Bayesian Graph Convolutional Network for Traffic Prediction

Jun Fu[a], Wei Zhou[a], Zhibo Chen[a,*]

[a] *CAS Key Laboratory of Technology in Geo-spatial Information Processing and Application System, University of Science and Technology of China, Hefei 230027, China*

Abstract

Recently, adaptive graph convolutional network based traffic prediction methods, learning a latent graph structure from traffic data via various attention-based mechanisms, have achieved impressive performance. However, they are still limited to finding a better description of spatial relationships between traffic conditions due to: (1) ignoring the prior of the observed road network topology; (2) neglecting the presence of negative spatial relationships; and (3) lacking investigation on the uncertainty of the graph structure. In this paper, we propose a Bayesian Graph Convolutional Network (BGCN) framework to alleviate these issues. Under this framework, the graph structure is viewed as a random realization from a parametric generative model, and its posterior is inferred using the observed topology of the road network and traffic data. Specifically, the parametric generative model is comprised of two parts: (1) a constant adjacency matrix that discovers potential spatial relationships from the observed physical connections between roads using a Bayesian approach; (2) a learnable adjacency matrix that learns globally shared spatial correlations from traffic data in an end-to-end fashion and can model negative spatial correlations. The posterior of the graph structure is then approximated by performing Monte Carlo dropout on the parametric graph structure. We verify the effectiveness of our method on five real-world datasets, and the experimental results demonstrate

*Corresponding author
Email address: chenzhibo@ustc.edu.cn (Zhibo Chen)

that BGCN attains superior performance compared with state-of-the-art methods. The source code is available at https://github.com/JunFu1995/BGCN.git.
Keywords: traffic prediction, Bayesian, generative model

1. Introduction

Traffic congestion is a growing drain on the economy with the acceleration of urbanization. For example, the cost of traffic congestion in America reached $124 billion in 2014 and will rise to $186 billion in 2030, according to a report by Forbes [1]. One promising way to mitigate urban traffic congestion is to introduce Intelligent Transportation System (ITS). As an indispensable part of ITS, traffic prediction aims to predict future traffic conditions of roads based on historical measurements. Accurate traffic prediction plays a vital role in traffic scheduling and management.

Traffic forecasting is a challenging task due to the complex temporal dependencies (i.e, the traffic condition at a road is related to its historical observations) and spatial dependencies (i.e, the traffic conditions of adjacent roads influence each other). Traditional methods employ linear time series models [2, 3] for traffic forecasting. These methods fail to capture nonlinear temporal correlations and ignore the presence of spatial dependencies. Recently, a broad of learning-based traffic predictors have been developed. They typically model temporal dependencies using recurrent neural networks (RNNs) or temporal convolution modules. As for spatial dependencies, they commonly deploy GCNs [4] because of the graph-structured road network. Despite the impressive results achieved, these graph-based methods are still limited to achieving more accurate predictions. This is mainly because the graph structure employed in GCNs is heuristically defined (i.e., roads are nodes, physical connections between roads are edges, and edge weights rely on the Euclidean distance between two roads), which may miss some important spatial correlations for traffic prediction (as shown in Fig. 1).

More recently, researchers turn to adaptive graph-based methods and focus

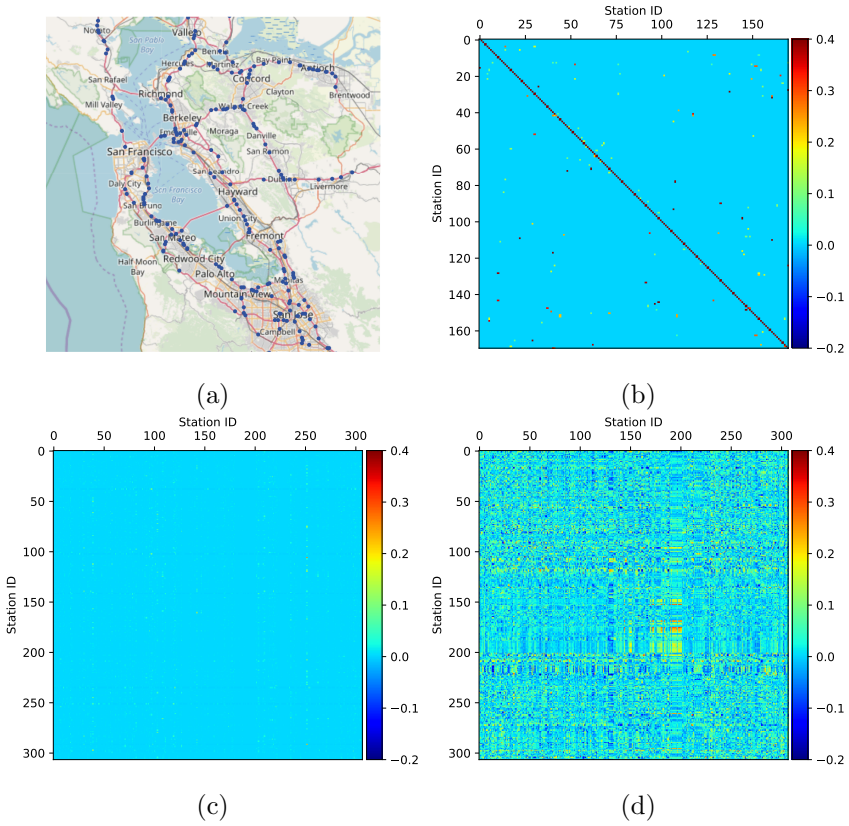


Figure 1: (a) PeMS sensor network in District 4 of California (b) Heuristically defined graph structure. (c) Learned graph structure by an attention-based method [5]. (d) Learned graph structure by our proposed BGCN. Compared to the attention-based method, BGCN discovers more edges and includes both positive and negative spatial relationships.

on designing various attention mechanisms to learn the latent graph structure. However, existing adaptive graph-based methods suffer from the following three limitations. First, they learn the latent graph structure from scratch, ignoring the prior of the observed road network topology. Specifically, they do not use observed connections between roads to infer potential ones. Second, the attention-based graph learning methods tend to depress the negative spatial relationships (i.e., the correlation strength between two roads is negative) due to the $SoftMax$ operation. However, some negative spatial relationships may be useful for traffic prediction [6]. Third, introducing uncertainty into the graph

structure (e.g., DropEdge [7]) has proven its effectiveness in improving the generalization ability of GCNs. Unfortunately, GCN-based traffic predictors regard the graph structure as a deterministic variable, lacking investigation on the uncertainty of the graph structure. Notably, uncertainty in traffic prediction means that the graph structure is not fixed, but randomly drawn from a parametric distribution.

The above concerns motivate us to propose a Bayesian Graph Convolutional Network (BGCN). BGCN considers the graph structure as a random sample drawn from a parametric generative model and infers the posterior of the graph structure based on the observed topology of the road network and traffic data. Specifically, BGCN first decomposes the graph structure into two parts: (1) a precomputed constant adjacency matrix that infers potential spatial relationships based on physical connections between roads using a Bayesian approach. (2) a learnable adjacency matrix that learns a spatial pattern shared in all time steps from traffic data in an end-to-end manner and can represent negative spatial correlations. Then, BGCN approximates the posterior of the graph structure by performing Monte Carlo dropout on the parametric graph structure.

The main contributions of our work lie in three folds:

- This work is the first attempt to apply Bayesian graph convolutional networks in traffic prediction.
- This work presents an efficient approach to infer the posterior of the graph structure based on traffic data and the observed topology of the road network, which introduces little extra cost in computation and parameters.
- This work validates the effectiveness of the proposed BGCN on five real-world traffic datasets, and the experimental results show that BGCN surpasses state-of-the-art methods by a noticeable margin.

The rest of the paper is organized as follows. Section 2 reviews research works related to traffic prediction. Sections 3 and 4 introduce details of our method and experimental results, respectively. Section 5 concludes this paper

and points out some future directions.

2. Related Work

2.1. Traffic prediction

As a key part of ITS, traffic prediction has been studied for decades. Traditional methods typically employ statistical models to predict future traffic conditions of roads. The common statistical models include historical average model (HA) [8], Kalman filtering model [9], Vector Auto-Regression (VAR) model [3], Auto-Regressive Integrated Moving Average (ARIMA) model [2], and variants of ARIMA [10, 11]. However, these traditional methods rely on the stationary assumption and forecast the traffic state at each road separately. Therefore, they are not well-generalized in practical applications since real-world traffic conditions usually are nonlinear and involve complex spatial-temporal correlations.

