MSDR: Multi-Step Dependency Relation Networks for Spatial Temporal Forecasting

Dachuan Liu*
University of Electronic Science and Technology of China
Chengdu, China
dcliu@std.uestc.edu.cn

Jin Wang*
University of California, Los Angeles
Los Angeles, USA
jinwang@cs.ucla.edu

Shuo Shang†
University of Electronic Science and Technology of China
Chengdu, China
jedi.shang@gmail.com

Peng Han†
Aalborg University, Denmark
Aalborg, Denmark
pengh@cs.aau.dk

ABSTRACT

Spatial temporal forecasting plays an important role in improving the quality and performance of Intelligent Transportation Systems. This task is rather challenging due to the complicated and long-range spatial temporal dependencies in traffic network. Existing studies typically employ different deep neural networks to learn the spatial and temporal representations so as to capture the complex and dynamic dependencies. In this paper, we argue that it is insufficient to capture the long-range spatial dependencies from the implicit representations learned by temporal extracting modules. To address this problem, we propose Multi-Step Dependency Relation (MSDR), a brand new variant of recurrent neural network. Instead of only looking at the hidden state from only one latest time step,MSDR explicitly takes those of multiple historical time steps as the input of each time unit. We also develop two strategies to incur the spatial information into the dependency relation embedding between multiple historical time steps and the current one in MSDR. On the basis of it, we propose the Graph-based MSDR (GMSDR) framework to support general spatial temporal forecasting applications by seamlessly integrating graph-based neural networks with MSDR. We evaluate our proposed approach on several popular datasets. The results show that the proposed GMSDR framework outperforms state-of-the-art methods by an obvious margin.

CCS CONCEPTS

• **Information systems → Data mining**; • **Computing methodologies → Neural networks**; *Artificial intelligence.*

*Both authors contributed equally to this research.
†Corresponding Author.

KEYWORDS

traffic forecasting, multi-step dependency, relation embedding, neural networks

ACM Reference Format:
Dachuan Liu, Jin Wang, Shuo Shang, and Peng Han. 2022. MSDR: Multi-Step Dependency Relation Networks for Spatial Temporal Forecasting. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3534678.3539397

1 INTRODUCTION

Spatial temporal forecasting has been an important problem in many real world applications, such as traffic prediction [15, 36], climate forecasting [16] and urban monitoring system analysis [24]. A typical example is traffic flow prediction, which attempts to predict the future traffic flow given historical traffic conditions and underlying road networks. Due to the ever growing transportation related datasets, spatial temporal forecasting is playing a more and more important role in improving the performance and service quality of Intelligent Transportation Systems.

Nevertheless, this task is inherently challenging due to the complicated spatial temporal correlations: On the one hand, the spatial and temporal dependencies are non-linear and dynamic; on the other hand, it is rather difficulty to capture the shifting long-range dependency. There is a long stream of previous studies aiming at solving this problem. Early works [21, 25] tried to model the utilize statistical and traditional machine learning models to capture the correlation. Deep learning techniques have been widely used in the applications of spatial temporal forecasting. These approaches usually utilized variants of Recurrent Neural Network (RNN) to capture temporal correlations [5, 20] and use Convolutional Neural Network (CNN) to extract the spatial features from the traffic network [35]. Some hybrid approaches [30, 31] further combined CNN and the Long Short-term Memory (LSTM) [14] network to jointly model the dependencies from both information. A recent trend in this field is to employ Graph Neural Network (GNN) [9, 18] to effectively model the spatial dependency and then integrate with temporal extraction models to learn the temporal dynamics [29, 33, 34]. The reason is that most of the traffic indicators are given along road network or associated with fixed station which can be naturally modeled as graph structures.

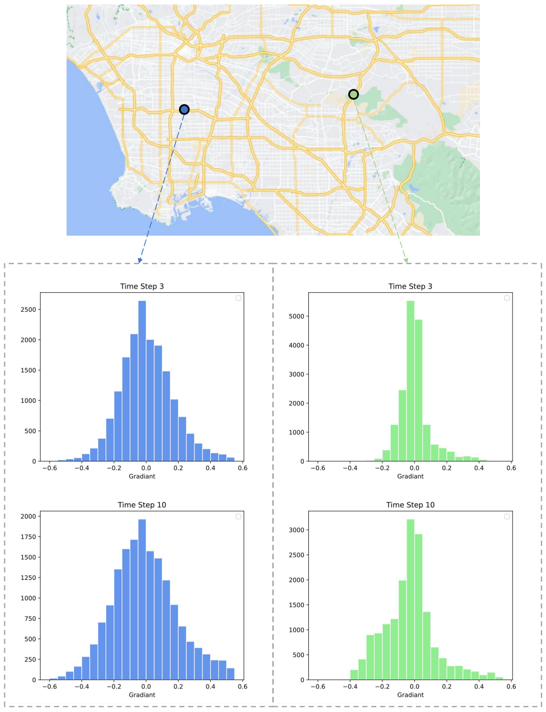


Figure 1: Example: Information from Multiple Time-steps in Traffic Network

Although significant efforts have been made in previous studies to address such issues, they are still sub-optimal in capturing the long-range spatial temporal dependencies. The reason is that they mainly rely on RNN variants to capture the temporal dynamics. Since RNN variants learn the representation from both the current input and the hidden state from the previous time step to involve historical information, they require iterative training, which would (i) lead to severe information oblivion in modeling the long-range temporal dependency and (ii) introduce noises into historical information due to the error accumulation by steps. We argue that the temporal representation of the current input will depend on multiple previous hidden states, which in turn have varying degrees of correlation with features of the current moment. Thus it is essential to explicitly handle the temporal correlations within certain time ranges. An example from real traffic data is illustrated in Figure 1. We can see that in both locations, the temporal dependencies between current time step and historical ones at different locations vary greatly. Meanwhile, in different previous time steps at the same location, there are similar patterns pattern of dependencies. That means, it would be easier to capture such temporal dependency by looking into the states from more previous time steps. As a result, it is essential to develop a method that can better utilize such historical information.

In this paper, we propose GMSDR, a brand new framework to support spatial temporal forecasting applications. Unlike previous RNN based approaches that only take one hidden state from the latest time step as the input of current time step, GMSDR explicitly includes the hidden states from multiple previous time steps. In this way, the complex spatial-temporal correlations can be captured from the interactions between historical hidden states and the input of current moment. To reach this goal, we propose Multi-Step Dependency Relation (MSDR), a novel RNN variant which calculates the temporal representation based on the input current time step and multiple hidden states from previous time steps. The core idea is that the dependency relations between multiple historical time steps and the current time step are explicitly modeled as relation embedding learned by a temporal interaction mechanism. Specifically, we propose two strategies to incur the spatial information into the dependency relation embedding of MSDR so as to improve the overall performance. Then we employ the attention mechanism to aggregate the features learned from different time steps so as to distinguish the impact from them. Finally, we develop the GMSDR framework by combining above proposed techniques with GNN module and the popular encoder-decoder architecture.

