Gated Residual Recurrent Graph Neural Networks for Traffic Prediction

Cen Chen,[⋆] Kenli Li,[⋆∗] Sin G. Teo,[†] Xiaofeng Zou,[⋆] Kang Wang[⋆]
Jie Wang,[†] Zeng Zeng[†∗]
[⋆]College of Information Science and Engineering, Hunan University, China
[†]Institute for Infocomm Research, Singapore
{chencen, lkl, zouxiaofeng, zztwk}@hnu.edu.cn, {teosg, wangjie, zengz}@i2r.a-star.edu.sg

Abstract

Traffic prediction is of great importance to traffic management and public safety, and very challenging as it is affected by many complex factors, such as spatial dependency of complicated road networks and temporal dynamics, and many more. The factors make traffic prediction a challenging task due to the uncertainty and complexity of traffic states. In the literature, many research works have applied deep learning methods on traffic prediction problems combining convolutional neural networks (CNNs) with recurrent neural networks (RNNs), which CNNs are utilized for spatial dependency and RNNs for temporal dynamics. However, such combinations cannot capture the connectivity and globality of traffic networks. In this paper, we first propose to adopt residual recurrent graph neural networks (Res-RGNN) that can capture graph-based spatial dependencies and temporal dynamics jointly. Due to gradient vanishing, RNNs are hard to capture periodic temporal correlations. Hence, we further propose a novel hop scheme into Res-RGNN to utilize the periodic temporal dependencies. Based on Res-RGNN and hop Res-RGNN, we finally propose a novel end-to-end multiple Res-RGNNs framework, referred to as “MRes-RGNN”, for traffic prediction. Experimental results on two traffic datasets have demonstrated that the proposed MRes-RGNN outperforms state-of-the-art methods significantly.

Introduction

Traffic prediction is one of the most challenging tasks in Intelligent Transportation Systems (ITS) (Jabbarpour et al. 2018). This task is important and critical for many transportation services, such as vehicle flow control, road trip planning and navigation and so on. The goal of traffic prediction is to predict future traffic states of road networks using sequential traffic historical states. Three key complex factors that can affect traffic conditions are investigated as follows.

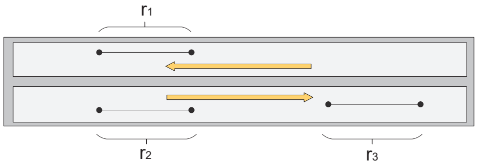


Figure 1: Three roads in the directed road network

- **Factor 1: Spatial dependencies on a directed road network.** Spatial correlation such as geographic distance and connectivity in directed road network can affect traffic prediction performance. For example, given three roads r_1, r_2 and r_3 in the directed road network as depicted in Figure 1, Roads r_2 and r_3 are obviously correlated, while roads r_2 and r_1 are not. Even r_2 is closer in the distance to r_1 than r_3, traffic conditions of r_2 have greater effect on r_3, rather than on r_1.
- **Factor 2: Multiple temporal dependencies.** Traffic conditions usually involve a mixture of the repeating time patterns. For example, the traffic jam of a road occurring at 6 pm will affect the same road’s traffic condition in the following hour, i.e., 7 pm. We also utilize the patterns such as on daily and weekly basis to predict traffic conditions of the road, e.g., using peak hour patterns for traffic prediction.
- **Factor 3: External factor.** Traffic conditions can be significantly affected by external factors such as vehicle accidents, road maintenance, weather conditions, holidays and other special events, and so on.

Many statistical approaches have been widely used in predicting traffic conditions. In recent years, deep learning based methods outperform many traditional statistical approaches (e.g., k-nearest neighbours and support vector machines) in various prediction tasks, including traffic prediction. Convolutional neural networks (CNN) (Zeng et al. 2018) have been proven be good spatial feature extractors. Many methods (Zhang, Zheng, and Qi 2017; Du et al. 2018; Yu et al. 2017; Yao et al. 2018b; Chen et al. 2018) that have combined CNNs with long short-term memory (LSTM) networks to capture spatial-temporal features from traffic data have outperformed many traditional machine learning methods (Hong 2011; Xia et al. 2016).

However, one of the main limitations of the above combination is that normal convolutional operations can capture the spatial features of regular grid structures existing in images or videos, rather than the features of general graph forms. The traffic network is a non-Euclidean and directional topology that makes the convolution operation less

[*] They are corresponding author of this paper.

effective. (Li et al. 2018; Yu, Yin, and Zhu 2017) have proposed different networks to fully utilize spatial information on the traffic networks. To the best of our knowledge, (Li et al. 2018; Yu, Yin, and Zhu 2017) have combined GCNs and some other neural networks to achieve the state-of-the-art results on traffic prediction. However, some limitations are still in these works, as described in the following: (i) It is hard to train deep conventional RNNs well, due to the vanishing gradient and exploding gradient problems (He et al. 2016a; He et al. 2016b), which can cause serious training issues when graph convolutions become complicated, (ii) The models are less sensitive to large linear scale changes of inputs, due to the non-linear nature of the convolutional and recurrent operations and (iii) They only consider near past traffic conditions, without capturing the existing periods and repeating patterns.

In this paper, we propose a novel framework for traffic prediction, referred to as "MRes-RGNN", based on multiple residual recurrent graph neural networks. Through rigorous experiments, we have demonstrated the performance of MRes-RGNN and proven the advantages of the proposed MRes-RGNN over other state-of-the-art methods in traffic prediction. Our contributions can be summarized as follows.

