Multi-view Cascading Spatial-Temporal Graph Neural Network for Traffic Flow Forecasting

Zibo Liu[1](✉), Kaiqun Fu[2], and Xiaotong Liu[3]

[1] Virginia Tech, Falls Church, VA 22043, USA
zbliu@vt.edu
[2] South Dakota State University, Brookings, SD 57007, USA
Kaiqun.Fu@sdstate.edu
[3] George Washington University, Washington, DC 20052, USA
liuxiaotong2017@gwu.edu

Abstract. Spatial-temporal patterns have been applied in many areas, such as traffic forecasting, skeleton-based recognition, and so on. In such areas, researchers can convert the prior knowledge into graphs and combine the latent graph dependencies into original features to get better representation. However, few works focus on the underlying pattern in the original feature, and they cannot capture the flexible interaction both spatially and temporally. What is more, they often ignore the heterogeneity in spatial-temporal data. In this paper, we solve this problem by designing a novel model, Multi-view Cascading Spatial-temporal Graph Neural Network. Our model has a cascading structure to enhance interaction and capture heterogeneity. Also, it takes the differencing orders of flow data into account to get a better representation and contains specific coupled graphs designed based on the sliding window technique. Extensive experiments are conducted on four real-world datasets, demonstrating that our method achieves state-of-the-art performance and outperforms other baselines.

Keywords: Traffic forecasting · Multi-view · Spatio-temporal · Attention · Graph neural network · Graph Convolutional Network

1 Introduction

With the urban expansion and increasing number of vehicles, intelligent transportation systems are developed to make our trips more efficient and safe. Using data like traffic flow/speed/density, there are many real-world applications such as optimal route and estimated arrival time [17,18], abnormal traffic behavior detection [15] and so on. Given the broad applications of traffic data, we focus on traffic flow forecasting, which is a technology based on the past traffic flow to predict the future traffic flow. It is challenging due to the complex intra-dependencies (i.e., temporal correlations within one traffic series) and inter-dependencies (i.e., spatial correlations among huge correlated traffic series) [12]

E. Pimenidis et al. (Eds.): ICANN 2022, LNCS 13530, pp. 605–616, 2022.
https://doi.org/10.1007/978-3-031-15931-2_50

generated from different sources such as the different traffic detectors in the intersections and various vehicles' data.

Researchers first used the traditional machine learning method to solve problems, like ARIMA [3]. Then when the outbreak of deep learning, they use Convolutional Neural Network (CNN) and some graph-based CNN, get >10% performance improvement compared with machine learning methods. The capstone work was STGCN [21] in 2017. It achieved better performance because it facilitated spatial and temporal dependencies and combined them with the CNN layers. Then is STSGCN [12], they divide the whole time series data into data pieces to capture the heterogeneity between different time stamps, designed a new pattern.

However, there are several disadvantages to the prior models: 1) The graph's topology is static and predefined based on the location map. It can not be guaranteed as optimal for traffic flow tasks. For example, when we concern with two nodes, A, B. A, B are big cities on the highway, but it is not connected due to the long distance. However, there are a great number of tracks to deliver groceries from A to B. It is difficult to capture this dependency through predefined graphs. Also, some models use the 0/1 spatial graphs instead of distance graphs. It is not fair to treat the node pair with a relatively high distance and the one with a low distance as the same (the distances are under the threshold). 2) The prior models use an entire network to deal with the entire spatial-temporal data flow. They could miss some essential dependencies like heterogeneity and homogeneity. For example, if a car accident happened on the road an hour ago, the current flow would be lower because the drivers always choose a faster route, not this road, which is more likely to be jammed. Nevertheless, if the car accident happened 10 h ago, we cannot say whether now it is jammed or not because the road is probably back to normal. 3) In the aggregation step and final output transformation step, exist model structure is relatively simple to extract the complete and higher dimensional information. Our work overcomes these shortages, and the contributions are as follows:

- We propose a novel spatial-temporal graph neural network to capture local and global spatial-temporal correlations. We discover that the various differencing orders of original data are additional support for feature representation in the traffic flow task. Based on that, we build our paralleled modules on the sliding window technique to find the same or different patterns, instead of using a single module to deal with the whole time series.
- We also build the hierarchical and temporal cascading structures to keep long-term dependencies and flexible interactions by learnable parameters to decide the weight of information to communicate through modules.
- Meanwhile, we developed a delicate graph block to aggregate the graph features, which could capture the subtle relationships of features. Extensive experiments are conducted on four real-world datasets, and the experimental results show that our model consistently outperforms all the baseline methods.

