediction step in METR-LA with Three variants. (a) The metrics of MAE, (b) The metrics of RMSE, and (c) The metrics of MAPE.

Table 3: Time-consuming of different models on PEMS-BAY.

Models	Computation-Time	
	Training(s/epoch)	Inference(s)
DCRNN	689.92	132.45
GMAN	245.87	15.34
Graph WaveNet	203.18	9.87
BMHA-GNN	359.42	19.22

5.3.5 Time Cost

We train and inference on a GPU machine, Tesla V100-SXM2-32GB. We list the time-consuming situation of several models, including training and inference process in Table 3. As shown in Table 3, DCRNN takes the highest computation time because it requires iterative calculation to generate 12 steps prediction score. Our model takes higher time than GMAN and Graph WaveNet. The reason is that we add the calculation logic of bilinear, and the parameters of the model have also been relatively increased. However, compared to the improvement of the model performance, we believe that this conversion is cost-effective.

6 CONCLUSION

We propose an end-to-end Bilinear Multi-Head Attention Graph Neural Network for Traffic Prediction, which not only utilize the linear weighted neighbor nodes to represent the target spatial and temporal node, but also use the bilinear aggregator in spatial and temporal representations. Extensive experiments are carried out on two real-word traffic datasets, and the results show that our proposed model achieves the state-of-the-art performance in most scenes. For future work, we will consider encoding high-order interactions among multiple neighbors to represent the target node and apply our model to other related applications.

ACKNOWLEDGEMENTS

This work was supported by the National Key R&D Program of China under Grant No.2018AAA0101204 and No.2018AAA0101200. Kai Han is the corresponding author.

REFERENCES

Atwood, J. and Towsley, D. (2015). Search-convolutional neural networks. volume abs/1511.02136.

Beutel, A., Covington, P., Jain, S., Xu, C., Li, J., Gatto, V., and Chi, E. H. (2018). Latent cross: Making use of context in recurrent recommender systems. In *WSDM*, pages pages 46–54.

Bruna, J., Zaremba, W., Szlam, A., and Lecun, Y. (2014). Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*.

Chen, W., Chen, L., Xie, Y., Cao, W., Gao, Y., and Feng, X. (2019). Multi-range attentive bicomponent graph convolutional network for traffic forecasting. volume abs/1911.12093.

Chun-Hsin Wu, Jan-Ming Ho, and Lee, D. T. (2004). Travel-time prediction with support vector regression. volume 5, pages 276–281.

Chung, J., Gülçehre, Ç., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. volume abs/1412.3555.

D., N., N., N., Y., G., , and H., K. (2005). Univariate short-term prediction of road travel times. In *In Proceedings of IEEE Intelligent Transportation Systems Conference*, pages 1074–1079.

Grover, A. and Leskovec, J. (2016). node2vec: Scalable feature learning for networks. volume abs/1607.00653.

J.D., H. (1994). Time series analysis. volume Volume 2, pages 690–696.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization.

Li, Y., Yu, R., Shahabi, C., and Liu, Y. (2018). Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations (ICLR '18)*.

Ma, X., Dai, Z., He, Z., and Wang, Y. (2017). Learning traffic as images: A deep convolution neural network for large-scale transportation network speed prediction. volume abs/1701.04245.

Nikovski, D., Nishiuma, N., Goto, Y., and Kumazawa, H. (2005). Univariate short-term prediction of road travel times. In *Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005.*, pages 1074–1079.

Oreshkin, B. N., Amini, A., Coyle, L., and Coates, M. J. (2020). Fc-gaga: Fully connected gated graph architecture for spatio-temporal traffic forecasting.

Q., Z., J., C., G., M., S., X., and C., P. (2020). Spatio-temporal graph structure learning for traffic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1177–1185.

Rendle, S. (2010). Factorization machines. In *2010 IEEE International Conference on Data Mining*, pages 995–1000.

S., G., Y., L., N., F., C., S., and H., W. (2019). Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. pages 922–929.

Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., and Vandergheynst, P. (2012). Signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular data domains. volume abs/1211.0053.

S.I.J., C. and C.M., K. (2003). Dynamic travel time prediction with real-time and historic data. In *Journal of Transportation Engineering 129(6)*, pages 608–616.

Song, C., Lin, Y., Guo, S., and Wan, H. (2020). Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. volume 34, pages 914–921.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, u., and Polosukhin, I. (2017). Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Wu, Z., Pan, S., Long, G., Jiang, J., and Zhang, C. (2019). Graph wavenet for deep spatial-temporal graph modeling. volume abs/1906.00121.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2019). How powerful are graph neural networks? In *International Conference on Learning Representations*.

Xuan, S., Hiroshi, K., and Ryosuke, S. (2016). Deeptransport: Prediction and simulation of human mobility and transportation mode at a citywide level. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 2618–2624. AAAI Press.

Yao, H., Tang, X., Wei, H., Zheng, G., Yu, Y., and Li, Z. (2018a). Modeling spatial-temporal dynamics for traffic prediction. volume abs/1803.01254.

Yao, H., Wu, F., Ke, J., Tang, X., Jia, Y., Lu, S., Gong, P., Ye, J., and Li, Z. (2018b). Deep multi-view spatial-temporal network for taxi demand prediction. volume abs/1802.08714.

Yu, B., Yin, H., and Zhu, Z. (2017). Spatio-temporal graph convolutional neural network: A deep learning framework for traffic forecasting. volume abs/1709.04875.

Zheng, C., Fan, X., Wang, C., and Qi, J. (2020). Gman: A graph multi-attention network for traffic prediction. In *AAAI*, pages 1234–1241.

Zheng, Z. and Su, D. (2014). Short-term traffic volume forecasting: A k-nearest neighbor approach enhanced by constrained linearly sewing principle component algorithm. volume 43, pages 143–157. Special Issue on Short-term Traffic Flow Forecasting.

Zhu, H., Feng, F., He, X., Wang, X., Li, Y., Zheng, K., and Zhang, Y. (2020). Bilinear graph neural network with node interactions.