- We propose to utilize residual recurrent graph neural networks (Res-RGNN) to capture the graph-based spatial dependencies and temporal dynamics jointly. Res-RGNN can properly extract spatio-temporal features for traffic networks, and make the models more sensitive to unexpected changes.
- We design a novel hop Res-RGNN to improve the performance of traffic prediction that can discover the periodic patterns among the time series signals.
- By combining the outputs of Res-RGNN and hop Res-RGNN branches with dynamic weights, we propose a novel end-to-end framework, named as MRes-RGNN, that considers spatial and multiple temporal dependencies with external factors for traffic prediction.

Related Work

In this section, we discuss some deep learning methods applied in traffic forecasting that outperform many traditional statistical methods (Hong 2011; Xia et al. 2016). CNN or its related convolutional-based residual network can extract the spatial dependencies of the traffic networks by converting the dynamic traffic data into images (Ma et al. 2017). This neural network architecture only capture spatial-temporal information or correlation of the traffic flow data, separately. (Zhang et al. 2018; Zhang, Zheng, and Qi 2017; Yao et al. 2018b; Yao et al. 2018a) proposed to combine of both RNN and CNN networks that can learn spatio-temporal information simultaneously to overcome the limitation as discussed previously. To further improve accuracy performance in traffic forecasting, (Yu et al. 2017; Yao et al. 2018b; Du et al. 2018) proposed different networks as discussed in the following.

(Yu et al. 2017) proposed a deep LSTM model using the normal traffic hours and a mixture of deep LSTM model using the incident traffic period. (Yao et al. 2018b) proposed a multi-view spatio-temporal network that combines local CNN, LSTM and semantic network to predict short-term traffic conditions. Lastly, (Du et al. 2018) proposed a hybrid multi-modal deep learning framework based on multiple CNN-GRU algorithms, which can effectively extract local spatial features and long dependency features together with spatio-temporal correlations from the multi-modal traffic data. However, the discussed approaches cannot capture the spatial features of traffic networks.

To overcome the limitations above, (Defferrard, Bresson, and Vandergheynst 2016; Yu, Yin, and Zhu 2017; Li et al. 2018) proposed different networks in traffic forecasting. (Defferrard, Bresson, and Vandergheynst 2016) proposed the graph convolutional neural networks (GCN) to capture the non-Euclidean spatial features of traffic data. (Yu, Yin, and Zhu 2017) proposed a traffic forecasting framework that uses GCN to learn spatio-temporal features of traffic data applicable only to undirected graph. (Li et al. 2018) also proposed a traffic forecasting framework to combine both of the diffusion convolutional and recurrent neural network together in forecasting traffic conditions. These above the two frameworks even can capture the spatial dependency of the traffic data using bidirectional random walks on the graph, and the temporal dependency of the traffic data using the encoder-decoder network. However, they do not use more complex traffic features such as the multiple temporal dependencies and external factors.

Preliminaries

Traffic Prediction on Graphs

Traffic prediction is to perform prediction on future traffic states (i.e., traffic speed and traffic flow), as given historical traffic states from a series of road segments or observation sensors. In this section, we first define the traffic prediction problems in Definition 1.

Definition 1 **(Traffic Prediction):** Formally, given a series of fully observed time series signals $X = x_1, x_2, ..., x_t$ of all the road segments or sensors, the predicted future signals can be defined as $Y = y_{t+1}$. Since roads contain directional information, traffic networks can be modelled as directed graphs with structured time-series data.

In the following, we shall present the key definitions that are used for directed graph based traffic prediction.

Definition 2 (Directed Graph for Traffic Network): A traffic network can be modeled as a weighted directed graph, $G = (V, E, W)$, where V is a set of nodes, e.g., sensors, that can monitor road segments in the traffic network, E is the connectivity among the nodes, e.g., $\varepsilon_{(i,j)} = 1$ if υ_i and υ_j are connected, and $\varepsilon_{(i,j)} = 0$ if not, and $W \in R^{N \times N}$ is the weighted adjacency matrix of G representing the nodes' proximities, e.g., the distance between any pair of nodes, N is the number of nodes. Specifically, $w_(i, j)$ can be defined as the edge weight from υ_i to υ_j.

Definition 3 **(Traffic Prediction on Graphs):** Given a G, let $x_t \in R^{N \times P}$ be a graph signal observed at time t, where P is the number of features observed by the node, e.g., vehicle speed, flow, etc. The prediction of y_{t+1} can be formulated as follows:

$$y_{t+1} = f\left([x_{t-\tilde{h}+1}, ..., x_t], G\right). \tag{1}$$

Convolutions on Graphs

Conventional 2D CNNs that are used well for regular grids, such as images, cannot be applied directly to learn features from general graphs, as indicated in (Niepert, Ahmed, and Kutzkov 2016; Defferrard, Bresson, and Vandergheynst 2016; Bruna et al. 2013). In the literature, two main approaches have been proposed to generalize CNNs for graphs. The first one is to transfer the data into spectral domain using graph Fourier transforms (Bruna et al. 2013). Thus, the performance of graph convolution has been improved significantly, i.e., the computational complexity is reduced from $O(n^2)$ to linear (Defferrard, Bresson, and Vandergheynst 2016; Kipf and Welling 2016). However, this approach can only extract spatial features of undirected graphs. The other one is to extend CNNs to support general directed graph-structured data by using *diffusion convolution operation* (Atwood and Towsley 2016; Li et al. 2018; Teng 2016), as defined in the following:

Definition 4 **(Diffusion Convolution):** Diffusion convolution operation over a graph signal $x \in R^{N \times P}$ and a filter Θ is defined as:

$$x \star \Theta = \sum_{k=0}^{K-1} \left(\theta_{k,1}(D_O^{-1}W)^k + \theta_{k,2}(D_I^{-1}W^T)^k\right)x, \tag{2}$$

where $\star$ denotes the diffusion convolution, K is the number of diffusion steps, $\theta \in R^{K \times 2}$ are the learned parameters of Θ for two directions of the graph; $D_O = diag(W1)$ are the out-degree diagonal matrix, and $1 \in R^N$ denotes the all one vector; $D_I = diag(W^T)$ is the in-degree diagnose; and $D_O^{-1}W$, $D_I^{-1}W^T$ represent the transition matrices of the diffusion process and the reverse one, respectively.