Benefiting from the rapid development of computational power and the growth of traffic data volume, data-driven traffic prediction methods have received considerable attention. At the early stage, researchers use shallow machine learning approaches, such as Random forest [12], Support Vector Regression (SVR) model [13], Bayesian model [14], and K-nearest neighbors methods [15, 16, 17] to capture non-linear regularity from traffic data. However, these shallow machine learning methods heavily depend on hand-crafted features. As a result, researchers shift attention to deep learning-based traffic prediction methods, which relieve the burden of feature engineering. For example, feed-forward neural network (FNN) [18], stacked auto-encoder (SAE) [19], and Deep Belief Network (DBN) [20] use multilayer perceptron to learn meaningful feature representations. Considering traffic data is time-series data, RNN (Recurrent Neural Network) and its variants [21, 22] are widely used in traffic prediction [23, 24, 25, 26, 27]. Despite the impressive capability in modeling temporal dynamics, RNN-based traffic predictors are still limited due to ignoring the spatial correlations between roads.

To address the above concern, researchers resort to convolutional neural networks (CNNs). Wu et al. [28] combine 1D CNN and LSTM for modeling spatial-temporal dependencies between traffic flows. Yu et al. [29] treat traffic states as a series of 2D images and design a spatio-temporal recurrent convolutional network. Wang et al. [30] leverage CNN to capture nearby road states to boost the prediction performance. Zhang et al. [31] build a very deep CNN called ST-ResNet for citywide crowd flow prediction. However, these CNN-based methods cannot fully characterize the spatial correlations especially in the road network with a complex topological structure. This is mainly because CNNs prefer to deal with grid-like data rather than graph-structured data. To address this issue, Zhang et al. [32] and Du et al. [33] extract spatial features of traffic flows from the neighbor road links using deformable convolution instead conventional CNNs.

Recently, graph convolutional networks (GCNs), which are suitable for dealing with non-grid data, have attracted a lot of attention. Zhao et al. [34] propose a temporal graph convolutional network (TGCN), which replaces convolutional operations in RNNs with GCNs. Li et al. [35] regard the traffic flow as a diffusion process on a directed graph, and design a Diffusion Convolutional Recurrent Neural Network (DCRNN) for traffic forecasting. Yu et al. [36] introduce a Spatio-Temporal Graph Convolutional Network (STGCN), which leverages GCNs and gated CNNs for extracting spatial and temporal features respectively. Lv et al. [37] propose a Temporal Multi-Graph Convolutional Network (TMGCN), which manually constructs four types of graphs for handling complex spatial dependence. Peng et al. [38] propose spatial temporal incidence dynamic graph neural networks for traffic flow forecasting, where the graph is built from precomputed statistics of history traffic flows. Generally, these GCN-based methods require accurate graph construction. Nevertheless, manually constructing a graph is a laborious and error-prone process.

Therefore, automatically learning a latent graph structure for traffic prediction has become a new trend. Wu et al. [5] explicitly learn representations of roads, and generate a static graph based on the similarity of two roads' repre-

sentations. Huang et al. [39] propose a Long Short-term Graph Convolutional Network (LSGCN) [39], which develops a graph attention mechanism. Guo et al. [40] employ an attention mechanism [41] to learn a data-dependent graph. Similar attention-based graph structure learning is also investigated in STSGCN [42], AGCRN [43], and GATCN [44]. Zhang et al. [45] propose a novel evolution temporal graph convolutional network to discover multiple spatial dependence. Li et al. [45] propose a novel multi-sensor data correlation GCN to model different period traffic patterns, and capture the dynamic spatio-temporal correlation. Yin et al. [46] develop a Multi-Stage Attention Spatial– Temporal Graph Network, which investigates dynamic temporal dependencies. Chen et al. [47] and Zheng et al. [48] use attention-based mechanism to learn graph structures that change over time steps. Unfortunately, in learning the underlying graph structure, these methods ignore the prior of the road network topology and lack investigation on the uncertainty of the graph structure.

Compared to the three works [32, 33, 38], the proposed BGCN is mainly different in two aspects. First, BGCN can mine spatial correlations between roads in an end-to-end fashion, while these three works are in heuristic manners. Specifically, [32] and [33] describe spatial correlations between roads only based on the physical road topology, while [38] is based on precomputed statistics of history traffic flows. Second, BGCN considers the uncertainty of the graph structure, which is not taken into consideration in these three works.

Compared to the two works [47, 48], the proposed BGCN is mainly different in four aspects. First, the two works adopt attention-based graph structure learning, while BGCN uses residual graph learning (i.e., the coarse-to-fine manner). Hence, BGCN can naturally represent negative spatial correlations between roads. Second, the graph structure in the two works adaptively changes over time steps, while the one in the BGCN is shared in all time steps. To some content, the graph structure in BGCN can be combined with the two works. Third, the two works directly use the observed graph structure, while BGCN uses the observed graph structure to dig out potential correlations between roads. Forth, BGCN considers the uncertainty of the graph structure,

which is not taken into consideration in the two works.

2.2. Graph Convolutional Network

GCNs have achieved remarkable success in a broad of applications, such as semi-supervised learning [49], action recognition [50], and quality assessment [51, 52]. Recently, researchers start to pay attention to the uncertainty of the graph structure. For instance, Rong et al. [7] develop DropEdge, which randomly drops edges in the pre-defined graph structure for enhancing the generalization ability of GCNs. You et al. [53] design various graph augmentation techniques for better graph presentation learning. Besides introducing uncertainty into the graph structure, more recent works called Bayesian Graph Convolutional Networks (BGCNs) [54, 55] derive the underlying connections between nodes from training data and the observed graph topology. However, BGCNs are designed for semi-supervised node classification and have not been investigated in the more complicated spatial-temporal time series prediction.

3. Method

3.1. Problem Definition

Traffic Network We assume the observed road network as a weighted directed graph $\mathcal{G}_{obs} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$. Here, $\mathcal{V}$ is a set of $N = |\mathcal{V}|$ vertices, where vertex v_i corresponds to the i-th road; $\mathcal{E}$ is a set of edges representing the connectivity between vertices; and $\mathcal{A} \in \mathbb{R}^{N \times N}$ is the weighted adjacency matrix, where $\mathcal{A}^{i,j}$ records the correlation between vertex v_i and v_j. The traffic condition at time step t is regarded as a graph signal $X_t \in \mathbb{R}^{N \times D}$ on the graph $\mathcal{G}_{obs}$, where D represents the number of traffic characteristics (e.g., traffic flow, traffic speed, etc.).

Problem Given historical T graph signals $\mathcal{X} = \{X_1, ..., X_T\} \in \mathbb{R}^{N \times T \times D}$, we aim to forecast τ future graph signals $\mathcal{Y} = \{X_{T+1}, ..., X_{T+\tau}\} \in \mathbb{R}^{N \times \tau \times D}$ using a mapping model $\mathcal{F}$:

$$\mathcal{Z} = \{\overline{X}_{T+1}, ..., \overline{X}_{T+\tau}\} = \mathcal{F}_\theta(X_1, ..., X_T; \mathcal{G}_{obs}), \tag{1}$$

where θ represents all the learnable parameters in the model $\mathcal{F}$.

3.2. Motivation

As aforementioned, the spatial correlation between traffic conditions is a key factor in traffic forecasting. Considering that the road network is naturally structured as a graph, existing works prefer to extract spatial features using a computation-friendly spectral graph convolution [4]:

$$f(X, \mathcal{A}) = \tilde{\mathcal{A}} X W, \tag{2}$$

where $\tilde{\mathcal{A}} \in \mathbb{R}^{N \times N}$ is the normalized adjacency matrix with self-loops, $X \in \mathbb{R}^{N \times D}$ denotes the traffic condition at a certain time step, and $W \in \mathbb{R}^{D \times M}$ is the parameter matrix. To capture more meaningful graph representation, recent studies propose adaptive graph convolution, which learns a self-adaptive adjacency matrix in an attention-based mechanism:

$$\begin{aligned} f(X, \tilde{\mathcal{A}}_{adp}) &= \tilde{\mathcal{A}}_{adp} X W, \\ \tilde{\mathcal{A}}_{adp} &= SoftMax(\sigma(E_1 E_2^T)), \end{aligned} \tag{3}$$

where $\tilde{\mathcal{A}}_{adp} \in \mathbb{R}_{+}^{N \times N}$ is the generated adjacency matrix. σ is the activation function, $E_1 \in R^{N \times\ c}$ and $E_2 \in R^{N \times\ c}$ are used as query and key respectively. It is worth noting that E_1 and E_2 can be generated from the input traffic flows on roads or two learnable road embeddings.