Our contributions in this paper are summarized as following:

- We recognize the importance of explicitly utilizing hidden states from previous multiple time steps in spatial temporal forecasting and proposed the novel MSDR model to learn high quality representation from them. Based on that, we further integrate MSDR into the encoder-decoder architecture and propose the GMSDR framework for spatial temporal forecasting.
- We propose a dependency relation operation to better capture the long-range dependency from the interactions of previous hidden states and the input of current time step.
- We conduct extensive evaluation on several popular datasets. Experimental results show that our proposed framework outperforms state-of-the-art methods by an obvious margin on all datasets.

2 RELATED WORK

Spatial temporal forecasting has attracted tremendous attention in recent years. Earlier studies tried to solve this problem by utilizing the statistical and traditional machine learning [21, 22, 25], such as autoregressive integrated moving average (ARIMA), k-nearest neighbors algorithm (kNN), and support vector machine (SVM). However, such approaches mainly considered the temporal features and are not able to properly learn spatial ones. Recently deep learning techniques have been widely adopted to various fields [11–13], especially in the problem of spatial temporal forecasting [23]. ST-ResNet [35] modeled the traffic network as grids and utilized CNN to capture spatial correlations. DMVST-Net [31] and STDN [31] integrated CNN and LSTM models to jointly learn the spatial and temporal dependencies. CoST-Net [32] utilized the correlations between different modes of transportation and made a co-prediction. GMAN [37] and MRA-BGCN [3] utilized attention mechanism to model the impact of spatial-temporal factors on traffic conditions.

A recent trend is to utilize GNN models for spatial temporal forecasting. Due to its power in dealing with graph structured data, GNN learns the node representation by aggregating representations from their neighbors and thus achieves effective performance in many tasks like node classification and link prediction. Thus, GNN can effectively learn the spatial features as well as complicated dependencies. STGCN [34] first utilized GNN to learn the

Table 1: Notations

Notations	Description
N	The number of locations
K	The number of previous time steps
d_l	The dimension of hidden state in l-th layer
$\boldsymbol{X}_t$	The input of time step t
$\boldsymbol{Y}_t, \hat{\boldsymbol{Y}}_t$	The labels and predicted values of time step t
$\boldsymbol{h}_{i,t}^{(l)}$	The hidden state of i-th location at time step t in l-th layer
$\boldsymbol{r}_k^{(l)}$	The dependency relation for the k-th previous time step in the l-th layer
$\boldsymbol{r}_{i,k}^{(l)}$	The dependency relation of i-th location for the k-th previous time step in the l-th layer
$\Theta_{\star\mathcal{G}}$	A type of graph neural network with parameter set θ
$\Vert$	Concatenation operation of vectors
$\boldsymbol{H}_{i,t}^{(l)}$	Stacking of the hidden state of i-th location and its neighbors at time step t in l-th layer

spatial features and developed a fully convolutional framework. STG2Seq [1] applied graph convolutional module on top of LSTM to learn spatial-temporal correlations. DCRNN [20] combined GNN with Gated Recurrent Unit (GRU) in an encoder-decoder architecture. Some other studies, such as Graph WaveNet [29], CGRNN [7], CCRNN [33],DMSTGCN [10] and STODE [6], aimed at improving the graph representation learning step and followed this paradigm to build the overall framework. They are orthogonal to our work. We will show that our proposed framework can be seamlessly integrated with such methods in Section 4 later. There are also some works about utilizing GNN for time series forecasting such as MT-GNN [28] and StemGNN [2]. These works have different problem settings with this paper. It is an interesting direction to explore how to extend our proposed MSDR model to support this task in the future work.

3 PROBLEM DEFINITION

In this section, we formally introduce the problem definition of spatial temporal forecasting as well as some typical applications. The notations used to describe our framework in this and next sections are summarized in Table 1.

Spatial Networks The spatial network is represented as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where $\mathcal{V}$ denotes the set of traffic locations scattered throughout the zone and $|\mathcal{V}| = N$. $\mathcal{E}$ corresponds to the set of edges connected between locations. $\mathcal{A}$ is the adjacency matrix on the graph that records the non-Euclidean distances between locations.

Spatial Temporal Forecasting Problem Let $\boldsymbol{X}_t = \{\boldsymbol{x}_{1,t}, ..., \boldsymbol{x}_{N,t}\} \in \mathbb{R}^{N\times f}$ represent the features of all locations at the t-th interval, where f is the input feature dimension. The goal of spatial temporal forecasting can be described as Equation (1):

$$[\boldsymbol{X}_{t-\tau+1}, ..., \boldsymbol{X}_t] \xrightarrow{\mathcal{H}(\cdot)} [\hat{\boldsymbol{Y}}_{t+1}, ..., \hat{\boldsymbol{Y}}_{t+\lambda}], \quad (1)$$

where $[\boldsymbol{X}_{t-\tau+1}, ..., \boldsymbol{X}_t]$ denotes a feature sequence of τ historical times. The goal is to learn a mapping function $\mathcal{H}(\cdot)$ that can predict the spatial temporal features at λ future time steps.

There might be different applications of the spatial temporal forecasting problem, such as *traffic flow prediction* and *traffic demand prediction*. The problem definition of them is similar: given the traffic network and several historical time steps, it aims at predicting the signals (traffic flow or demand) of the next several time steps. The difference lies in the concentration of the problems: demand prediction mainly focused on identifying the pattern of periodical changes, e.g. the start and end time, on the same location, which tends to stay stable within a long time period. Meanwhile, traffic flow usually changes rapidly due to factors like weather and accidents. Thus flow prediction cares more about predicting the values in real-time or within a short-range of time.

4 METHODOLOGY

In this section, we introduce our proposed framework. We first present the new RNN variant MSDR that can takes multiple previous time steps in Section 4.1. Then we propose two strategies to make full use of multiple time steps in Section 4.2. Next we introduce how to incorporate MSDR into the overall GMSDR framework for spatial temporal forecasting in Section 4.3 and present the loss function for training in Section 4.4.