In the case of the undirected graphs, (Li et al. 2018) has shown that many existing graph structured convolutional operations, including the spectral graph convolution (Bruna et al. 2013), are special cases of diffusion convolution.

Proposed MRes-RGNN Framework

The network architecture of the proposed MRes-RGNN is shown in Figure 2, where the inputs of MRes-RGNN are the historical traffic states and external features, while the outputs are the predictions on the future traffic states. Multiple temporal dependencies contain *near* and *periodic*, e.g., *daily*, *weekly*, and *monthly*, dependencies. Our framework consists of several MRES-RGNN modules with different hops. Obviously, Res-RGNN is a special case of MRES-RGNN with hop 1. The Res-RGNN branch is utilized to learn spatial information on graphs with the *near* dependency together, and multiple hop Res-RGNN branches are for spatial information with multiple *periodic* dependencies respectively. The extracted spatial and multiple temporal features are further fused with a weighted scheme and then, integrated by an output layer to achieve a final prediction. The entire end-to-end framework is trained by maximizing the likelihood of a graph signal using back-propagation through time (BPTT) (Werbos 1990; Sutskever, Vinyals, and Le 2014).

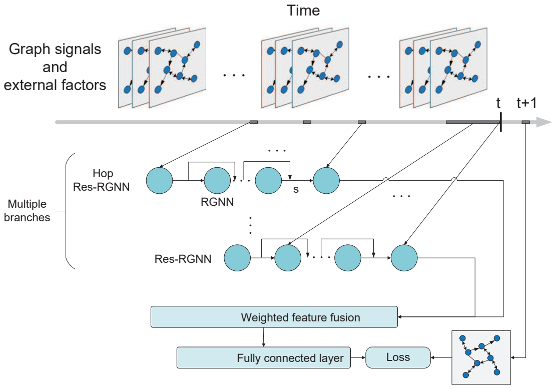


Figure 2: Network Architecture of MRes-RGNN.

Graph CNNs for Extracting Spatial Features

A traffic network can be modelled as a graph mathematically. However, in the previous studies (Yu et al. 2017; Yao et al. 2018b), which utilized CNNs to extract the spatial features without considering spatial attributes of traffic networks; as a result, the connectivity and globality of the networks were overlooked as roads were split into multiple segments or grids. In the paper, the traffic network is firstly modeled as a directed graph, and then the diffusion convolution (Atwood and Towsley 2016; Li et al. 2018; Teng 2016) is applied directly on graph-structured data to extract patterns and features in space domain. Based on the definition of the convolution operation, we have a new neural network layer, diffusion convolutional layer, that is defined in Definition 4 as follows.

Definition 5 **(Diffusion Convolutional Layer):** Based on the diffusion convolution defined in Definition 1, a diffusion convolutional layer that maps P-dimensional features to Q-dimensional outputs can be defined as follows. Let the parameter tensor be $\Theta \in R^{Q \times P \times K \times 2} = [\theta]_{q,p}$, where $\Theta_{q,p,:,:} \in R^{K \times 2}$ parameterizes the convolutional filter for the p-th input and the q-th output. Hence, the diffusion convolutional layer is:

$$H_{:,q} = \sigma\left(\sum_{p=1}^{P} x_{:,p} \star \Theta_{q,p,:,:}\right), q \in 1, ..., Q, \tag{3}$$

where $x \in R^{N \times P}$ is the input, $H \in R^{N \times Q}$ is the output, $\Theta_{q,p,:,:}$ are the filters and σ is an activation function (e.g., ReLU and Sigmoid functions).

This diffusion convolutional layer can be used to learn the representations of the graph structured data, which can be trained by using stochastic gradient descent (SGD).

Residual Recurrent Graph Neural Networks (Res-RGNN) for Extracting Spatio-Temporal Features

In MRes-RGNN, we utilize graph convolutional, recurrent, and residual neural networks together to extract the spatial-

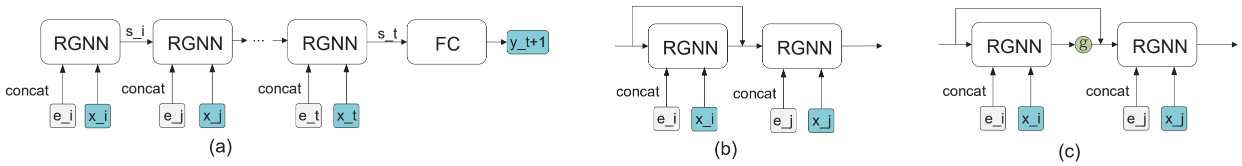


Figure 3: (a) Recurrent graph neural networks (RGNN) which combine diffusion convolution operation (a type of graph convolution operation); (b) RGNN with residual shortcuts (Res-RGNN); (c) RGNN with gated residual learning (Gated Res-RGNN).

temporal features with the external features under consideration.