However, such an adaptive graph convolution suffer from three limitations in the graph structure learning. First, it learns adjacency matrix from scratch, neglecting the prior of the observed topology of the road network. This may not be a good choice as the physical connection between roads is a vital clue for learning potential spatial correlations. Second, it is limited in modeling negative spatial correlations between traffic conditions because of the $SoftMax$ operation. Third, $\tilde{\mathcal{A}}_{adp}$ is still a deterministic variable, lacking investigation on the uncertainty of the graph structure.

3.3. Bayesian Graph Convolutional Network

We address the above concerns with a Bayesian approach. Specifically, we regard the graph structure $\mathcal{A}$ as a sample drawn from a parametric generative model, and aim to infer the posterior predictive distribution :

$$p(\mathcal{Z}|\mathcal{X},\mathcal{Y},\mathcal{G}_{obs},W)=\int p(\mathcal{Z}|\mathcal{X},W,\mathcal{G})p(\mathcal{G}|\mathcal{X},\mathcal{Y},\mathcal{G}_{obs})d\mathcal{G}, \tag{4}$$

where the item $p(\mathcal{Z}|\mathcal{X},W,\mathcal{G})$ is modeled by a graph-based network, and the item $p(\mathcal{G}|\mathcal{X},\mathcal{Y},\mathcal{G}_{obs})$ aims to infer the posterior of the graph structure using the train data $\{\mathcal{X},\mathcal{Y}\}$ and the observed topology of the road network $\mathcal{G}_{obs}$. Notably, unlike conventional BGCNs, we only perform the posterior inference of the graph structure. This is mainly because we find introducing uncertainty into the weights of GCNs has little impact on traffic prediction in our experiment.

In this paper, we perform the posterior inference of the graph structure in a coarse-to-fine fashion:

$$\begin{aligned}p(\mathcal{Z}|\mathcal{X},\mathcal{Y},\mathcal{G}_{obs},W)=\int p(\mathcal{Z}|\mathcal{X},W,\mathcal{G})p(\mathcal{G}|g,\mathcal{X},\mathcal{Y})\\ p(g|\mathcal{G}_{obs})d\mathcal{G}dg,\end{aligned} \tag{5}$$

where we aim to first learn a coarse graph structure g from the topology of the road network $\mathcal{G}_{obs}$, and then learn a fine-level graph structure $\mathcal{G}$ based on g and the paired traffic data $\{\mathcal{X},\mathcal{Y}\}$. Since there is no closed-form solution for the integral in Eq. 5, we introduce a Monte Carlo approximation:

$$p(\mathcal{Z}|\mathcal{Y},\mathcal{X},\mathcal{G}_{obs},W)\approx\frac{1}{SC}\sum_{s=1}^{S}\sum_{c=1}^{C}p(\mathcal{Z}|W,\mathcal{G}_{s,c},\mathcal{X}), \tag{6}$$

where C weights g_c sampled from $p(g|\mathcal{G}_{obs})$, S weights samples $\mathcal{G}_{s,c}$ drawn from $p(\mathcal{G}|g_c,\mathcal{X},\mathcal{Y})$. Consider sampling graph from $p(g|\mathcal{G}_{obs})$ is time-consuming [55], we replace the integral over g with a MAP process:

$$\overline{g}=\arg\max_{g}\quad p(g|\mathcal{G}_{obs}). \tag{7}$$

As described in the work [55], solving Eq. 7 is equivalent to learning a $N\times N$ symmetric adjacency matrix of g:

$$\mathcal{A}_{\overline{g}}=\underset{\substack{\mathcal{A}_g\in\mathbf{R}_{+}^{N\times N},\\ \mathcal{A}_g=\mathcal{A}_g^T}}{\arg\min}\|\mathcal{A}_g\odot Z\|_1-\alpha\mathbf{1}^T\log(\mathcal{A}_g\mathbf{1})+\beta\|\mathcal{A}_g\|^2, \tag{8}$$

where α and β control the scale and sparsity of $A_{\overline{g}}$. Here, $Z \in \mathbb{R}^{N \times N}$ denotes the pairwise distance of roads in the embedding space:

$$Z_{p,q} = \|e_p - e_q\|^2, \tag{9}$$

where e_p and e_q are the embedding vector of the p-th and q-th road. These embedding vectors of nodes are learned based on the physical connections between roads using Graph Variational Auto-Encoder algorithm [56]. After obtaining Z, we solve the Eq. 8 via the prevalent optimization-based method [57]. Then, we model the item $p(\mathcal{G}|g, \mathcal{X}, \mathcal{Y})$ using a Monte Carlo approximation:

$$p(\mathcal{G}|g, \mathcal{X}, \mathcal{Y}) = Dropout(\tilde{\mathcal{A}}_{\overline{g}} + \phi), \tag{10}$$

where $\tilde{\mathcal{A}}_{\overline{g}}$ is the normalized adjacency matrix with self-loops, and $\phi \in \mathbb{R}^{N \times N}$ is a learnable adjacency matrix. We can notice that the graph structure comprises two parts: (1) a constant adjacency matrix $\tilde{\mathcal{A}}_{\overline{g}}$ which offers some prior knowledge on spatial correlations; (2) a learnable adjacency matrix ϕ which can include both positive and negative spatial correlations. Moreover, the graph structure is random due to the operation of *Dropout*. Finally, Eq. 6 is simplified as follows:

$$p(\mathcal{Z}|\mathcal{X}, \mathcal{Y}, \mathcal{G}_{obs}, W) \approx \frac{1}{S} \sum_{s=1}^{S} p(\mathcal{Z}|\mathcal{X}, W, \mathcal{G}_s), \tag{11}$$

where S weights samples $\mathcal{G}_s$ drawn from $A_{\overline{g}} + \phi$ via dropout [58]. The graph convolution in the item $p(\mathcal{Z}|\mathcal{X}, W, \mathcal{G}_s)$ is performed as follows:

$$f(X, \tilde{\mathcal{A}}_{\overline{g}}, \phi) = Dropout(\tilde{\mathcal{A}}_{\overline{g}} + \phi)XW. \tag{12}$$

3.4. Network Architecture

We adopt the architecture of Graph WaveNet. As shown in Fig. 2, It is composed of stacked spatial-temporal layers and an output layer. A spatial-temporal layer consists of GCN and a gated temporal convolution layer (Gated TCN) that contains two parallel temporal convolution layers (TCN-a and TCN-b). We replace GCN in all spatial-temporal layers with our proposed BGCN.

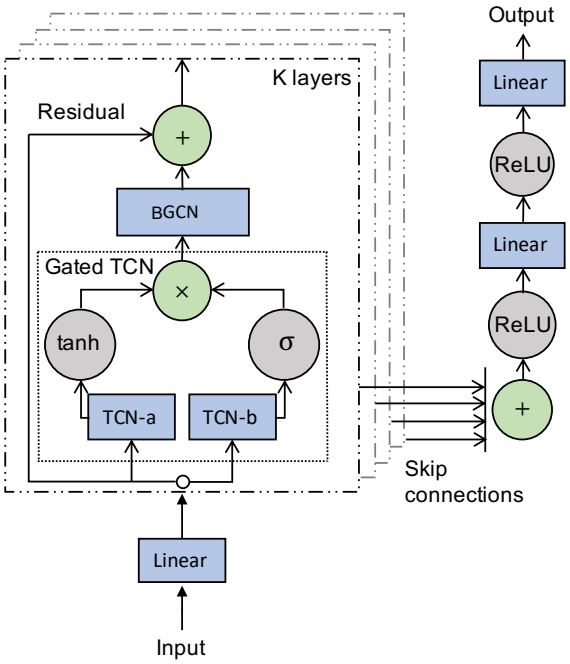


Figure 2: Framework of BGCN-applied Graph WaveNet. TCN-a and TCN-b are two types of temporal convolutional networks. Linear represents fully-connected layers. σ denotes the sigmoid activation function.

Algorithm 1 Training methodology of BGCN

Input: $\mathcal{G}_{obs}$, training datasets $\mathcal{D}$

Output: W, $\mathcal{A}_{\overline{g}}$, ϕ

Randomly initialize W

Initial ϕ: $\phi \leftarrow 1e^{-6}$

Obtain $\mathcal{A}_{\overline{g}}$ via solving Eq. 8

for $epoch = 1$ **to** $MaxEpoch$ **do**

  Sample the graph structure $\mathcal{G}$ from $\mathcal{A}_{\overline{g}} + \phi$ via dropout

  Sample a batch of data $\{\mathcal{X}, \mathcal{Y}\}$ from D

  Obtain the predicted result $\mathcal{Z}$

  Optimize W and ϕ by minimizing Eq. 13.

end for

We choose to use mean absolute error (MAE) as the training objective, which is formulated as follows:

$$L = \frac{1}{\tau N D} \sum_{i=1}^{\tau} \sum_{j=1}^{N} \sum_{k=1}^{D} \| \overline{X}_{T+i}^{jk} - X_{T+i}^{jk} \|, \tag{13}$$

where $\overline{X}_{T+i}$ and X_{T+i} are the predicted results and the ground truth at time step $T + i$. The whole training algorithm is described in Algorithm 1.

