International Journal of Advanced Smart Convergence Vol.12 No.4 88-97 (2023)
http://dx.doi.org/10.7236/IJASC.2023.12.4.88

IJASC 23-4-10

Traffic Flow Prediction with Spatio-Temporal Information Fusion using Graph Neural Networks

[1]Huijuan Ding, [2]Giseop Noh

[1] PhD Candidate, Dept. of Computer Information Engineering, Cheongju Univ., Korea
[2]Assistant Prof., Division of Software Convergence, Cheongju Univ., Korea
Dinghuijuan346@gmail.com, kafa46@cju.ac.kr

Abstract

Traffic flow prediction is of great significance in urban planning and traffic management. As the complexity of urban traffic increases, existing prediction methods still face challenges, especially for the fusion of spatiotemporal information and the capture of long-term dependencies. This study aims to use the fusion model of graph neural network to solve the spatio-temporal information fusion problem in traffic flow prediction. We propose a new deep learning model Spatio-Temporal Information Fusion using Graph Neural Networks (STFGNN). We use GCN module, TCN module and LSTM module alternately to carry out spatiotemporal information fusion. GCN and multi-core TCN capture the temporal and spatial dependencies of traffic flow respectively, and LSTM connects multiple fusion modules to carry out spatiotemporal information fusion. In the experimental evaluation of real traffic flow data, STFGNN showed better performance than other models.

Keywords: *Time Dependence; Spatial Dependence; Long-term Dependence; Deep Fusion Model; Traffic Prediction Model.*

1. INTRODUCTION

Transportation systems play a vital role in modern society. Traffic flow prediction can effectively manage traffic, ensure mobility, reduce congestion, and strengthen transportation infrastructure. However, due to the complexity and dynamics of traffic patterns in cities, accurate and reliable traffic flow prediction remains a challenging task.

Traditional methods are usually difficult to capture the complex spatiotemporal dependencies in traffic data. With the development of neural network technology, deep learning has made many significant achievements in obtaining complex nonlinear relationships in spatiotemporal data [1]. Graph convolutional neural network (GCN) can extend convolution operations to graph structure data in non-Euclidean space, and can well handle complex traffic network structure graph data [2]. Most of the existing traffic predictions have achieved good results in short-term predictions, but the long-term prediction effects still need to be improved. HixHop [3] obtains different order information based on GCN to mix different neighborhood information to obtain spatiotemporal features. The prediction effect is greatly improved, especially for short-term effects. TCN [4] is a temporal convolutional network that can effectively obtain time-dependent

Manuscript Received: october. 10, 2023 / Revised: october. 15, 2023 / Accepted: october. 21, 2023
Corresponding Author: kafa46@cju.ac.kr
Tel: +82-43-229-8496, Fax: +82-43-229-8110
Professor, Dept. of Software Convergence, Cheongju Univ., Korea

features. GWnet [5] uses GCN and TCN, stacked as different modules, and short-term prediction has been significantly improved. T-GCN [6] further uses GCN combined with gated recursive units to improve long-term prediction of traffic prediction. MTGNN [7] optimizes the graph structure and combines the advantages of HixHop and TCN to obtain the spatiotemporal characteristics of traffic spatiotemporal data, improving short-term and medium-term predictions. To further improve the effect of longer-term traffic prediction, we propose a Spatio-Temporal Information Fusion Graph Neural Network (STFGNN).

First, we propose a multi-model fusion approach to capture spatiotemporal correlations more comprehensively. It uses GCN adaptive composition to capture the spatial dependence characteristics of traffic flow and uses TCN multi-convolution kernel fusion to capture the time-dependence characteristics of traffic flow. The fusion module of GCN and TCN can not only capture spatial dependencies more deeply, but also capture temporal dependencies more comprehensively. The fusion model of GWnet [5] based on simple distance composition only performs simple model stacking when fusing TCN, which cannot capture spatiotemporal correlation more effectively and comprehensively.