RGNN + External Features Gated Recurrent Unit (GRU) is a simple, yet powerful variant of RNNs for time series prediction due to the gating mechanism (Chung et al. 2014). GRUs are carefully designed to memorize historical information, e.g., long-term dependencies. Inspired by (Chung et al. 2014), we combine graph convolution and GRU together, denoted as RGNN, to discover the spatial-temporal dynamics jointly as depicted in Figure 3(a). The matrix multiplications in GRU can be replaced with the graph convolutions that is similar to the methods used in (Seo et al. 2016; Li et al. 2018). This method combines the traditional feature engineering methods with deep learning based methods. The external factor attributes are first embedded and concatenated with graph signals and then, used as the input of RGNN model as shown in Figure 3. Thus, the process of a RGNN unit at time t can be formulated as:

$$\begin{aligned} r_t &= \sigma(\Theta_r \star [x_t, e_t, s_{t-1}] + b_r), \\ u_t &= \sigma(\Theta_u \star [x_t, e_t, s_{t-1}] + b_u), \\ c_t &= tanh(\Theta_c \star [x_t, e_t, (r_t \odot s_{t-1})] + b_c), \\ s_t &= u_t \odot s_{t-1} + (1 - u_t) \odot c_t, \\ y_{t+1} &= W_o s_t, \end{aligned} \tag{4}$$

where x_t, e_t and s_t are the graph signal, external feature and the hidden state output at time t, r_t and u_t denote the reset gate and update gate at the time t, respectively, $\star$ denotes the graph convolution, Θ_r, Θ_u and Θ_c are the learned parameters of filters, y_{t+1} is the output at $t+1$ and lastly W_o denotes the learned parameters of the output layer.

RGNN unit is similar to the diffusion convolutional layer. The RGNN unit can be used to build recurrent neural network layers and also trained by using BPTT. Like the deep RNN methods in (Pascanu et al. 2013; Du, Wang, and Wang 2015), hierarchical RGNN layers can be stacked to form the deep RGNN model.

Res-RGNN Residual neural network is used with a reference to the direct hidden state, instead of an unreferenced function. In (He et al. 2016a; He et al. 2016b), the authors introduced the residual-shortcut structures to build deep networks with a depth of 152 layers. The networks have achieved good performance on COCO and ImageNet object detection datasets (He et al. 2016a). The residual learning and linear shortcut connections can help to solve the exploding and vanishing gradient problems in the long-term backpropagation (He et al. 2016a; He et al. 2016b), especially in the networks with deep structures.

With the benefits of (He et al. 2016a; He et al. 2016b), we introduce the residual error into RGNN. A RGNN cell is considered as a computation block where the residual information is passed by shortcut of the blocks so as to speed up a convergence rate, as illustrated in Figure 3(b). Hence, the hidden state of t after adding the residual shortcut path can be formulated as follows:

$$\tilde{s}_t = \phi(\tilde{s}_{t-1}, x_t, e_t, \Theta_r, \Theta_u, \Theta_c) + W_{linear}\tilde{s}_{t-1}, \tag{5}$$

where ϕ denotes the function of GRU cell, $\tilde{s}_t$ denotes the state of t after the residual shortcut path added and W_{linear} is the linear projection weight.

Equation 5 is composed of a shortcut connection and an element-wise addition. Importantly, the shortcut connection has not added extra parameters and increased computation complexity. This reconstruction can make the loss function approximate to an identity mapping. Thus, the training errors are not increasing since the recurrent connections are formulated as the identity mapping.

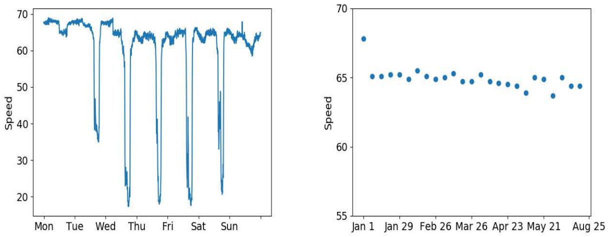
It is obvious that the previous hidden states of RGNN units are added to the hidden states of the next RGNN units without non-linear activation in Equation 5. This linear additive nature makes RGNN more sensitive and robust to sudden changes in traffic historical states.

Gated Res-RGNN Motivated by the gate function in GRU (Chung et al. 2014), we also add a gate function to control the flow of residual information as depicted in Figure 3(c) in the proposed MRes-RGNN. This gating mechanism helps to make decision on the threshold that how much the previous residual information can affect next time segments. The hidden state at time t with the gated residual shortcut path can be formulated as follows:

$$\begin{aligned} \tilde{s}_t &= g\phi(\tilde{s}_{t-1}, x_t, e_t, \Theta_r, \Theta_u, \Theta_c) + W_{linear}\tilde{s}_{t-1}, \\ g &= \sigma W_g[x_t, e_t] + U_g s_{t-1}, \end{aligned} \tag{6}$$

where W_g is the linear projection weight, U_g is the state-to-state weight matrix and σ is the activation function.

Hop Res-RGNN for Very Long-term Dependencies One of the common ways to predict near future traffic condition is to use latest time segments. Besides, the periodic patterns can also be utilized to improve the traffic prediction performance. In Figure 4(a), we describe the speed value during each time interval in 7 days plotting by a sensor in METR-LA dataset. The plotting curves show a certain repeatability pattern. Therefore, we can easily identify the daily patterns in the given dataset. Moreover, we plot Figure 4(b) to illustrate the weekly periodic patterns of the traffic



(a) (b)

Figure 4: Periodic temporal dependencies for METR-LA. (a) daily; (b) weekly.

data. The two figures have clearly shown that the temporal dependencies of periods give significant impacts on the traffic state; e.g., *near* time, *daily*, and *weekly* periodic patterns regardless of the degrees of influences which are not completely the same.