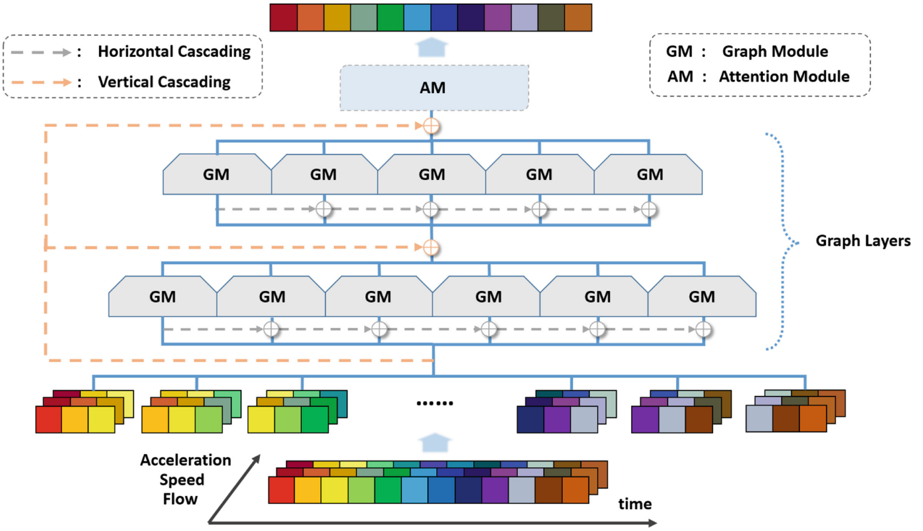


Fig. 1. This is a diagram of the whole model. The bottom color rows represent the three channels, flow, speed, and acceleration. Then the data is divided into several windows to feed into each Graph Module. Between layers and between Graph Modules are cascading connections. After a few Graph Layers, the Attention Module is the final part of getting the output. (Color figure online)

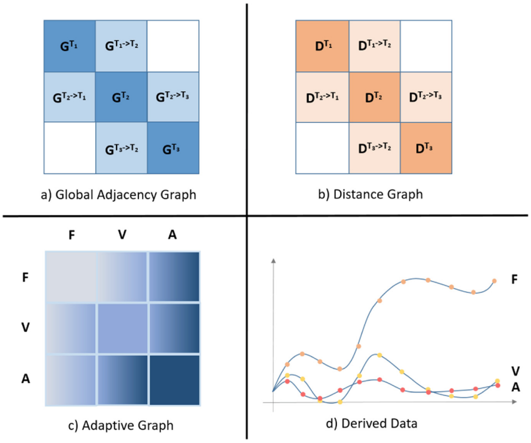
2 Related Work

2.1 Machine Learning and Convolutional Method

Like K-Nearest Neighbor (KNN) [4], they calculate the center of cluster and assign each data point to the specific cluster. ARIMA used the differencing function to find the fit pattern. The derivation of it, SARIMA [16] adds a specific ability to recognize the seasonal pattern. The machine learning models are designed by human-set rules and are sensitive to outliers. It is challenging for them to learn the actual relationship of features. But the deep learning methods could work well because millions of neurons and kernels could learn the higher dimensional features. CNN/RNN based models and their derivation could be transferred to solve time series, such as ConvLSTM [20], ST-LSTM [11], STCNN [7].

2.2 Graph Method

Graph-based methods could be summed up into three parts. First, constructing the data into the nodes and finding their neighbors based on the latent relationship. Second, using the developing method to upgrade the information of the node. Third, merging the upgraded information and using the aggregation method to reach the final output. For example, Graph Convolutional Network



a) Global Adjacency Graph

b) Distance Graph

c) Adaptive Graph

d) Derived Data

Fig. 2. This figure contains key components in our model. (Color figure online)

(GCN) [9] is a derivation of Graph Neural Network (GNN), using the Convolutional Network to get node representation. Recent work like GAT [14], which introduces attention mechanism into the graph field, and GraphSAGE [6], which generates node embeddings by sampling and aggregating features.