3.5. Discussion

In this paper, we believe that directly transferring BGCN [55] to the traffic prediction task is not optimal. The reason is as follows. The graph structure derived by the original BGCN is of two main characteristics: symmetric and non-negative. However, in the traffic scene, the mutual influence of two roads typically is not equal, and the influence can be positive and negative [59]. In addition, BGCN cannot learn the latent spatial graph structure in an end-to-end fashion.

To address above issues, we develop a differentiable pipeline, which approximates the posterior of the graph structure by applying dropout on a parametric graph. Specifically, considering the physical connection between roads is a strong prior, we decompose the parametric graph into two parts: a constant adjacency matrix and a learnable adjacency matrix. The constant adjacency matrix is designed for mining underlying connections from the observed graph. To achieve this goal, we can reuse the method in the original BGCN. The learnable adjacency matrix is designed for mining possible dependence of the graph structure on traffic data. Considering the graph structure is typically asymmetric and has arbitrary values, we do not apply any constraints on the learnable adjacency matrix.

4. Experiments

4.1. Datasets

To verify the effectiveness of the proposed method, we conduct experiments on five real-world traffic datasets: PeMS3, PeMS4, PEMS7, PeMS8, and PeMS-

Bay. These datasets are provided by Caltrans Performance Measure System (PeMS), which records the highway traffic in California every 30 seconds. More details for the datasets are presented in Table 1.

PeMS3: It refers to the traffic data collected by 358 loop detectors in District 3 of California from September 1st to November 30th in 2018.

PeMS4: It refers to the traffic data collected by 307 loop detectors in the San Francisco Bay Area from January 1st to February 28th in 2018.

PeMS7: It refers to the traffic data collected by 883 loop detectors in District 7 of California from May 1st to August 31th in 2017.

PeMS8: It refers to the traffic data gathered by 107 loop detectors in the San Bernardino Area from July 1st to August 31th in 2016.

PeMS-Bay: It refers to the traffic data gathered by 325 loop detectors in the Bay Area from January 1st to May 31th in 2017.

Table 1: Details for the datasets.

Dataset	PeMS3	PeMS4	PeMS7	PeMS8	PeMS-Bay
Sensors	358	307	883	170	325
Time steps	26208	16992	28224	17856	52116
Mean	181.40	207.25	309.59	229.99	62.75
STD	144.41	156.49	189.49	145.61	9.37
Time interval	5 minutes				
Daily range	00:00-24:00				

4.2. Data preprocessing

We adopt the following strategies to preprocess the traffic data, which is consistent with previous studies [42, 43]. We aggregate the traffic data in a 5-minute interval. As a result, every loop detector contains 288 traffic data points per day. We split all the traffic datasets into training sets, validation sets, and test sets in a ratio of 6:2:2. We discard the missing values and use the Z-score method to normalize traffic data.

Regarding the observed adjacency matrix $\mathcal{A}_{obs}$, we construct it using the same method as DCRNN [35]:

$$\mathcal{A}_{obs}^{i,j} = \begin{cases} \exp(-\frac{d_{i,j}^2}{\xi^2}), i \neq j \text{ and } \exp(-\frac{d_{i,j}^2}{\xi^2}) \geq \epsilon, \\ \\ 0, \text{otherwise} \end{cases} \tag{14}$$

where $\mathcal{A}_{obs}^{i,j}$ is the edge weight between the i-th road and the j-th road, $d_{i,j}$ denotes the distance from the i-th road to the j-th road, ξ is the standard deviation of distances, ϵ is a threshold and set as 0.1.

4.3. Experimental Settings

Implement Details We implement BGCN based on the popular deep learning framework PyTorch [60], and conduct all experiments on a Linux server with one NVIDIA 1080Ti GPU card. The parameter setting of our framework keeps the same with Graph WaveNet. All the traffic datasets share the following training settings. Our goal is to predict traffic conditions in the next hour based on the measurements of the past one hour. We set the Monte Carlo dropout probability to 0.5. For the sake of reproducibility, the learnable matrix *phi* is initialized by the product of the constant adjacent matrix $\tilde{\mathcal{A}}_{\overline{g}}$ and 1e-6. Benefiting from the optimization-based approach [57], we use one hyperparameter, i.e., the number of edges per node, to control the sparsity of the constant adjacency matrix, and heuristically set it to 20. We optimize all trainable variables with the Adam [61] optimizer for 100 epochs. The learning rate is initialized as 0.001 and decreases to 0.0001 at the 50th epoch. During training, 64 pairs are randomly generated from the training dataset per iteration.

Baselines To evaluate the overall performance of our work, we compare BGCN with the following baselines:

- Historical Average (HA) [8]. It considers the traffic conditions of each road as a seasonal process and uses the average of previous seasons as the prediction. The duration of a season is set to a week.

- Vector Auto-Regression (VAR) [3]. It is a linear time series model, which can capture spatial correlations between traffic conditions. We implement it based on *statsmodel* python package. The number of lags for PeMS4, PeMS8, and PeMS-Bay is set to 8, 3, and 3, respectively. We do not report the performance on the PeMS3 and PeMS7 dataset due to the poor prediction results of VAR.

- GRU-ED. It adopts an encoder-decoder framework, where both encoder and decoder consist of a stacked GRU. The hidden size of GRU is set to 128.

- DCRNN [35]. Similar to GRU-ED, it also uses an encoder-decoder architecture, but equips both the encoder and decoder with diffusion convolution recursive layers for capturing spatial-temporal features.

- STGCN [36]. It extracts spatial-temporal correlations using spatial-temporal graph convolutional networks, which is a combination of temporal gated CNNs and spatial GCNs. Different from the original STGCN, we implement the output layer to generate prediction for all horizons at one time instead of one horizon per time.

- Graph WaveNet [5]. It combines graph convolution with dilated casual convolution to capture spatial-temporal dependecies. Moreover, it learns a latent graph structure using an attention-based method.

- STSGCN [42]. It proposes a spatial-temporal synchronous graph convolutional network to capture localized spatial-temporal correlations.

- AGCRN [43]. It enhances GCNs using a node adaptive parameter learning module and a data adaptive graph generation module.

We reuse the results of DCRNN and STSGCN reported in the previous work [42]. As for AGCRN and Graph WaveNet, we directly reuse the released codes without any modifications. The best parameters for all deep learning models are selected through a parameter-tuning process on the validation set.

Table 2: Performance comparison of different approaches on five datasets. "*" means reported results in the STSGCN [42]. GWN denots the Graph WaveNet [5].

Dataset	Metric	HA	VAR	GRU-ED	DCRNN*	STGCN	GWN	STSGCN*	AGCRN	BGCN
PeMS3	MAE	24.96	-	19.90	18.03	16.29	14.66	17.33	15.70	**14.35**
	RMSE	46.07	-	32.68	30.06	27.73	25.32	28.65	27.71	**25.28**
	MAPE (%)	25.84	-	18.83	18.09	17.80	15.29	16.58	14.82	**14.47**
PeMS4	MAE	24.56	22.82	25.63	24.48	21.09	19.23	21.09	19.86	**18.82**
	RMSE	39.91	35.26	39.84	37.86	33.08	30.71	33.45	32.00	**30.34**
	MAPE (%)	16.56	16.35	16.41	16.75	14.57	13.22	13.85	12.90	**12.87**
PeMS7	MAE	28.49	-	27.20	24.78	22.63	20.77	24.12	21.81	**20.09**
	RMSE	52.59	-	43.13	37.88	35.50	33.39	38.76	34.97	**32.86**
	MAPE (%)	11.98	-	11.56	11.33	10.01	8.91	10.16	9.18	**8.45**
PeMS8	MAE	21.23	19.87	20.10	17.83	16.98	15.43	17.04	16.29	**14.65**
	RMSE	36.72	29.29	31.68	27.68	26.58	24.19	26.62	25.66	**23.43**
	MAPE (%)	13.75	13.04	12.33	11.42	11.58	10.25	10.89	10.32	**9.42**
PeMS-Bay	MAE	2.88	2.24	1.96	-	1.77	1.65	-	1.71	**1.61**
	RMSE	5.59	3.97	4.69	-	3.87	3.66	-	3.88	**3.63**
	MAPE (%)	6.77	4.83	4.47	-	4.05	**3.65**	-	3.88	3.71

Evaluation Metrics We use three widely-used evaluation metrics, i.e., Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE), to measure the performance of predictive models.