As the recurrent layers with GRU and LSTM units are carefully designed to memorize the historical information, relatively long-term dependencies is aware by the layers. Both GRU and LSTM cannot capture very long-term (*daily*, *weekly*) correlation in practice due to the notorious gradient vanishing problem (He et al. 2016a). To solve this issue, we propose a novel hop scheme in our MRes-RGNN framework which leverages the periodic patterns in traffic data. Specifically, in our model, hop-links are added to connect the current RGNN cell and the RGNN cells in a same phase of adjacent periods. Suppose we want to predict the traffic state at $t+1$, the updating process formula of the last RGNN cell for a specific hop Res-RGNN branch without any gate can be formulated as follows:

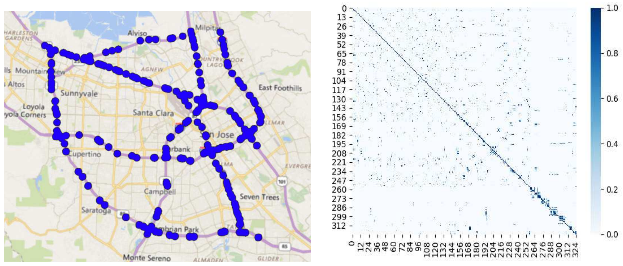
$$\begin{aligned} r_{t+1-\beta} &= \sigma(\Theta_r \star [x_{(t+1)-\beta}, e_{(t+1)-\beta}, \tilde{s}_{(t+1)-\beta\times 2}] + b_r), \\ u_{t+1-\beta} &= \sigma(\Theta_u \star [x_{(t+1)-\beta}, e_{(t+1)-\beta}, \tilde{s}_{(t+1)-\beta\times 2}] + b_u), \\ c_{t+1-\beta} &= tanh(\Theta_c \star [x_{(t+1)-\beta}, e_{(t+1)-\beta}, (r_t \odot \tilde{s}_{(t+1)-\beta\times 2})] \\ &\quad + b_c), \\ \tilde{s}_{t+1-\beta} &= u_t \odot s_{(t+1)-\beta\times 2} + (1-u_t) \odot c_t \\ &\qquad + W_{linear}\tilde{s}_{(t+1)-\beta\times 2}, \end{aligned} \tag{7}$$

where β is the number of hidden cells skipped through and $\odot$ is the element-wise multiplication for tensors. β can be calculated by periods (*daily* or *weekly*).

Weighted Fusion for Multiple Res-RGNN Branches

As discussed previously, traffic states are affected by multiple temporal dependencies (e.g., *near*, *daily* and *weekly*). The degrees of influence on the traffic states may be different. In our MRes-RGNN framework, a branch of Res-RGNN targets at *near* dependency, while multiple hop Res-RGNNs target at multiple periodic (*daily* and *weekly*) dependencies. Inspired by the observations, we propose a novel parametric-tensor-based fusion method that can fuse multiple branches in the framework as well.

The fusion method of the hidden state at time t can be formulated as follows:



(a) (b)

Figure 5: (a) Sensor networks of PEMS-BAY, each dot denotes a sensor station. (b) Heat map of weighted adjacency matrix.

$$s_f = W_c \otimes s_c + \sum_{i=1}^{\gamma} W_i \otimes s_i, \tag{8}$$

where s_f denotes the fused hidden state features, $\otimes$ is Hadamard product (i.e., element-wise multiplication for tensors), s_c are the hidden state features of the Res-RGNN branch, γ denotes the number of hop Res-RGNN branches, s_i means the hidden state features of the hop Res-RGNN branch with the index i, and lastly $W_c, W_1, ..., W_\gamma$ are the learnable parameters that adjust the degrees affected by different branches.

Once the features of the two branches are fused by the proposed weighted strategy, the merged features s_f can be used as the input of a dense layer to obtain the output of the MRes-RGNN framework.

Experiments

In this section, we evaluate the performance of our proposed MRes-RGNN compared with other existing methods in traffic prediction.

Experiment Settings

We configure a Linux server and the other configurations to run the experiment as follows: 8 Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHZ; 256GB RAM; 4 NVIDIA P100 GPUs. Two large real-world datasets, i.e., METR-LA and PEMS-BAY datasets, are used in the experiment:

- **METR-LA:** Traffic data are collected from observation sensors in the highway of Los Angeles County. We use 207 sensors and 4 months of data dated from 1st Mar 2012 until 30th Jun 2012 in the experiment.
- **PEMS-BAY:** Traffic data are collected by California Transportation Agencies Performance Measurement System (PeMS). We use 325 sensors in the Bay Area and 6 months of data dated from 1st Jan 2017 until 31th May 2017 in the experiment. The distribution of the sensors and the road network are shown in Figure 5(a).

Table 1: Baseline comparison on METR-LA and PEMS-BAY.

Methods	METR-LA			PEMS-BAY		
	MAE	MAPE	RMSE	MAE	MAPE	RMSE
HA	3.65	11.2%	6.4	2.88	6.8%	4.76
ARIMA	3.47	8.0%	7.81	2.33	5.4%	4.76
VAR	3.93	8.2%	6.59	2.32	5.0%	4.25
SVR	3.45	8.0%	7.05	2.48	5.5%	5.18
FNN	3.31	7.8%	7.43	2.30	5.4%	4.63
FC-LSTM	2.94	7.5%	5.92	2.20	5.2%	4.55
DCRNN	2.49	5.5%	3.95	1.72	3.9%	3.97
M-RGNN	2.34	5.3%	3.94	1.67	3.8%	3.73
MRes-RGNN-NG	2.19	5.1%	3.86	1.61	3.7%	3.54
MRes-RGNN-G	**2.07**	**4.9%**	**3.57**	**1.52**	**3.2%**	**3.23**