4.1 MSDR: the new RNN variant

Existing RNN variants, such as LSTM [14] and GRU [4], capture the temporal dependency relationship implicitly with complex gate computations. However, they suffer from severe information oblivion in modeling the long-range temporal dependency and thus lead to sub-optimal performance in spatial temporal forecasting. To solve this problem, we propose a new time sequence prediction model MSDR, which could explicitly capture a location biased dependency relationship.

The intuition of MSDR is that the representation of current time step could be influenced by multiple previous time steps. Thus we choose to involve the hidden states from the previous K time steps into the computation of the current output. Here K is a tunable parameter. Since the duration of influence might be different for different location-based applications, MSDR could be robust for different scenarios by properly tuning the value of K (Detailed experiments about it will be shown in Section 5.3 later).

The overall computation process of our temporal explicit dependency is illustrated in Figure 2. To explicitly model the dependency between different time steps, we use a explicit *dependency relation operation* $\boldsymbol{h}_{t-k} + \boldsymbol{r}_k$ to denote the influence caused by the k-th previous time step, where $\boldsymbol{h}_{t-k}$ is the hidden state of k-th previous time step and $\boldsymbol{r}_k$ is the dependency relation between the k-th previous time step and current one. We make $\boldsymbol{r}_k$ a trainable embedding which could learn the temporal dependency globally. In this way, our dependency operation could satisfy the requirements of layer-wise model structure that is common in RNN variants. For different layers and different values of k, the dependency relation embedding could be different. Then for t-th time step from K previous time steps, the l-th layer temporal dependency gate of i-th location

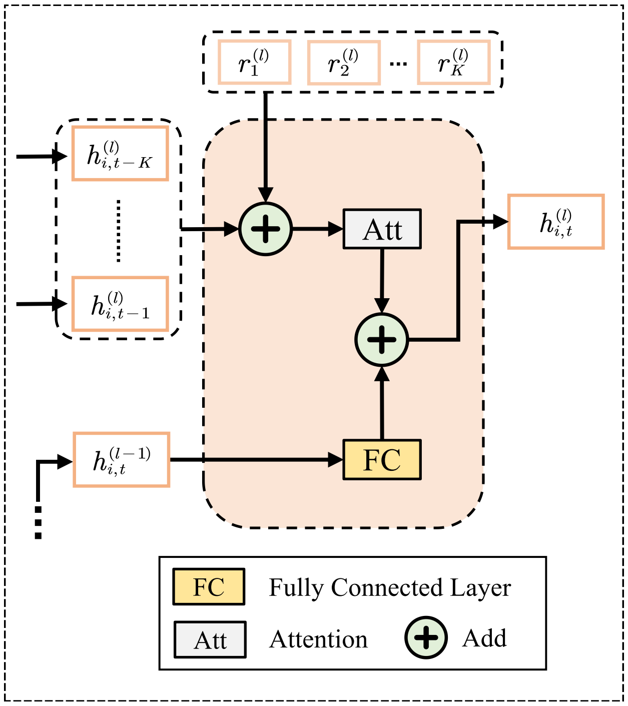


Figure 2: Computation Process of Temporal Explicit Dependency

$g_{i,t}^{(l)} \in \mathbb{R}^{d_l}$ can be computed as Equation (2):

$$g_{i,t}^{(l)} = \sum_{k=1}^{K} \alpha_k^l (h_{i,t-k}^{(l)} + r_k^{(l)}). \tag{2}$$

where $h_{i,t-k}^{(l)} \in \mathbb{R}^{d_l}$ is the hidden state of the k-th previous time step in the l-th layer of i-th location, $r_k^{(l)} \in \mathbb{R}^{d_l}$ is the corresponding dependency for the t-th time step in the l-th layer and α_k^l is the weight to control the contribution of the k-th latest time step.

With above computation, $g_{i,t}^{(l)}$ could explicitly capture the temporal dependencies between K previous time steps and the current time step. Taking traffic flow prediction as the example, when l-th layer is the top most layer, the meaning of $h_{i,t-k}^{(l)} \in \mathbb{R}^1$ could be explained as the predicted traffic flow at $(t-k)$-th time step. Then the computation of predicted traffic flow at t-th time step will explicitly contain the traffic flows of K latest time steps.

Since the influence parameter α_k^l could control the contribution of different latest time steps, it is crucial to find a proper way to automatically set its value. Inspired by the attention mechanism, we exploit it to assign the value of α_k^l and aggregate the results from each previous time step as Equation (3):

$$\alpha_k^{(l)} = \frac{exp\left((h_{i,t-k}^{(l)} + r_k^{(l)})W_\alpha^{(l)} + b_\alpha^{(l)}\right)}{\sum_{k=1}^{K} exp\left((h_{i,t-k}^{(l)} + r_k^{(l)})W_\alpha^{(l)} + b_\alpha^{(l)}\right)}, \tag{3}$$

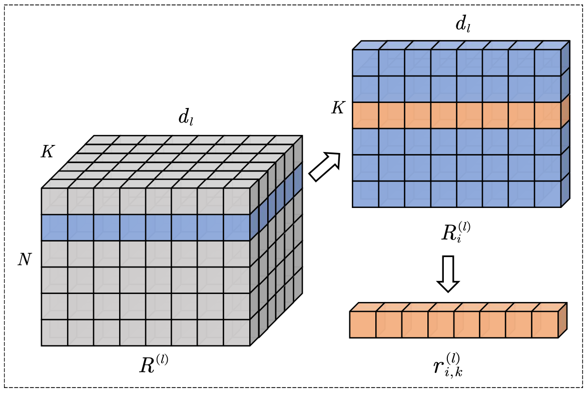


Figure 3: Selection Process of Explicit Spatial Dependency

where $W_\alpha^{(l)}$ and $b_\alpha^{(l)}$ are trainable parameters. Besides, the attention weight can also reflect the dependency between K previous time steps and current time step.

Moreover, the obtained information of current time step in $(l-1)$-th layer should also be considered in its computation of the embedding in l-th layer. Given the embedding $h_{i,t}^{(l-1)}$ in $(l-1)$-th layer of i-th location, we use $s_{i,t}^{(l)} \in \mathbb{R}^{d_l}$ to denote the information propagated from $(l-1)$-th layer as Equation (4):

$$s_{i,t}^{(l)} = h_{i,t}^{(l-1)} W^{(l)} + b^{(l)}, \tag{4}$$

where $W^{(l)} \in \mathbb{R}^{d_{l-1} \times d_l}$ and $b^{(l)}$ are trainable parameters to project the $h_{i,t}^{(l-1)}$ from the d_{l-1}-dimension space into the d_l-dimension space. The meaning of $h_{i,t}^{(l-1)}$ could be user-specified when $l = 1$. In our setting, the $h_{i,t}^{(0)}$ could be set as $x_{i,t}$ in the training process of corresponding location, which is the ground truth of t-th time step. If the specific application needs to involve more kinds of contextual information, e.g. the weather and date of i-th location, in the given input, we could also include them in the composition of $h_{i,t}^{(0)}$.