4.4. Overall Comparison

Table 2 presents the results, i.e., the averaged MAE, RMSE, and MAPE over 12 prediction horizons, of BGCN and eight representative baselines on five datasets. According to this table, we draw the following conclusions.

- Traditional methods (i.e., HA and VAR) achieve competitive performance in the PeMS-Bay dataset with a small standard deviation, but poor performance in the datasets with large standard deviations (e.g., PeMS3 and PeM8). This is mainly because traditional methods follow the stationary assumption, which is easily violated in datasets with large standard deviations.

- Compared to traditional approaches, GRU-ED typically achieves better performance. This verifies the effectiveness of RNNs in capturing non-linear temporal regularity from traffic data.

- Compared to GRU-ED, GCN-based methods show obvious advantages in terms of MAE, RMSE, and MAPE. This reveals that the spatial-temporal dependence is important for accurate traffic prediction.

- Compared to STGCN using heuristically defined graph structure, the methods that learn latent graph structures from traffic data, including Graph WaveNet, AGCRN, and BGCN, significantly improve the prediction performance. This indicates that the road-network-topology-based graph structure is not the optimal description of the spatial relationships between traffic conditions.

- Compared to Graph WaveNet and AGCRN, BGCN achieves better performance in almost all evaluation metrics. This verifies the superiority of BGCN in graph structure modeling.

Fig. 3 further shows the prediction performance at each horizon on the PeMSD4 and PeMS8 datasets. For clear visualization, we only present three methods (i.e., AGCRN, Graph WaveNet, and BGCN). From this figure, we have the following findings.

- With the increase of the prediction interval, the prediction performance of three methods significantly drops. This indicates that long-term traffic prediction faces more challenges than short-term one.

- Compared to Graph WaveNet, BGCN achieves better performance at all prediction timestamps. This shows that BGCN can better mine the dynamic spatial-temporal dependence from traffic data.

- Compared to Graph WaveNet and AGCRN, BGCN shows a slower performance decline trend. This is mainly benefited from the better graph structure modeling.

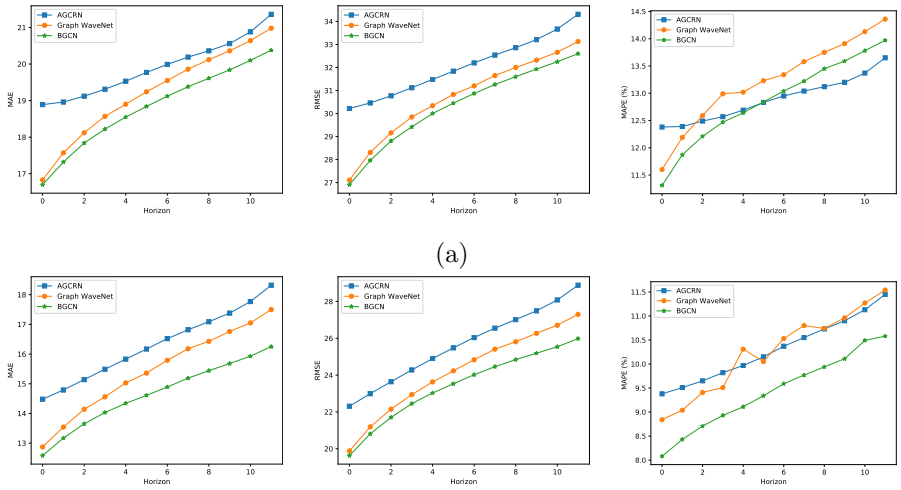


Figure 3: Prediction performance comparison at each horizon. (a) PeMS4 dataset (b) PeMS8 dataset.

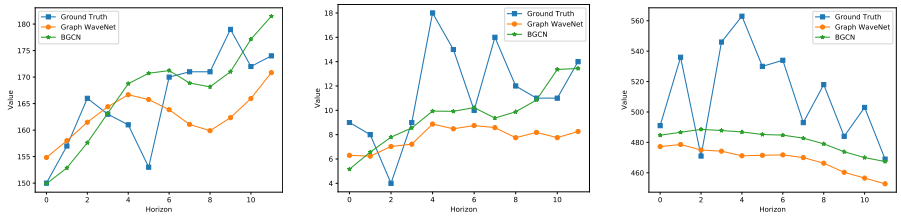
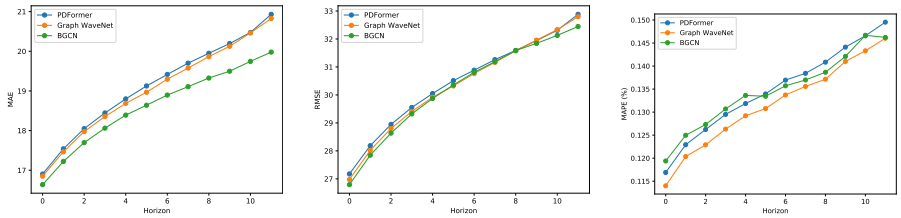


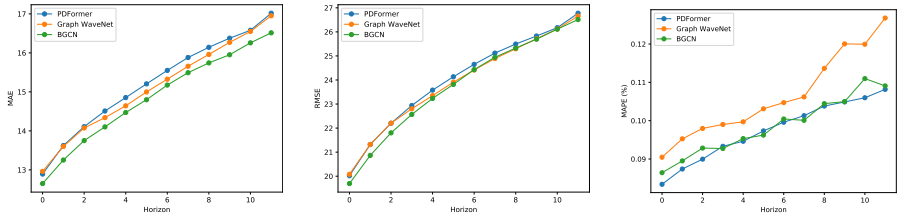
Figure 4: Visualization of some predicted results on the PeMS8 dataset.

Fig. 4 visualizes some predicted results of Graph WaveNet and BGCN on the PeMS8 dataset. We can observe that the estimated results of BGCN are more related to the ground truth values compared to those of Graph WaveNet.

We also compare the proposed method with a transformer-based approach named PDFormer [62]. For fair comparison, we retrain our proposed model and Graph Wavenet [5] using the same framework called LibCity [63], as utilized in PDFormer. Furthermore, PDFormer is also retrained with the same training settings as the proposed method. Specifically, we set batch size, training epoch, and random seed to 64, 100, and 0, respectively. The training tech-



(a)



(b)

Figure 5: Single-step performance comparison with the transformer-based method [62]. (a) PeMS4 dataset (b) PeMS8 dataset.

niques, such as data augmentation and curriculum learning, have been disabled, along with the exclusion of time-related information including daytime and weekday embeddings. The single-step and average performance are presented in Figure 5 and Table 3 , respectively. Based on these results, we can draw the following conclusions. Firstly, under the training settings provided by LibCity, our proposed BGCN outperforms the baseline model, Graph WaveNet. This confirms the effectiveness of the proposed method. Secondly, the proposed BGCN demonstrates comparable performance to PDFormer in terms of RMSE and MAPE, while exhibiting a noticeable improvement in terms of MAE compared to PDFormer. The reason for this could be attributed to the fact that transformer-based methods necessitate a substantial amount of data in order to achieve optimal performance, whereas traffic data is typically limited in scale. Thirdly, the proposed BGCN is more memory-efficient than PDFormer, as the latter fails to be trained on the PeMS07 dataset with a large road structure. This is mainly because that the multi-head attention in PDFormer is memory-consuming.

Table 3: Average performance comparison with the transformer-based method [62]. GWN denots the baseline, i.e., Graph WaveNet [5]. OOM means that GPU is out of memory.

Model	PeMS4			PeMS7			PeMS8		
	MAE	RMSE	MAPE (%)	MAE	RMSE	MAPE (%)	MAE	RMSE	MAPE (%)
PDFormer	19.13	30.44	13.48		OOM		15.23	24.02	**9.75**
GWN	19.04	30.34	**13.17**	20.94	33.87	8.91	15.11	23.90	10.64
BGCN	**18.60**	**30.24**	13.46	**20.29**	**33.39**	**8.56**	**14.85**	**23.75**	9.86

4.5. Computational Complexity

Table 4: The computation cost on the pems8 dataset. "dim" means the dimension of node embedding. "*" denotes the setting of node embedding for the pems4 dataset.

Model	# Parameters	Training Time	Inference Time
STGCN	211596	12.35 s	3.6 s
Graph WaveNet	305228	31.20 s	5.4 s
AGCRN (dim = 2)	150386	33.88 s	13.75 s
AGCRN* (dim = 10)	748810	35.56 s	14.28 s
BGCN	256076	33.14 s	4.4 s

Table 4 reports the computation cost, i.e., the parametric size, training time per epoch, and inference time, of our method, STGCN, Graph WaveNet, and AGCRN on the PeMSD8 dataset. We have the following observations.