Secondly, we propose a new spatio-temporal information fusion method to capture longer-term spatio-temporal correlations. It takes advantage of LSTM [8] and adds LSTM between each fusion module, which can learn and feature fusion of sequence data through the long short-term memory mechanism, thereby effectively capturing long-term dependencies and timing information in the sequence. Through jump connections, multiple fusion modules are connected to perform spatio-temporal information fusion. Simple module stacking and single modules separate the spatiotemporal correlation and have poor effect on long-term traffic flow prediction. For example, STGCN [9] uses separate modules when extracting spatial and temporal dependencies, which is very detrimental to long-term spatiotemporal information fusion.

The main contributions of this paper are summarized as follows:

1. We consider the spatio-temporal correlation characteristics of traffic flow data and propose a multi-model fusion method to capture the spatio-temporal correlation in a deeper manner.

2. Considering the complex and changing spatiotemporal information of traffic flow, we propose a new spatiotemporal information fusion method to capture longer spatiotemporal correlations.

3. We propose a new deep learning model, Spatio-Temporal Information Fusion using Graph Neural Networks (STFGNN), which can obtain longer spatiotemporal dependencies.

4. We used real multi-city traffic flow data set for verification. Compared with the existing baseline model, our model STFGNN has improved in both short - and long-term prediction, especially in long-term prediction.

2. Related Work

Spatiotemporal prediction refers to the process of predicting specific phenomena or events in time and space. It involves modeling and predicting the changes and developments of things in time and space dimensions to infer possible situations or trends in the future. Spatiotemporal prediction has a wide range of application scenarios in the real world, and traffic prediction is a typical application scenario of spatiotemporal prediction. Traditional traffic forecasting completes forecasting by driving models, such as historical average (HA), autoregressive integrated moving average (ARIMA) [10] and vector autoregressive (VAR) [11] and other models. These models are suitable for linear data analysis and have very low prediction accuracy for the prediction of nonlinear traffic flow data. In order to solve complex non-linear data, traditional machine learning methods are used for traffic prediction, such as support vector regression (SVR) [12] and random forest regression (RFR) [13], although these models can handle high-dimensional complex nonlinear relationships, it becomes difficult and time-consuming to deal with large-scale traffic flow data sets. Deep learning utilizes more features and complex architectures to handle large-scale and complex spatio-temporal data. Recurrent Neural Network (RNN) [14] and its enhanced version LSTM [15], GRU [16] can handle the time dependence of time series, but cannot handle the dependence of spatial dimension nodes. Convolutional neural network (CNN) [17] can obtain traffic spatial dependence in a gridded manner in Euclidean space. Traffic network is an irregular graph structure, and the way of grid division cannot well represent the traffic network structure. Graph convolutional neural network (GCN) [18] has been widely used in traffic prediction, and has received much attention in the extraction of spatial dependency relationships.

GCN can model in non-Euclidean space to capture the correlation between spatial nodes. FastGCN [19] samples each convolutional layer and samples a certain number of nodes, but the connections between layers are sparse. Cluster-GCN [20] samples subgraphs through adaptive sampling, but when sampling subgraphs, neighborhood search is subject to certain limitations. NetGAN [21] uses random walks to generate new graphs. Since the graph is generated in a global way, it suffers from certain limitations when generating large graphs with many traffic data nodes. WaveNet [5] uses an adaptive method to automatically learn the structure of static graphs. In transportation networks, it is necessary to potentially learn the dependencies of dynamic spaces. ASTGCN [22] dynamically learns the spatial dependence between nodes based on CNN, but under the influence of external information, the efficiency of obtaining node features decreases. MixHop [3] is based on GCN to mix the features of different neighbors to obtain different order information, which can better capture the dependency features of spatial nodes, but in the time dimension, it cannot capture the time dependence features well. Graph Wave [5] captures the spatiotemporal dependence of the traffic network structure by gating TCN and stacking GCN at the same time. However, fixed convolution kernel is used in cross-layer information transmission, resulting in certain limitations in local feature learning of global kernel. MTGNN [7] carried out convolution operations based on the hybrid jump method through adaptive composition, stacked the initial TCN structure of multiple convolution cores, and completed the dynamic and static feature extraction of feature graphs through cross-graph connection. However, in the multi-level structure, local information capture was insufficient.