2.3 Traffic Flow Prediction

For this topic, the deep network could learn higher-dimensional features of data. AGCRN [1] uses the recurrent network as the main structure to obtain long-term relationships through time and automatically form a graph based on the data. STSGCN [12] divides the data into data pieces, takes the heterogeneities in spatial-temporal dependencies into account. STGODE [5] focuses on Ordinary Differential Equations to mimic the trajectory of traffic flow. STFGNN [10] is a work derived from STSGCN, used the Dynamic Time Warping algorithm [2] to gain a unique temporal graph.

3 Problem Formulation

We denote graph set as $\mathcal{G} = \{V, E, G, D, \mathcal{A}\}$, V is the set of nodes, E is set of edges, G is a global graph, D is a distance graph, $\mathcal{A}$ is set of data-based graph, $\mathcal{A} = (A^1, A^2, ..., A^{\mathcal{T}})$. The problem of spatial-temporal forecasting can be described as: learning a mapping function f which maps the current spatial-temporal data series $\mathcal{X} = (X_{t-T+1}, X_{t-T+2}, ..., X_t)$ into the future spatial-temporal data series $\mathcal{Y} = (X_{t+1}, X_{t+2}, ..., X_{t+T'})$, where T and T' denotes the length of historical and the target time series to forecast respectively. $\hat{\mathcal{X}} = f(\mathcal{X}, \mathcal{G})$, $\hat{\mathcal{X}}$ is the model output, we want the $\hat{\mathcal{X}}$ as close to $\mathcal{Y}$ as possible.

4 Model

The followings are three general characteristics of our model. 1) Differencing and multi-view: We derive the different orders of original flow data to enrich the model input. Furthermore, we design three graphs from different views. The first two are static graphs, which are global and distance graphs. And the third one is an adaptive graph. It has an independent graph for each data piece and provides extra hidden patterns for the model. 2) Cascading: The model has two types of cascading. One is temporal, which transfers the information from the previous data piece to the current data piece with a weight value, which captures data heterogeneity by paralleled modules. The other is hierarchical, which transfers the complete features to the following graph layer with a weight value. 3) In the aggregation step and final output transformation step, we borrow the idea of attention [13] to capture both global and local features better.

4.1 Differencing Process

The latent feature could be inconspicuously hidden in the original data. And it can be a great help for the model to have a better performance. Later in the ablation study, it was proved. In our opinion, this traffic flow data F $\in \mathbb{R}^{N*T*1}$ is not powerful enough to provide the learning features. Here, we borrow the concept in Physics, using speed and acceleration to express first-order differencing and second-order differencing information in traffic flow data, as shown in Fig. 2d). We use S_1, S_2, S_3 to represent the flow, speed, and acceleration channels, respectively. S_1, S_2, $S_3 \in \mathbb{R}^{T*N*1}$. We concatenate all three dimensions together in the last dimension, S $\in \mathbb{R}^{N*T*3}$. In Fig. 1, the three-color row on the bottom represents the flow, speed, and acceleration channel.

4.2 Graph

Global Graph *G*. The size of Global Graph is $\mathbb{R}^{3N*3N}$, here we set the 3 as the window size, the sliding window goes along the time axis. The diagonal of the adjacency matrix are three same $N \times N$ spatial graphs. For example in Fig. 2 a), the Global Graph contains three timestamps T_1, T_2, T_3, if v_i and v_j are connected in the navy blue squares, this pair of nodes is also connected in four shallow blue squares $T_1 - T_2$, $T_2 - T_1$, $T_2 - T_3$, and $T_3 - T_2$.

$$G_{i,j} = \begin{cases} 1, if\ v_i\ and\ v_j\ are\ neighbors \\ 0, \qquad\qquad otherwise \end{cases} \tag{1}$$

Distance Graph *D*. Even though the global graph gives the connection information among nodes, it still lacks the meaning of distance. Here, we provide the specific value between each connected node, instead of 0/1 in the Global Graph.