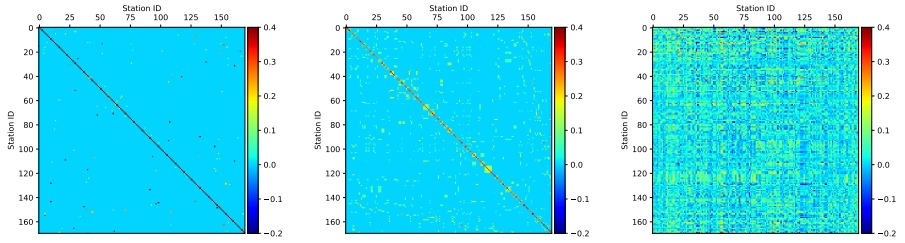
- Even with fewer parameters, AGCRN is not as good as STGCN and Graph WaveNet in terms of training time and inference time per epoch. This is mainly because AGCRN uses RNNs to sequentially process traffic data rather than CNNs.

- Compared with STGCN, Graph WaveNet uses more parameters and is slower. Considering the significant performance improvement (as shown in Table 2), this is moderate.

- Compared with Graph WaveNet, BGCN uses fewer parameters while achieving excellent predictive performance. In addition, BGCN slightly

shortens training and inference time.

4.6. Visualized Results of Graph Structure



(a) (b) (c)

Figure 6: Illustration of adjacency matrices on the PeMS8 dataset. (a) The normalized observed adjacency matrix with self-loops $\tilde{\mathcal{A}}_{obs}$ (b) The constant adjacency matrix $\tilde{\mathcal{A}}_{\overline{g}}$ (c) The learned adjacency matrix $\tilde{\mathcal{A}}_{\overline{g}} + \phi$.

We also show the observed adjacency matrix, the constant adjacency matrix, and the learned adjacency matrix in Fig. 6. From this figure, we summarize three characteristics of the learned graph structure as follows.

1) Asymmetry: In contrast to the constant adjacency matrix with a symmetric format, the learned adjacency matrix is asymmetric. The reason for this is explained as follows. The upstream traffic state impacts the downstream one through the transfer effect, and the downstream traffic status influences the upstream one through the feedback effect [6]. Moreover, the downstream traffic state has a greater impact on the future traffic status than the upstream one [35]. As a result, it is reasonable that we discover an asymmetric spatial structure from traffic data.

2) Denseness: Compared to the observed adjacency matrix with sparse connections, the learned adjacency matrix is denser. Intuitively, a connection between two roads that are physically disconnected but with similar traffic conditions is likely to be created for minimizing the prediction error. Therefore, the learned adjacency matrix mines some underlying spatial dependence, which cannot be captured in the observed graph.

3) Arbitrariness: The element in the learned adjacency matrix can be an arbitrary value. It is interesting to observe some negative spatial dependence. The reason for this is presented below. Traffic data collected in scenarios with traffic congestion usually involves negative spatial correlations [59]. This is mainly because passengers' reactions to congestion cause traffic to shift from one road to another. As a result, it makes sense that we find some negative spatial dependence.

Table 5: Ablation study on the PeMS8 dataset.

Methods	B_a	B_b	B_c	B_d	B_e	B_f	B_g	BGCN
$\tilde{\mathcal{A}}_{obs}$	✓		✓					
$\tilde{\mathcal{A}}_{\overline{g}}$				✓		✓	✓	✓
ϕ	✓	✓			✓	✓	✓	✓
Uncertainty	✓				✓		✓	✓
ReLU							✓	
MAE	14.74	16.91	18.67	17.17	15.06	16.45	14.68	**14.65**
RMSE	23.53	26.60	29.19	26.92	23.96	25.49	23.50	**23.43**
MAPE (%)	9.50	10.37	11.93	10.82	9.53	10.95	9.49	**9.42**

4.7. Ablation study

BGCN considers the learnable adjacency matrix ϕ, the constant adjacency matrix $\tilde{\mathcal{A}}_{obs}$, and uncertainty in graph structure modeling. We verify the importance of each component in Table 5.

1) Evaluation on learnable adjacency matrix: B_b, B_c, and B_d leverage the learnable adjacency matrix ϕ, the observed adjacency matrix $\tilde{\mathcal{A}}_{obs}$, and the constant adjacency matrix $\tilde{\mathcal{A}}_{\overline{g}}$ to model the spatial dependence, respectively. We can observe that B_b works better than B_c and B_d in three evaluation metrics. This is because ϕ mines the spatial dependence that is fully targeted to the traffic prediction task while $\tilde{\mathcal{A}}_{obs}$ and $\tilde{\mathcal{A}}_{\overline{g}}$ are derived from the topology of the road network. We also can notice that B_b is inferior to BGCN, which reminds

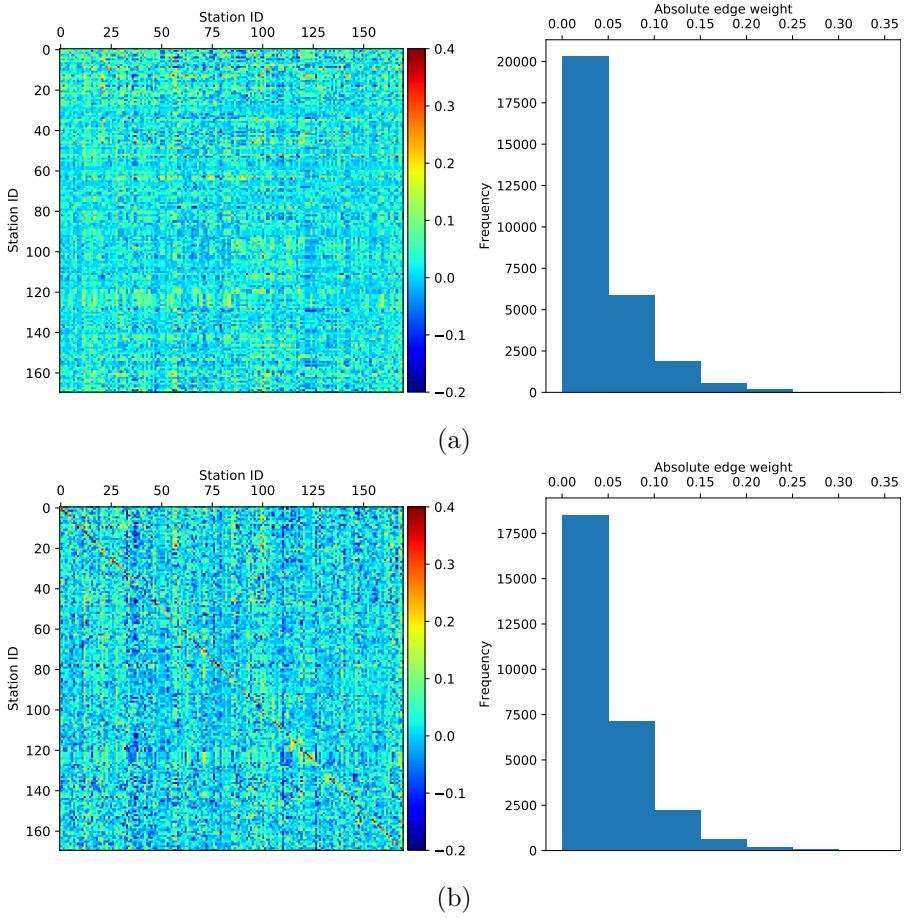


Figure 7: Impact of uncertainty on graph structure learning on the PeMS8 dataset. (a) Learning graph structure with uncertainty (b) Learning graph structure without uncertainty.

us that simply using ϕ for graph structure learning without two ingredients, i.e., $\tilde{\mathcal{A}}_{\overline{g}}$ and uncertainty, is not enough.

2) Evaluation on constant adjacency matrix: B_d outperforms B_c with a noticeable margin, which verifies the validity of posterior inference of the observed graph. This is mainly because $\tilde{\mathcal{A}}_{\overline{g}}$ discovers some underlying connections between roads (as shown in Fig. 6). We also explore a variant of BGCN, B_e which considers the uncertainty and learns the underlying graph structure from scratch. We can see that B_e performs worse than the original version, which

demonstrates the importance of $\tilde{\mathcal{A}}_{\overline{g}}$ in BGCN. Since $\tilde{\mathcal{A}}_{\overline{g}}$ already contains some prior knowledge of connections between roads, learning the residual of the latent graph is easier. Compared to B_a using $\tilde{\mathcal{A}}_{obs}$ as the guidance, BGCN achieves better performance. This confirms the superiority of $\tilde{\mathcal{A}}_{\overline{g}}$ over $\tilde{\mathcal{A}}_{obs}$.