Now, we could get the hidden state of t-th time step in l-th layer of i-th location by combining $g_{i,t}^{(l)}$ and $s_{i,t}^{(l)}$ as

$$h_{i,t}^{(l)} = s_{i,t}^{(l)} + g_{i,t}^{(l)}. \tag{5}$$

There are mainly two advantages of our model. Firstly, our model could explicitly capture the dependency between multiple previous time steps and current time step, which makes it more explainable. Secondly, with the similar input and output structure as traditional recurrent neural networks, it is easy to integrate MSDR into many existing spatial temporal forecasting frameworks to support multiple kinds of applications.

4.2 Spatial dependency

Since spatial information also plays an important role in spatial temporal forecasting, it is beneficial to include them in the process of calculating the dependency relation embedding. In this section, we propose two strategies to reach this goal.

4.2.1 Data-based Spatial dependency. Since MSDR relies on dependency relation embedding learned from each time step to capture such correlations, it would be rather helpful to involve spatial information into this learning process. For the dependency relation embedding $r_k^{(l)}$ in Eq. (4), it will be the same for different locations, which cannot reflect the spatial difference. To make the dependency relation embedding spatial-aware, we propose a data-based spatial dependency embedding mechanism.

Specifically, the dependency relation embedding $r_k^{(l)}$ is obtained from the k-th latest hidden embedding $\boldsymbol{h}_{i,t-k}^{(l)} \in \mathbb{R}^{d_l}$ for all t, and the values of $\boldsymbol{h}_{i,t-k}^{(l)}$ depend on the input sequence. On the basis of this observation, we initialize a three dimension tensor $\boldsymbol{R}^{(l)} \in \mathbb{R}^{d_l \times K \times d_l}$, which could make the dependency relation embedding $r_k^{(l)}$ highly related to the sequential data with following operations. Next we need to select $r_k^{(l)}$ from $\boldsymbol{R}^{(l)}$. Specifically, the choice of the first dimension in $\boldsymbol{R}^{(l)}$ for $\boldsymbol{r}_k^{(l)}$ can be decided by the value of $\boldsymbol{h}_{t-k}^{(l)}$ as Equation (6):

$$idx_k = argmax(\boldsymbol{h}_{i,t-k}^{(l)}), \tag{6}$$

where $argmax(\boldsymbol{h})$ is the operation to generate the index of maximum value in the vector $\boldsymbol{h}$. Then $\boldsymbol{R}_{idx_k}^{(l)} \in \mathbb{R}^{K \times d_l}$ is selected from $\boldsymbol{R}^{(l)}$ with the index of first dimension idx_k, and we could get the $r_k^{(l)}$ from $\boldsymbol{R}_{idx_k}^{(l)}$ as Equation (7):

$$r_k^{(l)} = L(\boldsymbol{R}_{idx_k}^{(l)}, k). \tag{7}$$

where $L(\boldsymbol{R}, k)$ is the operation to select the k-th row of matrix $\boldsymbol{R}$. In this way, the vector $r_k^{(l)}$ will be spatial-aware. The specific values will be decided by the locations in different time steps of the sequential input data.

4.2.2 Explicit Spatial dependency. Although the dependency relation embedding in Equation (7) could capture the spatial information, it does not directly reflect the location information. Since there might be information loss during the iterative training process of MSDR, the spatial dependency carried in the input data might also be insufficient. To capture richer location-biased temporal dependencies, we propose another strategy to construct the dependency embedding $r_k^{(l)}$ by explicitly introducing spatial information.

The whole selection process of spatial explicit dependency is illustrated in Figure 3. For each layer l, we initially construct a trainable 3-dimension tensor $\boldsymbol{R}^{(l)} \in \mathbb{R}^{N \times K \times d_l}$, where d_l is the embedding dimension of l-th layer. Then different locations could have different temporal dependency in $\boldsymbol{R}^{(l)}$. We use $\boldsymbol{R}_i^{(l)} \in \mathbb{R}^{K \times d_l}$, i.e. the i-th matrix in $\boldsymbol{R}^{(l)}$, to denote the dependency relation embedding of i-th location in l-th layer. Then for the i-th location, we could calculate the vector $\boldsymbol{r}_{i,k}^{(l)}$ as Equation (8):

$$\boldsymbol{r}_{i,k}^{(l)} = L(\boldsymbol{R}_i^{(l)}, k) \tag{8}$$

By replacing $\boldsymbol{r}_k^{(l)}$ in Equation (2) and (3) with $\boldsymbol{r}_{i,k}^{(l)}$, we could learn the location-biased temporal dependency explicitly.

4.3 Overall Framework

Next we introduce the overall framework GMSDR to support spatial temporal forecasting applications in Figure 4. It is based on the encoder-decoder architecture as previous studies [20, 33] did. Specifically, the encoder encodes the sequential data into hidden embeddings and the decoder turns such embeddings into predicted sequential values. Here in both the encoder and decoder, we use MSDR as the sequential model instead of GRU in previous studies. GNN models have been applied in many previous studies of spatial temporal forecasting to learn the latent representation of the spatial network as well as spatial temporal correlations since they are good at incorporating the information of neighbors. In our work, we do not aim at developing new GNN variants to fulfill this functionality. Instead, we show a framework which could combine existing graph-based methods with our proposed MSDR. This can be realized by the *graph-based gates* introduced as following.