We follow what we did in Global Graph, using the distance between node i and j, E_{ij}. The size of the Distance Graph is $\mathbb{R}^{3N*3N}$, shown as Fig. 2b).

$$D_{i,j} = \begin{cases} \frac{\sigma}{E_{ij}}, if\ G_{i,j}\ =\ 1 \\ 0,\ \ otherwise \end{cases} \tag{2}$$

Adaptive Graph *A*. Considering the previous track-highway example, we propose this well-designed graph to capture the hidden patterns. This graph aggregates S_1, S_2, S_3 based on the sliding window strategy. It is a data-driven graph that learns one unique graph for each window. We use the multiplication among S_1, S_2, S_3 to emphasize the underlying correlation and to determine how strong the connection is. Figure 2c) is a diagram of the Adaptive Graph. In Eq. 3., the multiplication is to obtain an $\mathbb{R}^{N*N}$ relevance matrix. The diagonal of the matrix A is the node similarity of S_1, S_2, S_3 from top to bottom. The side of this diagonal line is the relevance among them. We take a Softmax on the $3N$ dimension.

$$A_{i,j} = S_i^T \otimes S_j, A_{i,j} \in \mathbb{R}^{N*N}, A \in \mathbb{R}^{3N*3N}, i/j \in \{0,1,2\} \tag{3}$$

Mask. We apply a learnable mask and multiply it into the adjacency matrix to enhance the matrix learning ability and latent feature expression. Here, we multiply a Global Mask G_{mask} to a data-driven graph to give the Adaptive Graph more spatial restriction and let the model learn the underlying relationship between the static graphs and the adaptive graph. Here × represents the Hadamart product.

$$G' = G_{mask} \times G,\ D' = D_{mask} \times D,\ A' = G_{mask} \times A \tag{4}$$

4.3 Graph Layer

In the Graph Layer, we capture the hierarchical and temporal dependencies through the cascading structure. The whole model contains J cascading Graph Layers. Each Graph Layer contains several paralleled Graph Modules, Fig. 1 provides a general structure. This paralleled structure captures the unique pattern for different data pieces. Before the data was fed into the first Graph Layer, we first map the input S into higher embedding space $S' \in \mathbb{R}^{N*T*C}$, the input of the first Graph Layer $L^1 = S'$. The j-th Graph Layer $L^j = \|(M_1^j, M_2^j, M_3^j, ..., M_{T-(j\times2)}^j)$, j $\in \{1, 2, .., J\}$, $\|$ means the concatenation on the time dimension. The i-th Graph Module $M_i^j \in \mathbb{R}^{N*t*C}$, here t = 3 is the length of sliding window. For example, the first Graph Layer has $10 = 12 - (1 \times 2)$ paralleled Graph Modules.

$$M_{i+1}^j = \alpha * M_i^j + M_{i+1}^j,\ L^{j+1} = \beta * AdpAvgPool(L^j) + L^{j+1} \tag{5}$$

To save the long-term information hierarchically and temporally, we combine the input of the last layer with the current layer to make the final input and add the previous Graph Module input to the current one. We add control variables α and β to control the value. *AdpAvgPool* is an Adaptive Average Pool to adjust the tensor shape.

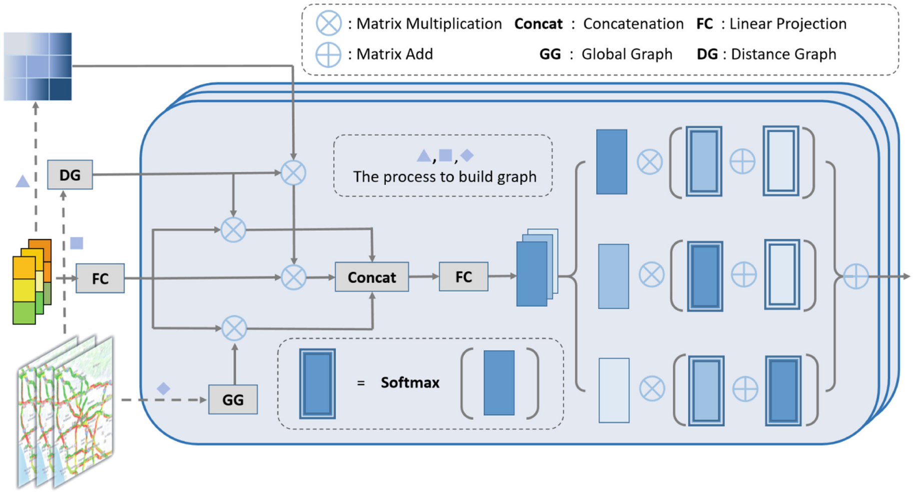


Fig. 3. The diagram of the Graph Block.