3. Methodology

3.1 Problem Definition

This paper mainly studies traffic flow prediction. Let $z_t \in \mathrm{R}^N$ be the value of a multi-variable with dimension N at time step t, and the value of the i-th variable at time step t is represented as $z_t[i] \in R$. Given the multivariate observation sequence $\mathrm{X} = \{\mathrm{z}_{\mathrm{t}_1}, \mathrm{z}_{\mathrm{t}_2}, \mathrm{z}_{\mathrm{t}_3}, \dots \mathrm{z}_{\mathrm{t}_\mathrm{m}}\}$ of the traffic historical sequence m time steps, predict the traffic flow $\mathrm{Y} = \{\mathrm{z}_{\mathrm{t}_{\mathrm{m}+1}}, \mathrm{z}_{\mathrm{t}_{\mathrm{m}+2}}, \mathrm{z}_{\mathrm{t}_{\mathrm{m}+3}}, \dots \mathrm{z}_{\mathrm{t}_{\mathrm{m}+\mathrm{n}}}\}$ of n time steps in the future. We define some traffic flow prediction problems, described as follows:

Definition 1: Traffic network graph structure. $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ represents the traffic network graph structure. Where V represents the set of nodes, which is the set of all traffic monitoring points, and E represents the set of edges, which is the connection relationship between monitoring points. The number of nodes in the traffic network is represented by N.

Definition 2: Traffic node neighborhood. Let $\mathrm{e} = (\mathrm{q}, \mathrm{p}) \in \mathrm{E}$, denote an edge from q to p, where $\mathrm{q} \in \mathrm{V}$ denotes a node. The neighborhood of node q is denoted as $\mathrm{N}(\mathrm{q}) = \{\mathrm{p} \in \mathrm{V} | (\mathrm{q}, \mathrm{p}) \in \mathrm{E}\}$.

Definition 3: Traffic network adjacency matrix. The adjacency matrix is a method of using a matrix to represent the graph structure, defined as $\mathrm{A} \in \mathrm{R}^{\mathrm{N} \times \mathrm{N}}$, if $(q_i, q_j) \in E$, then $\mathrm{A}_{ij} = c > 0$, if $(q_i, q_j) \notin E$, then $\mathrm{A}_{ij} = 0$.