Firstly, we denote the abstracted graph convolution operation as $\Theta_{\star\mathcal{G}}(\boldsymbol{X}_G, \boldsymbol{A})$, where Θ is the set of learned parameters, $\boldsymbol{X}_G$ is the notation of input features and $\boldsymbol{A}$ is the graph. Here $\boldsymbol{X}_G$ can be any kind of GNN methods developed in previous studies, such as [6, 7, 29, 33]. Following the route of previous studies, we apply the graph convolutional operation in the information gate operation $\boldsymbol{s}_{i,t}^{(l)}$ in our MSDR module to reinforce the information propagation by incurring neighbor embeddings. Given the graph A and location index i, we could get the embedding stack of i-th location and its $i_{\mathcal{N}}$ neighbors in l-th layer as $\boldsymbol{H}_{i,t}^{(l)} = \{\boldsymbol{h}_{i,t}^{(l)}, \boldsymbol{h}_{i_1,t}^{(l)}; \boldsymbol{h}_{i_2,t}^{(l)}; \ldots; \boldsymbol{h}_{i_{\mathcal{N}},t}^{(l)}\} \in \mathbb{R}^{(i_{\mathcal{N}}+1) \times d_l}$. Instead of only imposing the spatial neighbors from the graph A, we also utilize the temporal neighbors to construct the graph-based information gate $\boldsymbol{s}_{i,t\star\mathcal{G}}^{(l)}$ as Equation (9):

$$\boldsymbol{s}_{i,t\star\mathcal{G}}^{(l)} = \sigma\Big(\Theta_{\star\mathcal{G}}\big(\boldsymbol{H}_{i,t}^{(l-1)} \| \boldsymbol{H}_{i,t-V}^{(l)} \| \ldots \| \boldsymbol{H}_{i,t-1}^{(l)}, \boldsymbol{A}\big)\Big)\boldsymbol{W}_h^{(l)} + \boldsymbol{b}_h^{(l)}, \tag{9}$$

where $\|$ is the concatenation operation, $\Theta_{\star\mathcal{G}}$ is the graph convolution function with graph matrix A, σ represents an activation function(e.g. Leaky_ReLU), $V \leq K$ is the number of incurred time steps, $\boldsymbol{W}_h^{(l)} \in \mathbb{R}^{d_{gl} \times d_l}$ and $\boldsymbol{b}_h^{(l)} \in \mathbb{R}^{d_l}$ are the learned parameters to project the graph representations from d_{gl}-dimension into d_l-dimension space.

Now, we have the hidden state of i-th location at t-th time step in l-th layer as Equation (10):

$$\boldsymbol{h}_{i,t}^{(l)} = \boldsymbol{s}_{i,t\star\mathcal{G}}^{(l)} + \boldsymbol{g}_{i,t}^{(l)}. \tag{10}$$

4.4 Loss Function

Since GMSDR is a general framework for spatial temporal forecasting, we can use different graph convolutional operations and loss functions for different applications in our work. For the traffic flow prediction task, we utilize the graph convolutional operations in [20]. Given the label set Y and predicted values $\hat{Y}$, the objective for this task is to minimize the Mean Absolute Error (MAE) between the predicted and observed traffic speed as Equation (11):

$$\mathbf{MAE}(Y, \hat{Y}) = \frac{1}{|Y|}\sum_{i=1}^{|Y|}|Y_i - \hat{Y}_i|. \tag{11}$$

For the traffic demand prediction task, we combine the graph convolutional operation in [33] with our sequential model by our graph strategy. The loss function is set as Root Mean Square Error (RMSE) between the predicted and observed transportation demand as

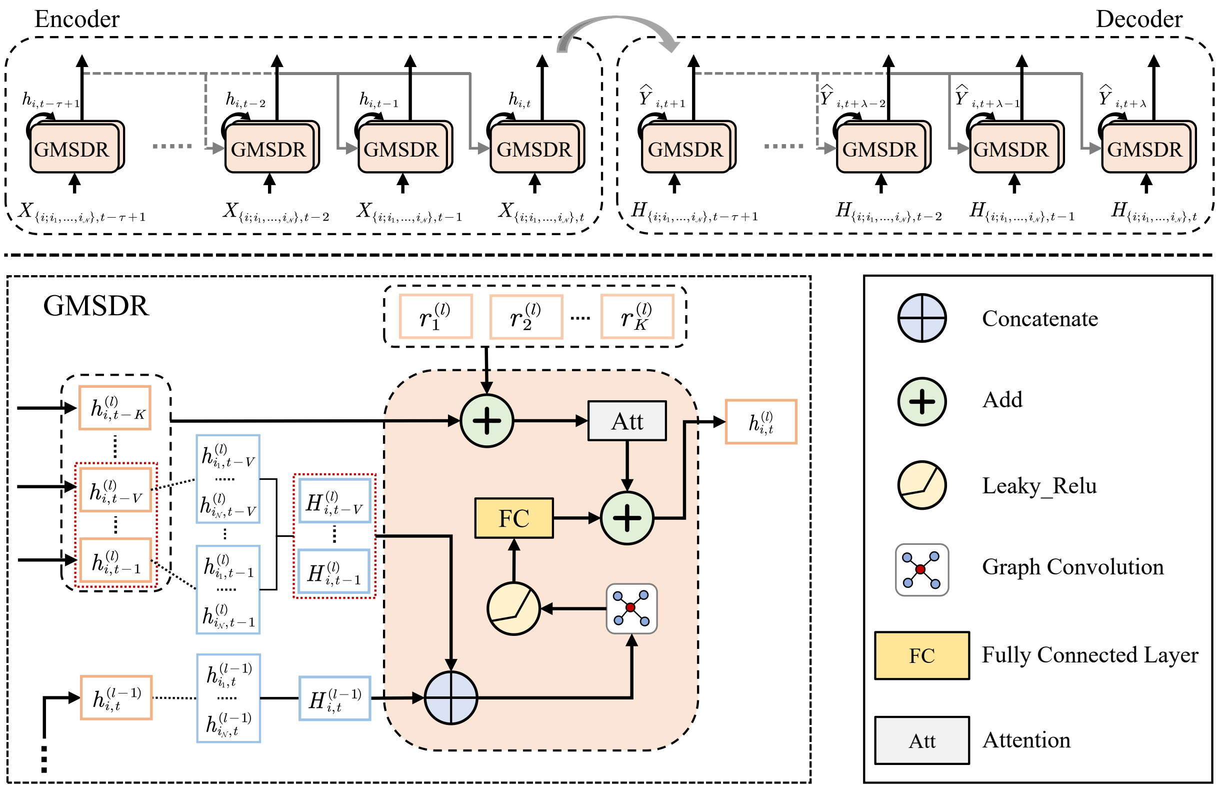


Figure 4: Overall Framework

Equation (12):

$$\mathbf{RMSE}(Y, \hat{Y}) = \sqrt{\frac{1}{|Y|}\sum_{i=1}^{|Y|}(Y_i - \hat{Y}_i)^2}. \tag{12}$$

5 EVALUATION

5.1 Experiment Setup

We evaluate our proposed GMSDR framework on two spatial temporal forecasting tasks: traffic flow prediction and traffic demand prediction to illustrate the generality of our work. The implementation of our method could be found in the link [1].