4.4 Graph Module and Graph Block

We develop delicate Graph Blocks to aggregate and capture the subtle relationships of features. Figure 3 shows the structure of Graph Block. Each Graph Module contains B Graph Blocks. In each Graph Block, we have K copied input, here $K = 3$, the input of the Graph Module M_i^j is the input of the first Graph Block P^1. These copies are multiplied with masked graphs, respectively. Next, these parts are concatenated together and mapped to a higher embedding space. Then the feature is divided into K branches on the channel dimension, $\{p_1^{g'}, p_2^{g'}, p_3^{g'}, ..., p_K^{g'}\} = P^{g'}$, each branch is multiplied by the sum of other branches, which are normalized by the softmax function. The output of P^1 is the input of P^2, after B iterations of Eqs. 6 to 7, we get P^1 to P^B. By using the MaxPool, we get the output of the Graph Module M_i^j, which is the input of Graph Module M_i^{j+1}. The input of (j+1)-th Graph Layer $L^{j+1} = \|(M_1^{j+1}, M_2^{j+1}, M_3^{j+1}, ..., M_{T-(j\times 2)}^{j+1})$. g $\in \{1, 2, .., B\}$ means the g-th Graph Block.

$$P^{g'} = W^i \otimes \|(A'P^g, D'P^g, G'P^g) + b^g, \qquad (6)$$

$$P^{g+1} = \frac{1}{2K}\sum_{m}^{K}\sum_{n\neq m}^{K} p_m^{g'} \otimes softmax(p_n^{g'}) \qquad (7)$$

$$M_i^{j+1} = MaxPool(P^1, P^2, ..., P^B) \qquad (8)$$

4.5 Attention Module

The Attention Module is designed to replace the simple Fully Connected Layer to capture the global feature from previous Graph Layers. We got the output of Graph Layers $L^J = H \in \mathbb{R}^{N*F'}$. Then we map the feature into a higher embedding space. Q:$\{Q_1, Q_2, .., Q_h\}$, K:$\{K_1, K_2, .., K_h\}$, V:$\{V_1, V_2, .., V_h\}$, h is number of heads, we set $h = 12$ in following experiments. This manipulation provides the self-attention [14] value between nodes and lever each node's value to the sum value of its neighbor to provide global information. $\sqrt{\frac{h}{F''}}$ is normalization factor. By concatenating the output of h heads, we get the final output. We take the softmax at the N dimension.

$$Q = H \otimes W_q + b_q,\ K = H \otimes W_k + b_k,\ V = H \otimes W_v + b_v \tag{9}$$

$$\hat{Y_i} = softmax\,(\sqrt{\frac{h}{F''}} * (\,Q_i^T \otimes K_i\,)) \otimes V_i \tag{10}$$

5 Experiments

5.1 Data Preparation

We use public traffic datasets PEMS03, PEMS04, PEMS07, and PEMS08 released by STSGCN [12]. The gap among time steps is 5 min. The whole day has 288 points in total. It has flow, occupation, and speed values at every time step on every location point. In this work, we followed previous work [10,12], only using the traffic flow value to forecast future traffic flow. More specifically, we use past 1 h, 12 continuous data to predict future 1 h, 12 continuous data. The spatial adjacency networks for each dataset is constructed by existing road network based on connectivity and distance.

5.2 Experiment Settings

The best model on these four datasets consists of $J = 4$ Graph layers, each Graph Module contains $B = 3$ Graph Blocks. The σ in the Distance Graph is 1. Before the input data is fed into the first Graph Layer, the channel has been increased from 3 to 64, $S' \in \mathbb{R}^{N*12*64}$, N is the number of nodes, it depends on datasets. In the Graph Block, W^i is $\mathbb{R}^{192*192}$. In Attention module, W_q and W_k are $\mathbb{R}^{256*864}$, W_v is $\mathbb{R}^{256*12}$. We choose Huber loss [8] as the loss function. To evaluate the effectiveness of the proposed model, Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and Root Mean Squared Error (RMSE).