Compared Methods and Evaluation Metrics

We compare traffic prediction performance of our proposed MRes-RGNN with widely used time series regression models that include (i) HA: Historical Average, which can model traffic flow as a seasonal process, and then use weighted average of previous seasons as traffic prediction; (ii) ARIMA: Auto-regressive integrated moving average is a well-known model that can understand and predict future values in a time series; (iii) SVR: Support vector regression uses linear support vector machine for regression tasks. Subsequently, the traffic performance of MRes-RGNN is also compared with the deep neural network based methods such as (iv) FNN: Feed forward neural network with two hidden layers; (v) FC-LSTM: Fully connected LSTM neural networks; (vi) DCRNN: MRes-RGNN (Li et al. 2018) is compared with DCRNN uses both of the diffusion convolutions and RNN together in traffic prediction. In summary, we compare the traffic prediction performance of our proposed MRes-RGNN with 6 existing methods as discussed previously.

Three evaluation metrics can be used to measure the traffic prediction performance of above the methods as follows, (i) Mean Absolute Error (MAE), (ii) Mean Absolute Percentage Error (MAPE), and lastly, (iii) Root Mean Squared Error (RMSE).

Implementation Details

We first aggregate traffic speed readings into 5-minutes windows for METR-LA, 60-minutes windows for PEMS-BAY, and then apply Z-Score normalization. In each dataset, 70%, 20% and 10% of its dataset are split into training, validation and testing datasets, respectively.

To construct a sensor graph, we compute the pairwise road network distances among sensors and also build the adjacency matrix using the threshold Gaussian kernel:

$$w(i,j) = \begin{cases} exp(-\frac{dij^2}{\varphi^2}), i \neq j \text{ and } \exp(-\frac{\mathrm{d}_{\mathrm{i,j}}^2}{\varphi^2}) \geq \epsilon, \\ 0, \text{ otherwise}, \end{cases} \tag{9}$$

where $w(i,j)$ represents the edge weight from sensor i to sensor j, $d_{i,j}$ denotes the road network distance from sensor i to sensor j, φ is the standard deviation of the distances and ϵ is the threshold to control distribution and sparsity of the matrix. At the end, the adjacency matrix of PEMS-BAY is generated as shown in Figure 5(b).

In our implementation, two Res-RGNN layers are utilized, and the graph convolution kernel size is set to 64. To have a fair comparison, these parameters are set same in DCRNN. The proposed framework utilizes one Res-RGNN branch for $near$ time, and one hop[1] Res-RGNN branch for $daily$ period. In the above Res-RGNN branch, 6 observed data points are used to forecast traffic conditions. Another branch, the hop Res-RGNN for $daily$ period, 4 historical data points are utilized in the experiments. Due to the limited size of the datasets, a branch for a week hop is not used in the experiments. To overcome this issue, we have utilized the "metadata" as external factors to explore the weekly patterns.

Notations of Variants of Our Framework

In these experiments, some variants of our proposed framework are (i) M-RGNN denotes a RGNN branch and a hop RGNN branch that both of them are utilized, (ii) MRes-RGNN-G denotes our proposed framework with the gated residual scheme and multiple RGNN branches, (iii) MRes-RGNN-NG presents the residual shortcut connections are adopted, while the gated residual scheme is not utilized, (iv) RGNN denotes the proposed recurrent graph neural networks with only a branch for the $near$ temporal dependency without gated residual shortcuts, (v) Res-RGNN-G stands for one branch of RGNN for the $near$ with gated residual shortcuts, and lastly, (vi) Res-RGNN-NG means one branch of RGNN for the $near$ with residual shortcuts.

Please note that, in the above variants, the external factors are utilized.

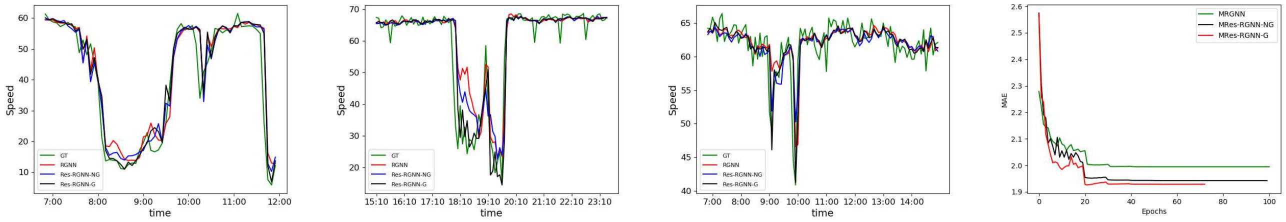
Overview of Performance Evaluation

The traffic prediction performance comparison of different approaches on both datasets is shown in Table 1. 3 out 7 variants of our proposed MRes-RGNN framework are utilized in the experiments.

Some phenomena can be clearly observed from Table 1. (1) Our proposed framework MRes-RGNN-G together with its variants M-RGNN and MRes-RGNN-NG have outperformed other baselines using measurement from above three evaluation metrics. (2) Even M-RGNN gets better results than other baselines. One of the reasons is the hop scheme that can effectively model temporal dependencies. (3) MRes-RGNN-G and MRes-RGNN-NG achieve better traffic prediction performance than M-RGNN using measurement from above three evaluation metrics. One of the reasons is to introduce residual shortcut connections into M-RGNN, while the performance is further improved by adopting the gated residual learning scheme.

The traditional time-series prediction methods such as HA and ARIMA cannot get good traffic prediction results as they rely on historical records to predict the future values without considering spatial and other related external features. Even the regression-based methods such as VAR

[1] The hop size of the daily branch is set to (all the minutes in a day)/q, where q is the length of the time segmentation in minute.