5.1.1 Datasets. For the task of traffic flow prediction, we evaluate on two popular datasets and utilize a trend-based approach for model training. PEMS03, PEMS08 are two datasets constructed from 2 districts in California, collected by the Caltrans Performance Measurement System(PeMS) in real time. The records are aggregated into 5-minute interval, so there are 12 points in the traffic flow for one day. PEMS03 contains data from 358 sensors and we select 3 months of data ranging from Sep. 1 2018 to Nov. 30 2018. PEMS08 contains data from 170 sensors and we select 2 months of data ranging from July 1 2016 to Aug. 31 2016.

For the task of traffic demand prediction, we conduct the evaluation on the NYC OpenData. There are two datasets including the order records of taxi and bike in NYC: NYC Citi Bike [2] includes the NYC Citi bike orders of people daily using. We use the 2.6 million records from April 1st, 2016 to June 30th, 2016. Each record consists of the following attributes: bike pickup station, bike drop-off station, bike pick-up time, bike drop-off time and trip duration. NYC Taxi [3] consists of 35 million taxicab trip records in New York from April 1st, 2016 to June 30th, 2016. Each record contains the attributes: pick-up time, drop-off time, pick-up longitude, pick-up latitude, drop-off longitude, drop-off latitude and trip distance. For the other settings such as hyper-parameters, length of time steps and the number of stations, we strictly follow the instructions mentioned in the previous study [33] to process the datasets.

5.1.2 Baseline Methods. We compare our proposed framework with the following baseline methods for traffic demand prediction:

- HA is the history average value, which is the average of historical values at previous time steps.
- FC-LSTM [27] is a variant of RNN using fully connected LSTM hidden units.
- DCRNN [20] integrates graph convolution into an encoder-decoder gated recurrent unit framework.
- STGCN [34] is the approach that employs graph convolution and 1D convolution to capture spatial dependencies and temporal correlations, respectively.

[1]https://github.com/dcliu99/MSDR

[2]https://www.citibikenyc.com/system-data

[3]https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

Table 2: Main Results of Traffic Demand Prediction

Method	NYC Citi Bike			NYC Taxi		
	RMSE	MAE	PCC	RMSE	MAE	PCC
HA	5.2003	3.4617	0.1669	29.7806	16.1509	0.6339
FC-LSTM	3.8139	2.3026	0.5675	18.0708	10.2200	0.8645
DCRNN	3.2094	1.8954	0.7227	14.7926	8.4274	0.9122
STGCN	3.6042	2.7605	0.7316	22.6489	18.4551	0.9156
STG2Seq	3.9843	2.4976	0.5152	18.0450	9.9415	0.8650
GraphWaveNet	3.2943	1.9911	0.7003	13.0729	8.1037	0.9322
CCRNN	2.8382	1.7404	0.7934	9.5631	5.4979	0.9648
GMSDR	**2.7218**	**1.6760**	**0.8107**	**8.6533**	**4.9831**	**0.9711**

Table 3: Main Results of Traffic Flow Prediction

Method	PEMS03			PEMS08		
	MAE	MAPE(%)	RMSE	MAE	MAPE(%)	RMSE
FC-LSTM	21.33 ± 0.24	23.33 ± 4.23	35.11 ± 0.50	22.20 ± 0.18	14.20 ± 0.59	34.06 ± 0.32
DCRNN	18.18 ± 0.15	18.91 ± 0.82	30.31 ± 0.25	17.86 ± 0.03	11.45 ± 0.03	27.83 ± 0.05
STGCN	17.49 ± 0.46	17.15 ± 0.45	30.12 ± 0.70	18.02 ± 0.14	11.40 ± 0.10	27.83 ± 0.20
ASTGCN(r)	17.69 ± 1.43	19.40 ± 2.24	29.66 ± 1.68	18.61 ± 0.40	13.08 ± 1.00	28.16 ± 0.48
GraphWaveNet	19.85 ± 0.03	19.31 ± 0.49	32.94 ± 0.18	19.13 ± 0.08	12.68 ± 0.57	31.05 ± 0.07
STSGCN	17.48 ± 0.15	16.78 ± 0.20	29.21 ± 0.56	17.13 ± 0.09	10.96 ± 0.07	26.80 ± 0.18
STFGNN	16.77 ± 0.09	16.30 ± 0.09	28.34 ± 0.46	16.64 ± 0.09	10.60 ± 0.06	26.22 ± 0.15
GMSDR	**15.78 ± 0.10**	**15.33 ± 0.11**	**26.82 ± 0.08**	**16.36 ± 0.07**	**10.28 ± 0.08**	**25.58 ± 0.12**

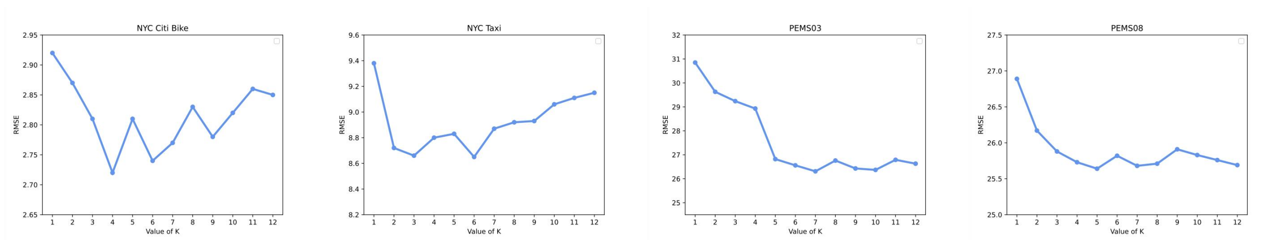


Figure 5: Parameter Analysis: Varying K

- STG2Seq [1] utilizes multiple localized spatial-temporal subgraph modules to synchronously capture the localized spatial-temporal correlations.
- GraphWaveNet [29] is a GNN based approach that combines adaptive graph convolution with dilated casual convolution to learn spatial-temporal dependency.
- CCRNN [33] is the state-of-the-art approach for traffic demand prediction. It uses different adjacent matrices in different layers of the GNN and proposes a layer-wise coupling mechanism to bridge different layers.

For the traffic flow prediction task, we compared with above methods FC-LSTM, STGCN, DCRNN, GraphWaveNet as well as the following methods:

- ASTGCN(r) [8] is an attention based spatial temporal graph convolutional network, which introduces spatial and temporal attention mechanisms to model three temporal properties of traffic flows. In order to ensure fair comparison, we only take its recent components.
- STSGCN [26] named Spatial-Temporal Synchronous Graph Convolutional Networks, which utilizes an elaborately designed spatial-temporal synchronous modeling mechanism to capture localized spatial-temporal correlations.
- STFGNN [19] is a CNN-based framework for traffic flow forecasting, which could handle long sequences by learning more spatial-temporal dependencies with layers stacked.