Table 1. Performance evaluation results

Dataset	Metric	ARIMA [3]	STGCN [21]	GraphWaveNet [19]	STSGCN [12]	STGODE [5]	STFGNN [10]	**Our model**
PEMS03	MAE	33.51	17.49	19.85	17.48	16.50	16.77	**15.88**
	MAPE (%)	33.78	17.15	19.31	16.78	16.69	16.30	**15.77**
	RMSE	47.59	30.12	32.94	29.21	27.84	28.34	**26.81**
PEMS04	MAE	33.73	22.70	25.45	21.19	20.84	19.83	**19.47**
	MAPE (%)	24.18	14.59	17.29	13.90	13.77	13.02	**12.47**
	RMSE	48.80	35.55	39.70	33.65	32.82	31.88	**31.51**
PEMS07	MAE	38.17	25.38	26.85	24.26	22.99	22.07	**21.95**
	MAPE (%)	19.46	11.08	12.12	10.21	10.14	**9.21**	9.28
	RMSE	59.27	38.78	42.78	39.03	37.54	35.80	**35.53**
PEMS08	MAE	31.09	18.02	19.13	17.13	16.81	16.64	**16.49**
	MAPE (%)	22.73	11.40	12.68	10.96	10.62	10.60	**10.43**
	RMSE	44.32	27.83	31.05	26.80	25.97	26.22	**25.73**

5.3 Experiment Result

In Table 1, we find that our model outperforms other models on four datasets with different metrics, except the MAPE in PEMS07, which is slightly larger than that of STFGNN.

The traditional machine learning method ARIMA does not consider spatial dependency, whereas deep learning models can take advantage of spatial-temporal information. The relatively poor performance of GraphWaveNet reveals its struggle because it can not stack its spatial-temporal layers and enlarge receptive fields of 1D CNN concurrently. STGCN repeatedly use ten layers of Graph Convolution operations to capture the feature, we believe it could cause missing features and it's feature extraction part is not so strong compared with attention-based model. STGODE focuses on a relatively new aspect, the whole model is built on the ordinary equation function to simulate the time series. Due to the unique and effective tensor-based manipulation, it reaches a good performance. Among the deep learning baselines, STSGCN and STFGNN utilize paralleled modules to model spatial-temporal flow data and capture heterogeneity of temporal information, they outperform other models. But they only concentrate on localized spatial-temporal correlations and do not focus on global dependencies enough.

6 Ablation Study

To understand the effectiveness of different techniques in this model, we design seven models. First, the base is a plain model without any techniques. We then add them one by one to the model. Then, we tune the parameters to reach their best performance.

(1) Base: This model only contains the flow channel as an input and then maps it into three channels. In the Graph Block, it only has one branch, $K = 1$. This branch is multiplied by the masked Global Graph, which is the only graph in this model. Also, it does not have intra-interaction, which is the knowledge from previous modules or layers. (e.g., The dashed line in Fig. 1) In the output layer, it uses a simple Fully Connected Layer instead of the Attention Module.

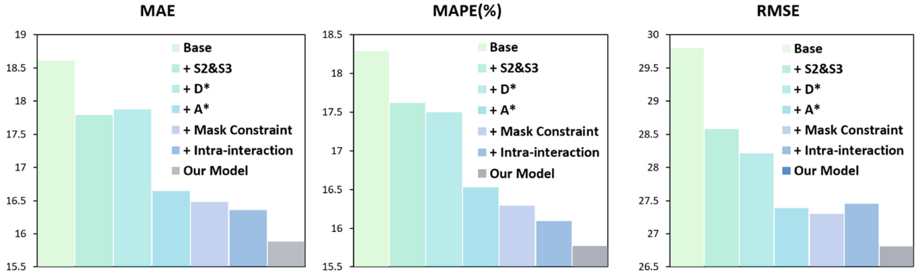


Fig. 4. The result of Ablation Study.

(2) +S2& S3: Beyond (1), we derive the first-order and second-order differencing of flow data and concatenate them to three channels as input.

(3) +D*: We add the masked Distance Graph and have two branches in the Graph Block.

(4) +A*: We add the non-masked Adaptive Graph and have three branches in the Graph Block.

(5) +Mask Constraint: Beyond (4), we multiply the mask of the Global Graph to the Adaptive Graph to get mask constraint.

(6) +Intra-interaction: We add intra-interaction hierarchically and temporally, which is knowledge from previous modules or layers.

(7) Our Model: We replace the simple Fully Connected Layer with the Attention Module.

Figure 4 is the result. We can find out that the various differencing orders of flow data are additional support for feature representation in the traffic flow task. One channel of flow data is not so informative to provide learning features. The first-order and second-order differencing of flow data gives a considerable performance improvement to the model.