3) Evaluation on uncertainty: B_f is another variant of BGCN, which ignores the uncertainty in the graph structure learning. As we can see, B_f is inferior to BGCN in all three metrics, which indicates the effectiveness of the uncertainty in BGCN. To figure out the role of the uncertainty in BGCN, we visualize the learned adjacency matrix and the histogram of absolute edge weights in Fig. 7. We can observe that the adjacency matrix learned without uncertainty has a larger scale than that of BGCN. This indicates that the uncertainty acts like a regularizer, which prevents ϕ from overfitting.

4) Evaluation on negative spatial dependence: B_g applies the $ReLU$ activation function on the result of ϕ plus $\tilde{\mathcal{A}}_{\overline{g}}$, which only considers postive spatial dependence. As we can see, B_g is sligtly inferior to BGCN, which verifies the importance of negative spatial dependence in traffic prediction.

4.8. Statistical Significance Analysis

Since learning-based methods may be sensitive to dataset splitting, we evaluate the proposed BGCN and the baseline Graph WaveNet in 10 train-validation-test trials in the PeMS8 dataset. The mean and standard deviation (std) of performance values are reported in Fig. 8. To avoid the issue of scale, we divide performance values by the maximum values. As shown in Fig. 8, the proposed BGCN can achieve a lower mean value and smaller std compared to Graph WaveNet in terms of MAE, RMSE, and MAPE, which confirms that BGCN performs more precisely and consistently.

4.9. Parameter Experiments

One key parameter in BGCN is the dropout rate in the Monte Carlo approximation. Fig. 9 shows the effects of different dropout rates to BGCN on the

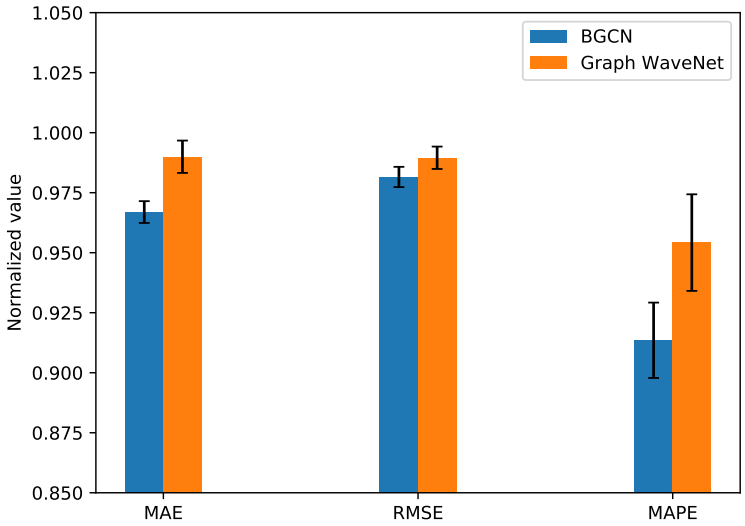


Figure 8: Mean performance values and standard error bars for learning-based algorithms across 10 train-validation-test trials in the PeMS8 database.

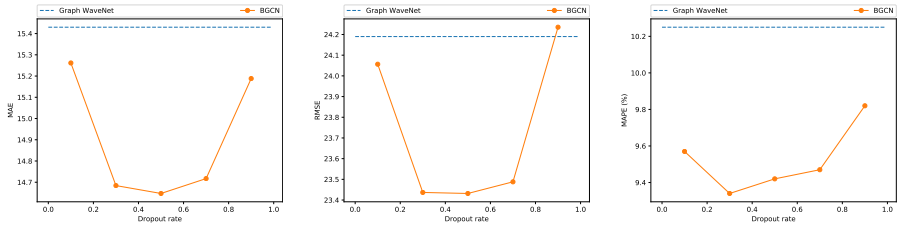


Figure 9: Investigation on the dropout rate in the Monte Carlo approximation.

PeMSD8 dataset. We can observe that BGCN improves the prediction performance of Graph WaveNet at almost all the tested dropout rates, showing the robustness of BGCN. We can also notice that BGCN achieves the best performance when the dropout rate is assigned to 0.5. In addition, both an excessively small and large dropout rate will weaken the prediction performance.

4.10. Generalization Experiments

Our proposed BGCN is a plug-and-play module. We generalize it to some other graph-based traffic prediction networks (i.e., STGCN and AGCRN). The results are listed in Table 6. We can observe that BGCN-applied STGCN and

Table 6: Exploration of generalization of BGCN on the PeMS8 dataset.

Metric	STGCN	STGCN-BGCN	AGCRN	AGCRN-BGCN
MAE	16.98	**16.70**	16.29	**15.63**
RMSE	26.58	**25.85**	25.66	**24.79**
MAPE (%)	11.58	**11.03**	**10.32**	10.38

AGCRN exceed original versions in terms of MAE and RMSE. In addition, we can find that BGCN slightly impairs the MAPE performance of AGCRN. One possible reason is that the objective function used guides the traffic prediction network towards minimizing MAE rather than minimizing MAPE.

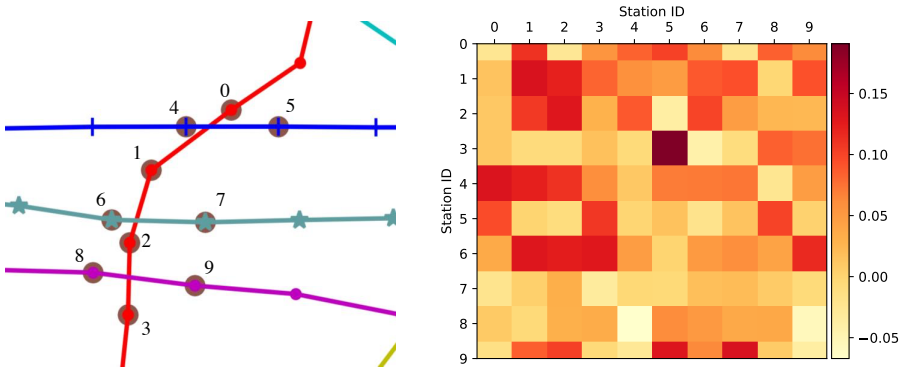


Figure 10: The adjacent matrix of a sub-graph with 10 roads.

4.11. Case Study

In order to investigate the proposed BGCN intuitively, we perform a case study: picking out a sub-graph with 10 roads from the PeMSD8 and showing its adjacent matrix. As shown in the right side of Fig. 10, in the adjacent matrix, the i-th row represents the correlation strength between each road and the i-th road. For example, from the seventh row, we can observe that traffic flows on the road 6 are closely related to those on the road 1, road 2, and road 3. This is reasonable because these four roads are spatially close, as shown on the left side of Fig. 10. In addition, from the fourth row, we can surprisingly observe

traffic flows on the road 3 are also closely related to those on the road 5, even though the two roads are far away from each other. This makes sense since the proposed BGCN considers both spatial distances between roads and traffic data in the construction of the adjacent matrix.

5. Conclusion and Future Work

In this paper, we propose a Bayesian Graph Convolutional Network for traffic prediction. It introduces the information of traffic data and uncertainty into the graph structure using a Bayesian approach. Moreover, it is a plug-and-play module for graph-based traffic prediction networks. Experimental results on five real-world datasets verify the effectiveness and the generalization ability of BGCN in traffic prediction. In the future, we focus on extending BGCN to other spatio-temporal time series forecasting tasks, such as forecasting ride demand.

Acknowledgment

This work was supported in part by NSFC under Grant U1908209, 61632001 and the National Key Research and Development Program of China 2018AAA0101400.

References

[1] F. Guerrini, Traffic congestion costs americans $124 billion a year, report says, Forbes, October 14.

[2] M. S. Ahmed, A. R. Cook, Analysis of freeway traffic time-series data by using Box-Jenkins techniques, no. 722, 1979.

[3] K. Holden, Vector auto regression modeling and forecasting, Journal of Forecasting 14 (3) (1995) 159–166.

[4] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv:1609.02907.

[5] Z. Wu, S. Pan, G. Long, J. Jiang, C. Zhang, Graph wavenet for deep spatial-temporal graph modeling, arXiv preprint arXiv:1906.00121.

[6] C. J. Dong, C. F. Shao, C.-X. Zhuge, M. Meng, Spatial and temporal characteristics for congested traffic on urban expressway, Journal of Beijing Polytechnic University 38 (8) (2012) 128–132.

[7] Y. Rong, W. Huang, T. Xu, J. Huang, Dropedge: Towards deep graph convolutional networks on node classification, in: International Conference on Learning Representations, 2019.

[8] J. Liu, W. Guan, A summary of traffic flow forecasting methods [j], Journal of highway and transportation research and development 3 (2004) 82–85.