5.1.3 Evaluation Metrics. For both tasks, we choose root mean squared errors(RMSE) and mean absolute errors(MAE) as the evaluation metrics. Based on the difference in problem definition, we choose different third evaluation metrics for the two tasks [20, 33]: For traffic demand prediction we choose Pearson Correlation Coefficient (PCC), which is suitable to describe the co-evolving trends of two groups of data; For traffic flow prediction we choose Mean Absolute Percentage Error (MAPE), which is suitable to describe the difference between predicted value and true value.

Table 4: Different Strategies on Traffic Demand Prediction

	Metric	E-Spatial	D-Spatial	Simple
Bike	RMSE	**2.722**	2.838	2.961
	MAE	**1.677**	1.740	1.832
	PCC	**0.811**	0.793	0.761
Taxi	RMSE	**8.653**	8.932	9.561
	MAE	**4.983**	5.141	5.498
	PCC	**0.971**	0.965	0.943

Table 5: Different Strategies on Traffic Flow Prediction

	Metric	E-Spatial	D-Spatial	Simple
PEMS03	MAE	**15.78**	16.69	17.72
	MAPE(%)	**15.33**	15.81	16.63
	RMSE	**26.82**	28.52	29.36
PEMS08	MAE	**16.36**	17.54	18.33
	MAPE(%)	**10.28**	11.47	12.24
	RMSE	**25.58**	27.45	28.52

5.1.4 Hyper-parameters. We use grid search to decide the best hyper-parameters as following: For the number of previous time steps K, we select it from the range [1,12]. The dimension of hidden state d_l is chosen from the range $\{32, 64, 128\}$. Once K is fixed, we select V from the range [1,K]. We use Adam [17] as the optimizer in the training process. The learning rate for Adam is selected from the range $\{0.0005, 0.001, 0.0015\}$ and the batch size is fixed as 64. The number of epochs is set as 200 with early stop strategy.

5.2 Main Results

The results of traffic demand prediction are shown in Table 2. We can see that GMSDR outperforms all baseline methods by an obvious margin. The reason is that in the task of traffic demand prediction, it is essential to capture long-term periodical patterns. By explicitly involving more historical hidden states in each time step, it is easier for GMSDR to learn such information. Meanwhile, the other RNN based approaches such as CCRNN, DCRNN and STG2Seq can only make use of one previous hidden state where certain long-term dependency information might already be discarded. As a result, the performance of them is limited by their inherited design.

Then we look at the results of traffic flow prediction. As shown in Table 3, GMSDR also achieves the best performance among all settings. Compared with the RNN based approaches like DCRNN and FC-LSTM, GMSDR can make better capture the temporal dynamics with the help of historical hidden states. The other GNN based methods tried to learn the spatial temporal correlations with convolutional operations over the graph constructed by their proposed techniques and avoided using RNN models. However, it is rather difficult to find a perfect solution to involve temporal dynamics in such a graph and thus the overall performance is worse than our method.

5.3 Detailed Analysis

Next we perform more detailed analysis to demonstrate the effectiveness of our proposed methods. First, we evaluate different strategies of learning temporal interactions over the hidden states from previous time steps and the current input used in MSDR. We implement 3 methods: Simple is the naive method that directly use a global learned embedding for the dependency relation (Equation (2)); D-Spatial is the approach that utilizes implicit spatial dependency from the input data (Equation (7)); E-Spatial is the approach that applies explicit spatial dependencies (Equation (8)).

The results of traffic demand prediction are shown in Table 4. We can see that on both datasets Bike (NYC Citi Bike) and Taxi (NYC Taxi), E-Spatial achieves the best performance. The reason is that E-Spatial can explicitly introduce the spatial information and thus makes it easier to capture the spatial temporal correlations. Meanwhile, D-Spatial performs slightly better than Simple under most settings. The reason is that it involves richer information in the interactions which can help introduce more hints for dependency. The results of traffic flow prediction in Table 5 illustrate the similar trends. Such results demonstrate the generality of our proposed E-Spatial approach in modeling the spatial temporal dependencies for different applications.

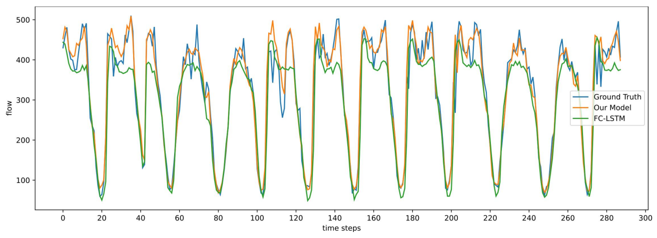
Since the core contribution of our proposed MSDR model is to introduce multiple historical hidden states, it is essential to make a parameter sensitivity analysis over the number of involved historical time steps K. The results of RMSE on both tasks are shown in Figure 5. For the task of traffic demand prediction, we can see that the best results are achieved when K is 4 and 6 on the NYC Citi Bike and NYC Taxi dataset, respectively. The reason might be that the traffic demand prediction task aimed at learning periodical patterns within a relatively long time range. As a result, it is rather helpful to involve hidden states from more recent time steps. When the value of K is 1, it has the worst performance since in this case MSDR degenerated to the original GRU model. For the task of traffic flow prediction,we can see that when the value of K varies from 5 to 12, the results stay relatively stable. This is because due to the problem setting of traffic flow prediction, the historical information is not as important as that in demand prediction. Nevertheless, it is still helpful to introduce hidden states from multiple historical time steps.

5.4 Case Study

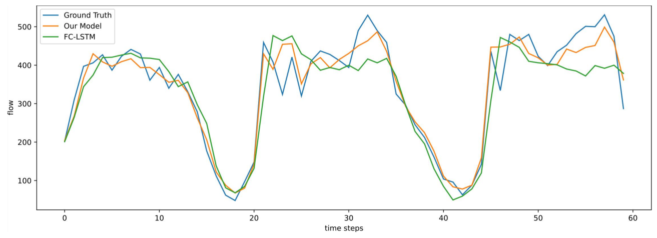
To show the superiority and stability of our framework, we give the case study on traffic flow prediction tasks in Figure 6. As shown in Figure 6(a), GMSDR produces smooth prediction of the mean when small oscillation exists in the ground truth. Meanwhile, Figure 6(b) illustrates the results of traffic flow prediction with smooth data. We can see that GMSDR is able to accurately predict the start and end time of peak hours. From the results, we could find that our framework could fit ground truth well and performs much better than the FC-LSTM model. Specifically, when faced with urgent changes of the multiple peak data, the better results prove that our framework can overcome the overfitting problems by explicitly capturing the dependency.