(a) (b) (c) (d)

Figure 6: Effectiveness of residual and gated mechanisms. (a) Speed prediction in the morning peak; (b) Speed prediction in the evening peak;(c) Speed prediction for sudden changes in the morning; (d) MAE versus the number of training epochs.

and SVR can take spatial correlations as their features. As a result, the regression-based methods can achieve better performance results than other conventional time-series approaches. However, they are still hard to capture the complex non-linear temporal dependencies and the dynamic spatial relationships. Deep neural networks methods (e.g., FNN and FC-LSTM) can overcome above the limitation. In Table 1, FNN and FC-LSTM have outperformed VAR and SVR.

Again, In Table 1, our proposed MRes-RGNN framework and DCRNN have outperformed FNN and FC-LSTM. One of the reasons is that both our framework and DCRNN can effectively model spatio-temporal dependencies using diffusion convolutions and RNN together.

Effectiveness of Gated Residual Mechanism

In this section, we evaluate the effects of gated residual mechanism. In the experiment, we only adopt a branch to extract spatial and $near$ temporal features. Figures 6(a) and 6(b) show the forecasting visualization in the morning peak hours and evening rush hours using METR-LA dataset, while Figure 6(c) shows the forecasting results on situations with sudden changes in the morning and evening, respectively. In above these figures, GT denotes the ground truth.

In this experiment, Res-RGNN-NG has been proven that can capture the trend of rush hours more accurately than RGNN. Importantly, it can detect the ending of the rush hours earlier than others. Therefore, the traffic performance can be further improved by adopting the gated mechanism. Figure 6(c) has shown that Res-RGNN is more sensitive to sudden changes than RGNN.

To investigate the running performance of above the compared deep learning models, the MAE of the testing data of METR-LA is plotted on the training phase, as shown in Figure 6(d). This figure has shown that the gated residual mechanism can achieve much faster training procedure.

Effectiveness of Multiple Res-RGNN and External Facotrs

In this section, we also investigate the traffic prediction performance of multiple Res-RGNN branches and external factors. Table 2 shows that the results of our proposed MRes-RGNN and its variants on the METR-LA dataset. In these figures, MRes-RGNN means our proposed framework, which combines gated residual learning, the *near* and the *daily* branches with external factors; Res-RGNN-N means only the *near* branch is adopted without external factors; Res-RGNN-ND denotes *near* and *daily* branches are adopted without external factors.

Table 2: Results of MRes-RGNN and its variants on the METR-LA dataset

Methods	MAE	MAPE	RMSE
Res-RGNN-N	2.32	5.18%	3.88
Res-RGNN-ND	2.16	4.98%	3.72
MRes-RGNN	**2.07**	**4.91%**	**3.57**

Table 3: Results of MRes-RGNN and its variants on the PEMS-BAY dataset

Methods	MAE	MAPE	RMSE
Res-RGNN-N	2.32	5.18%	3.88
Res-RGNN-ND	2.16	4.98%	3.72
MRes-RGNN	**2.07**	**4.91%**	**3.57**

From these figures, we can see that the Res-RGNN-N that only uses the *near* branch performs better than the other baselines as shown in Tables 2 and 3 . This demonstrates the effectiveness of applying gated residual RGNN to learn spatio-temporal features in traffic prediction. The performance is improved by adding the *daily* branch. Besides, considering the external factors can also improve the performance. Lastly, in the Tables 2 and 3, the proposed method that combines the *near* and *daily* with *external* features together can achieve the best results.

Conclusions

Traffic prediction is a vital part and challenging task in the domain of intelligent transportation system as it is affected by many complex factors, such as spatio-temporal dependencies with external influences. In this paper, we propose to adopt gated residual recurrent graph neural networks to capture graph-based spatial dependencies and temporal dynamics jointly. Experimental results have demonstrated that the proposed MRes-RGNN framework outperforms the state-of-the-art baselines. In our future work, we plan to investigate the spatio-temporal features learned by MRes-RGNN for better interpretability. We will also investigate to apply our framework in network anomaly detection (Xie et al. 2018a; Xie et al. 2018b; Xie et al. 2017)

Acknowledgements

The research was partially funded by the National Key R&D Program of China (Grant No.2018YFB1003401), the National Outstanding Youth Science Program of National Natural Science Foundation of China (Grant No. 61625202), the International (Regional) Cooperation and Exchange Program of National Natural Science Foundation of China (Grant No. 61661146006, 61860206011), the National Natural Science Foundation of China (Grant No.61602350), the Singapore-China NRF-NSFC Grant (Grant No. NRF2016NRF-NSFC001-111).

References

[Atwood and Towsley 2016] Atwood, J., and Towsley, D. 2016. Diffusion-convolutional neural networks. In *Advances in Neural Information Processing Systems*, 1993–2001.

[Bruna et al. 2013] Bruna, J.; Zaremba, W.; Szlam, A.; and LeCun, Y. 2013. Spectral networks and locally connected networks on graphs. In *arXiv preprint arXiv:1312.6203*.

[Chen et al. 2018] Chen, C.; Li, K.; Teo, S. G.; Chen, G.; Zou, X.; Yang, X.; Vijay, R. C.; Feng, J.; and Zeng, Z. 2018. Exploiting spatio-temporal correlations with multiple 3d convolutional neural networks for citywide vehicle flow prediction. In *IEEE International Conference on Data Mining*, 893–898.

[Chung et al. 2014] Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *arXiv preprint arXiv:1412.3555*.