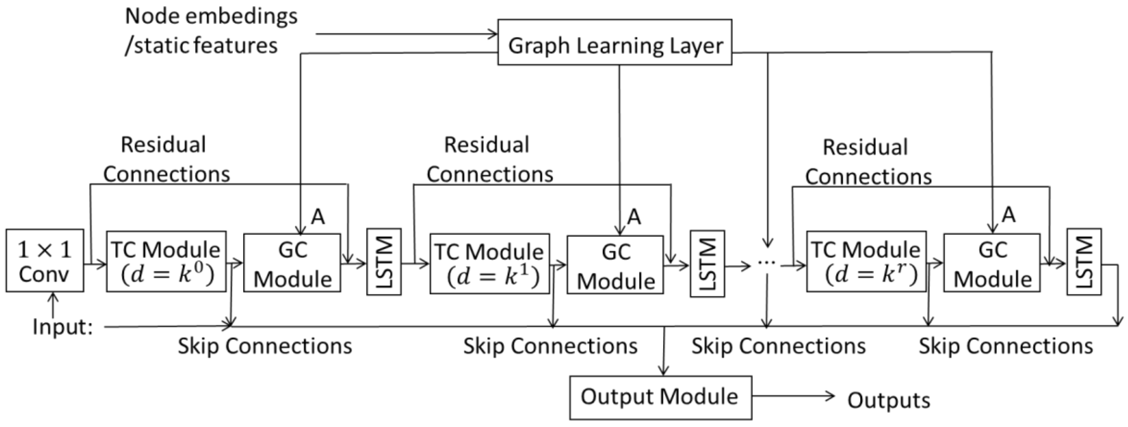


Figure 1. The framework of STFGNN

3.2 Framework

We provide the overall framework structure diagram of STFGNN in Figure 1. In the framework structure diagram, the input of traffic flow is first mapped to the latent space through a standard convolution of 1. Then, the temporal convolution module, the graph convolution module and the LSTM module alternate with each other. The temporal convolution module captures the temporal dependence more deeply, and the graph convolution module captures the spatial dependence more deeply. The addition of the LSTM module captures the long-term dependencies in the traffic flow input sequence, dynamically adjusts and updates the sequence features, and better adapts to the graph convolution module to extract features. In the temporal convolution module, the parameters that control the size of the receptive field are controlled, and the expansion factor d increases exponentially with k. In the model, residual connections are added to each graph convolution module, and skip connections are added after each temporal convolution module to alleviate the problem of gradient disappearance. Finally, the features of the hidden layer are mapped to the required dimensions through the output module, and the result is output.

3.3 Graph Convolution Module

We use graph learning layers to adaptively learn graph adjacency matrices. In existing studies, most researchers use distance to find similarities between nodes, and the distance measurement is bidirectional [23], which limits the efficiency of the model in processing larger graphs. Since the scale of the traffic network map is relatively large, we use one-way distance measurement to compose the map. The composition process is as follows:

$$\mathrm{V}_1 = tanh(\gamma E_1 \mathcal{M}_1),\ \mathrm{V}_2 = tanh(\gamma E_2 \mathcal{M}_2), \tag{1}$$

$$\mathrm{A} = Relu\left(tanh\left(\gamma(\mathrm{V}_1 \mathrm{V}_2^T - \mathrm{V}_2 \mathrm{V}_1^T)\right)\right) \tag{2}$$

where V_1 and V_2 represent node embedding and random initialization, $\mathcal{M}_1$ and $\mathcal{M}_2$ are parameters that can be learned during the model learning process, and γ is a hyperparameter that can control the saturation rate of the activation function. Formula 2 can generate an adjacency matrix with asymmetric properties.

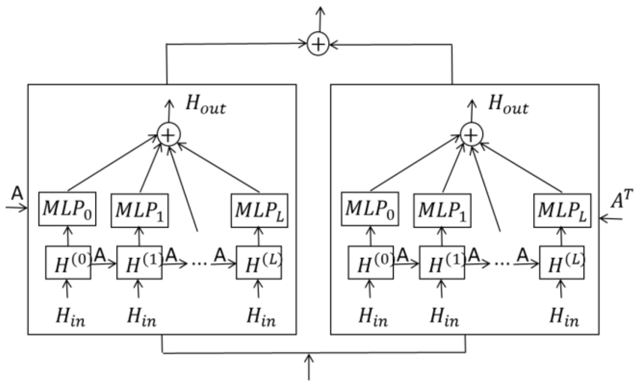


Figure 2. Graph Convolution Module

The graph convolution module mainly integrates the information of neighbor nodes with the node's own information to extract the spatial dependence of the traffic network. We use hybrid hop information propagation to complete information fusion. As shown in Figure 2, the inflow and outflow information of each node is processed through two hybrid hop layers, and finally the layer information is aggregated. The process of hybrid hop information propagation is defined as follows:

$$H^{(L)} = \varepsilon H_{in} + (1-\varepsilon)\widetilde{\mathrm{A}}\mathrm{H}^{(L-1)}, \tag{3}$$

$$H_{out} = \sum_{i=0}^{L} H^{(L)} W^{(L)} \tag{4}$$

where ε represents the hyperparameter that controls the ratio of the original state of the root node, L is the depth of information propagation, Formula 4 represents the process of information selection, and $W^{(L)}$ is a parameter matrix as a feature selector. H_{in} indicates that the hidden state of the previous layer is the current input, and H_{out} is the hidden state of the current output.$H^{(0)} = H_{in}, \widetilde{\mathrm{A}} = \widetilde{\mathrm{D}}^{-1}(\mathrm{A}+\mathrm{I}), \widetilde{D}_{ii} = 1 + \sum_j \mathrm{A}_{ij}$.