6 CONCLUSION

In this paper, we studied the problem of spatial temporal forecasting. We recognized the importance of learning from multiple time steps in capturing the long-range spatial temporal dependencies.



(a) Traffic Flow with Multiple Peaks



(b) Traffic Flow with Smooth Data

Figure 6: Case Study

Then we developed a novel RNN variant MSDR to make use of multiple historical hidden states and the current input. We investigated several techniques for temporal interaction between hidden states in historical time steps and current input and provided a highly effective dependency relation operation as the solution. Then we proposed the GMSDR framework by combing MSDR with GNN modules into the encoder-decoder architecture. Experimental results on several popular datasets demonstrated the superiority of our proposed framework.

ACKNOWLEDGMENTS

This work was supported by the NSFC (U2001212, 62032001, and 61932004).

REFERENCES

[1] L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Q. Z. Sheng. Stg2seq: Spatial-temporal graph to sequence model for multi-step passenger demand forecasting. In *IJCAI*, pages 1981–1987, 2019.
[2] D. Cao, Y. Wang, J. Duan, C. Zhang, X. Zhu, C. Huang, Y. Tong, B. Xu, J. Bai, J. Tong, and Q. Zhang. Spectral temporal graph neural network for multivariate time-series forecasting. In *NeurIPS*, 2020.
[3] W. Chen, L. Chen, Y. Xie, W. Cao, Y. Gao, and X. Feng. Multi-range attentive bicomponent graph convolutional network for traffic forecasting. In *AAAI*, pages 3529–3536, 2020.
[4] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, 2014.
[5] Z. Cui, R. Ke, and Y. Wang. Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction. *CoRR*, abs/1801.02143, 2018.
[6] Z. Fang, Q. Long, G. Song, and K. Xie. Spatial-temporal graph ODE networks for traffic flow forecasting. In *ACM SIGKDD*, pages 364–373, 2021.
[7] X. Geng, Y. Li, L. Wang, L. Zhang, Q. Yang, J. Ye, and Y. Liu. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *AAAI*, pages 3656–3663, 2019.
[8] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 922–929, 2019.
[9] W. L. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *NIPS*, pages 1024–1034, 2017.
[10] L. Han, B. Du, L. Sun, Y. Fu, Y. Lv, and H. Xiong. Dynamic and multi-faceted spatio-temporal deep learning for traffic speed forecasting. In *ACM SIGKDD*, pages 547–555, 2021.
[11] P. Han, J. Wang, D. Yao, S. Shang, and X. Zhang. A graph-based approach for trajectory similarity computation in spatial networks. In *ACM SIGKDD*, pages 556–564, 2021.
[12] P. Han, P. Yang, P. Zhao, S. Shang, Y. Liu, J. Zhou, X. Gao, and P. Kalnis. Gcn-mf: disease-gene association identification by graph convolutional networks and matrix factorization. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 705–713, 2019.
[13] P. Han, P. Zhao, C. Lu, J. Huang, J. Wu, S. Shang, B. Yao, and X. Zhang. Gnn-retro: Retrosynthetic planning with graph neural networks. In *AAAI*, 2022.
[14] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997.
[15] Y. Jeong, Y. Byon, M. M. Castro-Neto, and S. M. Easa. Supervised weighting-online learning algorithm for short-term traffic flow prediction. *IEEE Trans. Intell. Transp. Syst.*, 14(4):1700–1707, 2013.
[16] N. Jones. How machine learning could help to improve climate forecasts. *Nature*, 548(7668), 2017.
[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.
[18] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.
[19] M. Li and Z. Zhu. Spatial-temporal fusion graph neural networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4189–4196, 2021.
[20] Y. Li, R. Yu, C. Shahabi, and Y. Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *ICLR*, 2018.
[21] M. Lippi, M. Bertini, and P. Frasconi. Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning. *IEEE Trans. Intell. Transp. Syst.*, 14(2):871–882, 2013.
[22] J. Liu, L. Sun, W. Chen, and H. Xiong. Rebalancing bike sharing systems: A multi-source data smart optimization. In *ACM SIGKDD*, pages 1005–1014, 2016.
[23] Y. Liu, X. Ao, L. Dong, C. Zhang, J. Wang, and Q. He. Spatiotemporal activity modeling via hierarchical cross-modal embedding. *IEEE Trans. Knowl. Data Eng.*, 34(1):462–474, 2022.
[24] A. Longo, M. Zappatore, M. A. Bochicchio, and S. B. Navathe. Crowd-sourced data collection for urban monitoring via mobile sensors. *ACM Trans. Internet Techn.*, 18(1):5:1–5:21, 2017.
[25] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas. Predicting taxi-passenger demand using streaming data. *IEEE Trans. Intell. Transp. Syst.*, 14(3):1393–1402, 2013.
[26] C. Song, Y. Lin, S. Guo, and H. Wan. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 914–921, 2020.
[27] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.
[28] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *ACM SIGKDD*, pages 753–763, 2020.
[29] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang. Graph wavenet for deep spatial-temporal graph modeling. In *IJCAI*, pages 1907–1913, 2019.
[30] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In *AAAI*, pages 5668–5675, 2019.
[31] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li. Deep multi-view spatial-temporal network for taxi demand prediction. In *AAAI*, pages 2588–2595, 2018.
[32] J. Ye, L. Sun, B. Du, Y. Fu, X. Tong, and H. Xiong. Co-prediction of multiple transportation demands based on deep spatio-temporal neural network. In *ACM SIGKDD*, pages 305–313, 2019.
[33] J. Ye, L. Sun, B. Du, Y. Fu, and H. Xiong. Coupled layer-wise graph convolution for transportation demand prediction. In *AAAI*, pages 4617–4625, 2021.
[34] B. Yu, H. Yin, and Z. Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *IJCAI*, pages 3634–3640, 2018.
[35] J. Zhang, Y. Zheng, and D. Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *AAAI*, pages 1655–1661, 2017.
[36] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li. T-GCN: A temporal graph convolutional network for traffic prediction. *IEEE Trans. Intell. Transp. Syst.*, 21(9):3848–3858, 2020.
[37] C. Zheng, X. Fan, C. Wang, and J. Qi. GMAN: A graph multi-attention network for traffic prediction. In *AAAI*, pages 1234–1241, 2020.