[Defferrard, Bresson, and Vandergheynst 2016] Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, 3844–3852.

[Du et al. 2018] Du, S.; Li, T.; Gong, X.; Yu, Z.; and Horng, S.-J. 2018. A hybrid method for traffic flow forecasting using multimodal deep learning. In *arXiv preprint arXiv:1803.02099*.

[Du, Wang, and Wang 2015] Du, Y.; Wang, W.; and Wang, L. 2015. Hierarchical recurrent neural network for skeleton based action recognition. In *International conference on computer vision and pattern recognition*, 1110–1118.

[He et al. 2016a] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016a. Deep residual learning for image recognition. In *International conference on computer vision and pattern recognition*, 770–778.

[He et al. 2016b] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016b. Identity mappings in deep residual networks. In *European conference on computer vision*, 630–645.

[Hong 2011] Hong, W.-C. 2011. Traffic flow forecasting by seasonal svr with chaotic simulated annealing algorithm. In *Neurocomputing*, 2096–2107.

[Jabbarpour et al. 2018] Jabbarpour, M. R.; Zarrabi, H.; Khokhar, R. H.; Shamshirband, S.; and Choo, K. R. 2018. Applications of computational intelligence in vehicle traffic congestion problem: a survey. In *Soft Computing*, 2299–2320.

[Kipf and Welling 2016] Kipf, T. N., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. In *arXiv preprint arXiv:1609.02907*.

[Li et al. 2018] Li, Y.; Yu, R.; Shahabi, C.; and Liu, Y. 2018. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International conference on learning representations (to be appeared)*.

[Ma et al. 2017] Ma, X.; Dai, Z.; He, Z.; Ma, J.; Wang, Y.; and Wang, Y. 2017. Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. In *Sensors*, 818.

[Niepert, Ahmed, and Kutzkov 2016] Niepert, M.; Ahmed, M.; and Kutzkov, K. 2016. Learning convolutional neural networks for graphs. In *International conference on machine learning*, 2014–2023.

[Pascanu et al. 2013] Pascanu, R.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2013. How to construct deep recurrent neural networks. In *arXiv preprint arXiv:1312.6026*.

[Seo et al. 2016] Seo, Y.; Defferrard, M.; Vandergheynst, P.; and Bresson, X. 2016. Structured sequence modeling with graph convolutional recurrent networks. In *arXiv preprint arXiv:1612.07659*.

[Sutskever, Vinyals, and Le 2014] Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

[Teng 2016] Teng, S.-H. 2016. Scalable algorithms for data and network analysis. In *Foundations and Trends® in Theoretical Computer Science*, 1–274.

[Werbos 1990] Werbos, P. J. 1990. Backpropagation through time: what it does and how to do it. In *Proceedings of the IEEE*, 1550–1560.

[Xia et al. 2016] Xia, D.; Wang, B.; Li, H.; Li, Y.; and Zhang, Z. 2016. A distributed spatial–temporal weighted model on mapreduce for short-term traffic flow forecasting. In *Neurocomputing*, 246–263.

[Xie et al. 2017] Xie, K.; Li, X.; Wang, X.; Xie, G.; Wen, J.; Cao, J.; and Zhang, D. 2017. Fast tensor factorization for accurate internet anomaly detection. In *IEEE/ACM transactions on networking*, 3794–3807.

[Xie et al. 2018a] Xie, K.; Li, X.; Wang, X.; Cao, J.; Xie, G.; Wen, J.; Zhang, D.; and Qin, Z. 2018a. On-line anomaly detection with high accuracy. In *IEEE/ACM Transactions on Networking*, 1222–1235.

[Xie et al. 2018b] Xie, K.; Peng, C.; Wang, X.; Xie, G.; Wen, J.; Cao, J.; Zhang, D.; and Qin, Z. 2018b. Accurate recovery of internet traffic data under variable rate measurements. In *IEEE/ACM Transactions on Networking*, 1137–1150.

[Yao et al. 2018a] Yao, H.; Tang, X.; Wei, H.; Zheng, G.; Yu, Y.; and Li, Z. 2018a. Modeling spatial-temporal dynamics for traffic prediction. In *arXiv preprint arXiv:1803.01254*.

[Yao et al. 2018b] Yao, H.; Wu, F.; Ke, J.; Tang, X.; Jia, Y.; Lu, S.; Gong, P.; and Ye, J. 2018b. Deep multi-view spatial-temporal network for taxi demand prediction. In *Association for the advancement of artificial intelligence (to be appeared)*.

[Yu et al. 2017] Yu, R.; Li, Y.; Shahabi, C.; Demiryurek, U.; and Liu, Y. 2017. Deep learning: A generic approach for extreme condition traffic forecasting. In *SIAM International Conference on Data Mining*, 777–785.

[Yu, Yin, and Zhu 2017] Yu, B.; Yin, H.; and Zhu, Z. 2017. Spatio-temporal graph convolutional neural network: A deep learning framework for traffic forecasting. In *arXiv preprint arXiv:1709.04875*.

[Zeng et al. 2018] Zeng, Z.; Liang, N.; Yang, X.; and Hoi, S. C. H. 2018. Multi-target deep neural networks: Theoretical analysis and implementation. In *Neurocomputing*, 634–642.

[Zhang et al. 2018] Zhang, J.; Zheng, Y.; Qi, D.; Li, R.; Yi, X.; and Li, T. 2018. Predicting citywide crowd flows using deep spatio-temporal residual networks. In *Artificial Intelligence*, 147–166.

[Zhang, Zheng, and Qi 2017] Zhang, J.; Zheng, Y.; and Qi, D. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Association for the advancement of artificial intelligence*, 1655–1661.