3.4 Temporal Convolution Module

The temporal convolution module consists of dilated convolutions of different convolution kernels to form a dilated inception layer, and then the inception layer completes the filtering of information through the gating mechanism, as shown in Figure 3. The selection of multiple convolution kernels can capture a wider range of information through different convolution kernel sizes. We choose a convolution kernel with a size of 2,3,6,7 that can also cover the time period, such as representing a period of 8, and the model can pass through an inception layer with a convolution kernel size of 3 and then through an inception layer with a convolution kernel size of 6. The form in which we dilated inception layer is as follows:

$$\mathrm{x} = concat(\mathrm{x} * \mathrm{g}_{1\times2}, \mathrm{x} * \mathrm{g}_{1\times3}, \mathrm{x} * \mathrm{g}_{1\times6}, \mathrm{x} * \mathrm{g}_{1\times7}), \tag{5}$$

$$\mathrm{x} * \mathrm{g}_{1\times\mathrm{k}^{(\mathrm{t})}} = \sum_{\mathrm{s}=0}^{\mathrm{k}-1} \mathrm{g}_{1\times\mathrm{k}^{(\mathrm{s})(\mathrm{t}-\mathrm{d}\times\mathrm{s})}} \tag{6}$$

where $\mathrm{x} \in \mathrm{R}^{\mathrm{T}}$ is a one-dimensional input sequence, and $\mathrm{g}_{1\times2} \in \mathrm{R}^2$,$\mathrm{g}_{1\times3} \in \mathrm{R}^3$,$\mathrm{g}_{1\times6} \in \mathrm{R}^6$,$\mathrm{g}_{1\times7} \in \mathrm{R}^7$ represents filters with different convolution kernel sizes. $\mathrm{x} * \mathrm{g}_{1\times\mathrm{k}^{(\mathrm{t})}}$ represents dilated convolution, and d is the dilation factor.

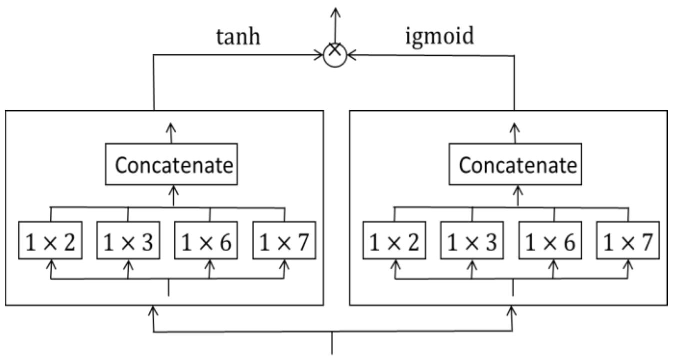


Figure 3. Temporal Convolution Module

3.5 LSTM Module

LSTM helps the network better handle the balance between long-term memory and short-term changes. Located among several Graph Convolution and Temporal Convolution modules, LSTM can receive the outputs of these modules as input sequences and learn the sequence data and integrate features through its long- and short-term memory mechanism, thus effectively capturing the long-term dependencies in the sequences. As shown in Figure 4, LSTM uses three gates to control the flow of information, namely input gate, forgetting gate and output gate. The process is described as follows:

$$\mathrm{g}_f = \sigma\left(w_f[h_{t-1}, \hat{\mathrm{x}}_t] + b_f\right) \ , \tag{7}$$

$$i_t = \sigma(w_i[h_{t-1}, \hat{\mathrm{x}}_t] + b_i) \ , \tag{8}$$

$$\grave{c}_t = tanh(w_c[h_{t-1}, \hat{\mathrm{x}}_t] + b_c) \ , \tag{9}$$

$$c_t = \mathrm{g}_f c_{t-1} + i_t \grave{c}_t \ , \tag{10}$$

$$o_t = \sigma(w_o[h_{t-1}, \hat{\mathrm{x}}_t] + b_o) \ , \tag{11}$$

$$h_t = o_t tanh(c_t) \tag{12}$$

where g_f represents the forget gate, i_t represents the input gate, c_t represents the gating unit, o_t represents the output gate, and h_t represents the final output. w_f and b_f are the parameters that can be learned by the forgetting gate, w_i and b_i are the parameters that can be learned by the input gate, w_c and b_c are the parameters that can be learned by the gate control unit, and w_o and b_o are the parameters that can be learned by the output gate. $\sigma(*)$ and $tanh(*)$ are the activation functions.