[9] J. Guo, W. Huang, B. M. Williams, Adaptive kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification, Transportation Research Part C: Emerging Technologies 43 (2014) 50–64.

[10] S. Lee, D. B. Fambro, Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting, Transportation Research Record 1678 (1) (1999) 179–188.

[11] B. M. Williams, L. A. Hoel, Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results, Journal of transportation engineering 129 (6) (2003) 664–672.

[12] Y. Hou, P. Edara, Y. Chang, Road network state estimation using random forest ensemble learning, in: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2017, pp. 1–6.

[13] C.-H. Wu, J.-M. Ho, D.-T. Lee, Travel-time prediction with support vector regression, IEEE transactions on intelligent transportation systems 5 (4) (2004) 276–281.

[14] S. Sun, C. Zhang, G. Yu, A bayesian network approach to traffic flow forecasting, IEEE Transactions on intelligent transportation systems 7 (1) (2006) 124–132.

[15] X.-l. Zhang, G.-g. He, H.-p. Lu, Short-term traffic flow forecasting based on k-nearest neighbors non-parametric regression, Journal of Systems Engineering 24 (2) (2009) 178–183.

[16] F. G. Habtemichael, M. Cetin, Short-term traffic flow rate forecasting based on identifying similar traffic patterns, Transportation research Part C: emerging technologies 66 (2016) 61–78.

[17] B. Sun, W. Cheng, P. Goswami, G. Bai, Short-term traffic forecasting using self-adjusting k-nearest neighbours, IET Intelligent Transport Systems 12 (1) (2018) 41–48.

[18] D. Park, L. R. Rilett, Forecasting freeway link travel times with a multilayer feedforward neural network, Computer-Aided Civil and Infrastructure Engineering 14 (5) (1999) 357–367.

[19] Y. Lv, Y. Duan, W. Kang, Z. Li, F.-Y. Wang, Traffic flow prediction with big data: a deep learning approach, IEEE Transactions on Intelligent Transportation Systems 16 (2) (2014) 865–873.

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

[21] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[22] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, arXiv preprint arXiv:1406.1078.

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

[24] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, Y. Liu, Deep learning: A generic approach for extreme condition traffic forecasting, in: Proceedings of the 2017 SIAM international Conference on Data Mining, SIAM, 2017, pp. 777–785.

[25] Z. Zhao, W. Chen, X. Wu, P. C. Chen, J. Liu, Lstm network: a deep learning approach for short-term traffic forecast, IET Intelligent Transport Systems 11 (2) (2017) 68–75.

[26] N. Laptev, J. Yosinski, L. E. Li, S. Smyl, Time-series extreme event forecasting with neural networks at uber, in: International Conference on Machine Learning, Vol. 34, 2017, pp. 1–5.

[27] Y. Tian, K. Zhang, J. Li, X. Lin, B. Yang, Lstm-based traffic flow prediction with missing data, Neurocomputing 318 (2018) 297–305.

[28] Y. Wu, H. Tan, Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework, arXiv preprint arXiv:1612.01022.

[29] H. Yu, Z. Wu, S. Wang, Y. Wang, X. Ma, Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks, Sensors 17 (7) (2017) 1501.

[30] J. Wang, Q. Gu, J. Wu, G. Liu, Z. Xiong, Traffic speed prediction and congestion source exploration: A deep learning method, in: 2016 IEEE 16th international conference on data mining (ICDM), IEEE, 2016, pp. 499–508.

[31] J. Zhang, Y. Zheng, D. Qi, R. Li, X. Yi, T. Li, Predicting citywide crowd flows using deep spatio-temporal residual networks, Artificial Intelligence 259 (2018) 147–166.

[32] Y. Zhang, S. Wang, B. Chen, J. Cao, Z. Huang, Trafficgan: Network-scale deep traffic prediction with generative adversarial nets, IEEE Transactions on Intelligent Transportation Systems 22 (1) (2019) 219–230.

[33] B. Du, H. Peng, S. Wang, M. Z. A. Bhuiyan, L. Wang, Q. Gong, L. Liu, J. Li, Deep irregular convolutional residual lstm for urban traffic passenger flows prediction, IEEE Transactions on Intelligent Transportation Systems 21 (3) (2019) 972–985.

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

[35] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, arXiv preprint arXiv:1707.01926.

[36] B. Yu, H. Yin, Z. Zhu, Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting, arXiv preprint arXiv:1709.04875.

[37] M. Lv, Z. Hong, L. Chen, T. Chen, T. Zhu, S. Ji, Temporal multi-graph convolutional network for traffic flow prediction, IEEE Transactions on Intelligent Transportation Systems.

[38] H. Peng, H. Wang, B. Du, M. Z. A. Bhuiyan, H. Ma, J. Liu, L. Wang, Z. Yang, L. Du, S. Wang, et al., Spatial temporal incidence dynamic graph neural networks for traffic flow forecasting, Information Sciences 521 (2020) 277–290.

[39] R. Huang, C. Huang, Y. Liu, G. Dai, W. Kong, Lsgcn: Long short-term traffic prediction with graph convolutional networks., in: IJCAI, 2020, pp. 2355–2361.

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

[41] X. Feng, J. Guo, B. Qin, T. Liu, Y. Liu, Effective deep memory networks for distant supervised relation extraction., in: IJCAI, 2017, pp. 4002–4008.

[42] C. Song, Y. Lin, S. Guo, H. Wan, Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 914–921.

[43] L. Bai, L. Yao, C. Li, X. Wang, C. Wang, Adaptive graph convolutional recurrent network for traffic forecasting, arXiv preprint arXiv:2007.02842.

[44] G. Guo, W. Yuan, Short-term traffic speed forecasting based on graph attention temporal convolutional networks, Neurocomputing 410 (2020) 387–393.

[45] W. Li, X. Wang, Y. Zhang, Q. Wu, Traffic flow prediction over muti-sensor data correlation with graph convolution network, Neurocomputing 427 (2021) 50–63.

[46] X. Yin, G. Wu, J. Wei, Y. Shen, H. Qi, B. Yin, Multi-stage attention spatial-temporal graph networks for traffic prediction, Neurocomputing 428 (2021) 42–53.

[47] W. Chen, L. Chen, Y. Xie, W. Cao, Y. Gao, X. Feng, Multi-range attentive bicomponent graph convolutional network for traffic forecasting, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 34, 2020, pp. 3529–3536.

[48] C. Zheng, X. Fan, C. Wang, J. Qi, Gman: A graph multi-attention network for traffic prediction, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 34, 2020, pp. 1234–1241.

[49] B. Jiang, Z. Zhang, D. Lin, J. Tang, B. Luo, Semi-supervised learning with graph learning-convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 11313–11320.

[50] S. Yan, Y. Xiong, D. Lin, Spatial temporal graph convolutional networks for skeleton-based action recognition, in: Thirty-second AAAI conference on artificial intelligence, 2018.

[51] J. Xu, W. Zhou, Z. Chen, Blind omnidirectional image quality assessment with viewport oriented graph convolutional networks, arXiv preprint arXiv:2002.09140.

[52] S. Sun, T. Yu, J. Xu, J. Lin, W. Zhou, Z. Chen, Graphiqa: Learning distortion graph representations for blind image quality assessment, arXiv preprint arXiv:2103.07666.

[53] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, Y. Shen, Graph contrastive learning with augmentations, Advances in Neural Information Processing Systems 33 (2020) 5812–5823.

[54] Y. Zhang, S. Pal, M. Coates, D. Ustebay, Bayesian graph convolutional neural networks for semi-supervised classification, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 5829–5836.

[55] S. Pal, F. Regol, M. Coates, Bayesian graph convolutional neural networks using non-parametric graph learning, arXiv preprint arXiv:1910.12132.

[56] T. N. Kipf, M. Welling, Variational graph auto-encoders, arXiv preprint arXiv:1611.07308.

[57] V. Kalofolias, N. Perraudin, Large scale graph learning from smooth signals, arXiv preprint arXiv:1710.05654.

[58] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, The journal of machine learning research 15 (1) (2014) 1929–1958.

[59] A. Ermagun, S. Chatterjee, D. Levinson, Using temporal detrending to observe the spatial correlation of traffic, PloS one 12 (5) (2017) e0176853.

[60] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in pytorch.

[61] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

[62] J. Jiang, C. Han, W. X. Zhao, J. Wang, Pdformer: Propagation delay-aware dynamic long-range transformer for traffic flow prediction, arXiv preprint arXiv:2301.07945.

[63] J. Wang, J. Jiang, W. Jiang, C. Li, W. X. Zhao, Libcity: An open library for traffic prediction, in: Proceedings of the 29th international conference on advances in geographic information systems, 2021, pp. 145–148.