The Distance Graph can only provide limited help compared with the previous model. However, the Adaptive Graph provides excellent help. Compared with the static Distance Graph, the adaptive graph find the hidden patterns inside the feature.

The mask constraint between the Global Mask and the Adaptive Graph can improve the model's performance and enhance the matrix learning ability. Even though the Adaptive Graph success in this model, the guide of spatial information is still essential. The spatial information comes from the Global Mask, which contains the real-world location for features. It gives the Adaptive Graph a rational direction to learn.

The hierarchical and temporal intra-interaction between Graph Layers and Graph Modules is also necessary for the model. In most metrics, it improves the performance of this cascading structure. The attention mechanism can obtain features based on the context, but the simple Fully Connected Layer is not strong enough.

7 Conclusion

We propose a novel model that could effectively capture the localized spatial-temporal correlations and take the underlying physical meaning into account by the differencing method. Meanwhile, to solve the problems shown in prior models, we design an adaptive graph for each sliding window, generate a robust graph feature extraction operation, and enhance the interaction between graph modules. Extensive experiments on four real-world datasets show that our model is superior to the existing models. Our proposed model is a general framework for spatial-temporal network data forecasting. It can be applied in many related applications.

References

1. Bai, L., Yao, L., Li, C., Wang, X., Wang, C.: Adaptive graph convolutional recurrent network for traffic forecasting. arXiv preprint arXiv:2007.02842 (2020)
2. Berndt, D.J., Clifford, J.: Using dynamic time warping to find patterns in time series. In: KDD Workshop, Seattle, WA, USA, vol. 10, pp. 359–370 (1994)
3. Box, G.E., Pierce, D.A.: Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. J. Am. Stat. Assoc. **65**(332), 1509–1526 (1970)
4. Cover, T., Hart, P.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theory **13**(1), 21–27 (1967)
5. Fang, Z., Long, Q., Song, G., Xie, K.: Spatial-temporal graph ode networks for traffic flow forecasting. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 364–373 (2021)
6. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 1025–1035 (2017)
7. He, Z., Chow, C.Y., Zhang, J.D.: STCNN: a spatio-temporal convolutional neural network for long-term traffic prediction. In: 2019 20th IEEE International Conference on Mobile Data Management (MDM), pp. 226–233. IEEE (2019)
8. Huber, P.J.: Robust estimation of a location parameter. In: Kotz, S., Johnson, N.L. (eds.) Breakthroughs in Statistics. Springer Series in Statistics, pp. 492–518. Springer, New York (1992). https://doi.org/10.1007/978-1-4612-4380-9_35
9. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
10. Li, M., Zhu, Z.: Spatial-temporal fusion graph neural networks for traffic flow forecasting. arXiv preprint arXiv:2012.09641 (2020)
11. Liu, J., Shahroudy, A., Xu, D., Wang, G.: Spatio-temporal LSTM with trust gates for 3D human action recognition. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 816–833. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_50
12. Song, C., Lin, Y., Guo, S., Wan, H.: Spatial-temporal synchronous graph convolutional networks: a new framework for spatial-temporal network data forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 914–921 (2020)
13. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)

14. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
15. Wang, H., Xu, M., Zhu, F., Deng, Z., Li, Y., Zhou, B.: Shadow traffic: a unified model for abnormal traffic behavior simulation. Comput. Graph. **70**, 235–241 (2018)
16. Williams, B.M., Hoel, L.A.: Modeling and forecasting vehicular traffic flow as a seasonal Arima process: theoretical basis and empirical results. J. Transp. Eng. **129**(6), 664–672 (2003)
17. Wu, N., Wang, J., Zhao, W.X., Jin, Y.: Learning to effectively estimate the travel time for fastest route recommendation. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 1923–1932 (2019)
18. Wu, N., Zhao, X.W., Wang, J., Pan, D.: Learning effective road network representation with hierarchical graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 6–14 (2020)
19. Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C.: Graph wavenet for deep spatial-temporal graph modeling. arXiv preprint arXiv:1906.00121 (2019)
20. Xingjian, S., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., Woo, W.C.: Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In: Advances in Neural Information Processing Systems, pp. 802–810 (2015)
21. Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. arXiv preprint arXiv:1709.04875 (2017)