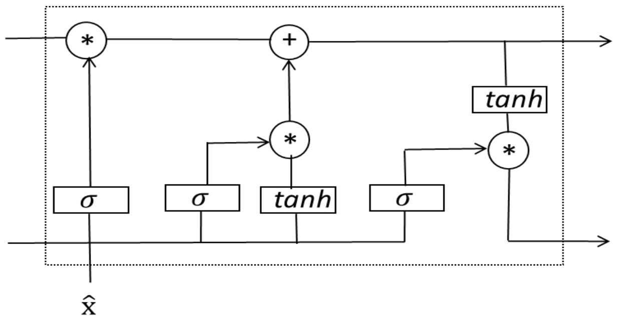


Figure 4. LSTM Module

4. Experiments

4.1 Experimental Setup

Our experimental data uses the public real traffic flow data set for experiments. We conduct verification on the Metr-LA public dataset, which is provided by Uber and records traffic sensor data in the Los Angeles area. We choose 207 sensors as 207 nodes, and the recorded data is collected every 30 seconds. Before model training, the 30 seconds of data are aggregated into a 5-minute time step, and the data set is divided, 70% of the data is used for training, 20% is used for testing, and 10% is used for verification.

Use Pytorch to run on GPU, Batch size is set to 64, initial learning rate is set to 0.001, and the number of iterations of all models is 100.We use 3 graph convolution modules, the graph convolution module and the time convolution module are set to 32 output channels, and the number of layers of the time convolution module is set to 3. In formula 3, ε is set to 0.5, and the depth L of the hybrid jump propagation layer in formula 4 is set to 5. When we evaluate the model, we use the more commonly used evaluation indicators MAE, RMSE and MAPE.

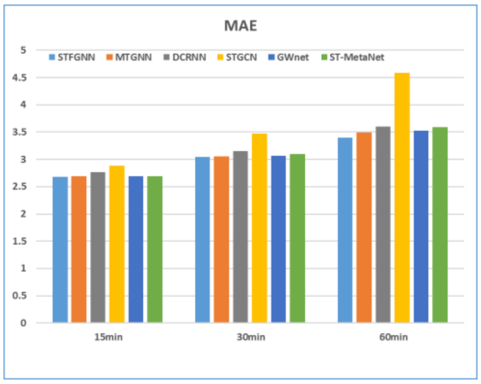
3.5 Experimental Results

We use different baseline models to compare with our model. The baseline models compared with our model are DCRNN [23], STGCN [9], GWnet [5], ST-MetaNet [24], MTGNN [7]. The experimental results of the model are shown in Table 1.

Table 1. Comparison of prediction performance between STFGNN and baseline models.

Model	Horizon 3			Horizon 6			Horizon 12		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
STFGNN	**2.68**	5.16	**6.85%**	**3.04**	**6.15**	**8.18%**	**3.40**	**7.20**	**9.70%**
MTGNN	2.69	5.18	6.86%	3.05	6.17	8.19%	3.49	7.23	9.87%
DCRNN	2.77	5.38	7.30%	3.15	6.45	8.80%	3.60	7.60	10.50%
STGCN	2.88	5.74	7.62%	3.47	7.24	9.57%	4.59	9.40	12.70%
GWnet	2.69	**5.15**	6.90%	3.07	6.22	8.37%	3.53	7.37	10.01%
ST-MetaNet	2.69	5.17	6.91%	3.10	6.28	8.57%	3.59	7.52	10.63%

Our model STFGNN performs well compared with similar models. Compared with the DCRNN and STGCN models that perform static graph calculations, STFGNN uses dynamic graph calculations to show excellent prediction performance in long-term, medium-term and short-term predictions. Compared with the model ST-MetaNet, which uses a self-attention mechanism to adjust the calculation method of static images, our model still has excellent prediction performance. Compared with the adaptive composition model GWnet, STFGNN still shows the best performance in medium and long-term prediction because GWnet still needs to be combined with static graph calculations while adaptive composition. Compared with the MTGNN model, although the composition method is the same, our model uses LSTM to adjust different modules, and the performance of short-term, medium-term, and long-term predictions is still improved, especially in long-term prediction, the performance improvement is outstanding.



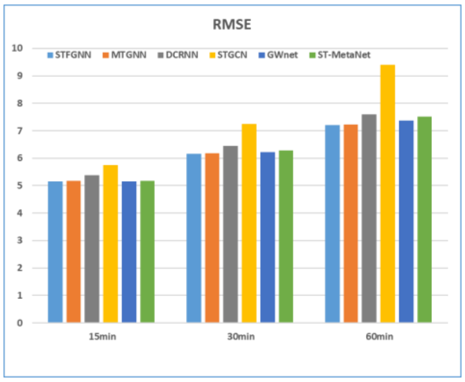


Figure 5. Comparison of MAE and RMSE in different baseline models.

To further illustrate the good performance of our model compared to the same baseline model, we made further comparisons, as shown in Figure 5. In the figure, the model STFGNN, which we can observe more clearly, is represented by the blue bar chart, showing the best results in both MAE and RMSE. In particular, the MAE and RMSE results of STFGNN are more prominent in long-term prediction.

5. Conclusion

In this paper, we address the problem of insufficient long-term memory capture of long-term spatiotemporal correlation in traffic prediction, taking advantage of the spatial dependence of GCN, the temporal dependence of TCN, and the advantages of LSTM with long-term memory function. Spatio-Temporal Information Fusion using Graph Neural Networks (STFGNN) is proposed. To capture the spatio-temporal correlation more deeply, we use multi-model fusion to deeply capture the spatio-temporal dependence of traffic flow. To deal with the complex spatiotemporal correlation of traffic flow, we propose a new spatiotemporal information fusion method that can capture longer spatiotemporal dependencies. We evaluated the model in real traffic flow data. Compared with similar models, STFGNN showed excellent performance, especially in long-term traffic flow prediction. In future work, we will further explore the optimization of different modules to make the neural network more flexible in application.

REFERENCES

[1] A. Ang and M. Piazzesi, ‘A no-arbitrage vector autoregression of term structure dynamics with macroeconomic and latent variables’, J. Monet. Econ., vol. 50, no. 4, pp. 745–787, May 2003, doi: 10.1016/S0304-3932(03)00032-1.

[2] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, ‘Spectral Networks and Locally Connected Networks on Graphs’. arXiv, May 21, 2014. doi: 10.48550/arXiv.1312.6203.

[3] S. Abu-El-Haija et al., ‘MixHop: Higher-Order Graph Convolutional Architectures via Sparsified Neighborhood Mixing’, in Proceedings of the 36th International Conference on Machine Learning, PMLR, May 2019, pp. 21–29. Accessed: Nov. 22, 2023. [Online]. Available: https://proceedings.mlr.press/v97/abu-el-haija19a.html

[4] S. Bai, J. Z. Kolter, and V. Koltun, ‘An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling’. arXiv, Apr. 19, 2018. doi: 10.48550/arXiv.1803.01271.

[5] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, ‘Graph WaveNet for Deep Spatial-Temporal Graph Modeling’. arXiv, May 31, 2019. doi: 10.48550/arXiv.1906.00121.

[6] L. Zhao et al., ‘T-GCN: A Temporal Graph Convolutional Network for Traffic Prediction’, IEEE Trans. Intell. Transp. Syst., vol. 21, no. 9, pp. 3848–3858, Sep. 2020, doi: 10.1109/TITS.2019.2935152.

[7] 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 Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, in KDD ’20. New York, NY, USA: Association for Computing Machinery, 20 2020, pp. 753–763. doi: 10.1145/3394486.3403118.

[8] R. C. Staudemeyer and E. R. Morris, ‘Understanding LSTM -- a tutorial into Long Short-Term Memory Recurrent Neural Networks’. arXiv, Sep. 12, 2019. doi: 10.48550/arXiv.1909.09586.

[9] B. Yu, H. Yin, and Z. Zhu, ‘Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting’, in Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, Jul. 2018, pp. 3634–3640. doi: 10.24963/ijcai.2018/505.

[10] ‘Modeling and Forecasting Vehicular Traffic Flow as a Seasonal ARIMA Process: Theoretical Basis and Empirical Results | Journal of Transportation Engineering | Vol 129, No 6’. Accessed: Nov. 22, 2023. [Online]. Available: https://ascelibrary.org/doi/abs/10.1061/(ASCE)0733-947X(2003)129:6(664)

[11] E. Zivot and J. Wang, Eds., ‘Vector Autoregressive Models for Multivariate Time Series’, in Modeling Financial Time Series with S-PLUS®, New York, NY: Springer, 2006, pp. 385–429. doi: 10.1007/978-0-387-32348-0_11.

[12] ‘Forecasting holiday daily tourist flow based on seasonal support vector regression with adaptive genetic algorithm - ScienceDirect’. Accessed: Nov. 22, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494614005328

[13] U. Johansson, H. Boström, T. Löfström, and H. Linusson, ‘Regression conformal prediction with random forests’, Mach. Learn., vol. 97, no. 1, pp. 155–176, Oct. 2014, doi: 10.1007/s10994-014-5453-0.

[14] J. L. Elman, ‘Distributed representations, simple recurrent networks, and grammatical structure’, Mach. Learn., vol. 7, no. 2, pp. 195–225, Sep. 1991, doi: 10.1007/BF00114844.

[15] S. Hochreiter and J. Schmidhuber, ‘Long Short-Term Memory’, Neural Comput., vol. 9, no. 8, pp. 1735–1780, Jan. 1997, doi: 10.1162/neco.1997.9.8.1735.

[16] K. Cho et al., ‘Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation’. arXiv, Sep. 02, 2014. doi: 10.48550/arXiv.1406.1078.

[17] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, ‘Convolutional Sequence to Sequence Learning’, in Proceedings of the 34th International Conference on Machine Learning, PMLR, Jul. 2017, pp. 1243–1252. Accessed: Nov. 23, 2023. [Online]. Available: https://proceedings.mlr.press/v70/gehring17a.html

[18] C. Song, Y. Lin, S. Guo, and H. Wan, ‘Spatial-Temporal Synchronous Graph Convolutional Networks: A New Framework for Spatial-Temporal Network Data Forecasting’, Proc. AAAI Conf. Artif. Intell., vol. 34, no. 01, Art. no. 01, Apr. 2020, doi: 10.1609/aaai.v34i01.5438.

[19] J. Chen, T. Ma, and C. Xiao, 'FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling'. arXiv, Jan. 30, 2018. doi: 10.48550/arXiv.1801.10247.

[20] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh, 'Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks', in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, in KDD '19. New York, NY, USA: Association for Computing Machinery, 25 2019, pp. 257–266. doi: 10.1145/3292500.3330925.

[21] A. Bojchevski, O. Shchur, D. Zügner, and S. Günnemann, 'NetGAN: Generating Graphs via Random Walks', in Proceedings of the 35th International Conference on Machine Learning, PMLR, Jul. 2018, pp. 610–619. Accessed: Nov. 23, 2023. [Online]. Available: https://proceedings.mlr.press/v80/bojchevski18a.html

[22] 'Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting | Proceedings of the AAAI Conference on Artificial Intelligence'. Accessed: Nov. 23, 2023. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/3881

[23] Y. Li, R. Yu, C. Shahabi, and Y. Liu, 'Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting'. arXiv, Feb. 22, 2018. doi: 10.48550/arXiv.1707.01926.

[24] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, 'Urban Traffic Prediction from Spatio-Temporal Data Using Deep Meta Learning', in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, in KDD '19. New York, NY, USA: Association for Computing Machinery, 25 2019, pp. 1720–1730. doi: 10.1145/3292500.3330884.