



Traffic Flow Prediction Based on Attention Mechanism Convolutional Neural Network

Sumin Li, Jing Li, Jie Lan, and Yong Lu^(✉)

School of Information Engineering, Minzu University of China, Beijing 100081, China
{sml, 21302002, 20011437, 2006153}@muc.edu.cn

Abstract. Traffic flow prediction is one of the core issues in the field of transportation planning and management. Traditional traffic flow prediction methods are limited by factors such as data sparsity, long-term interdependencies, and intricate spatiotemporal dynamics. To overcome these challenges, this paper proposes a novel predictive model called ARSTGCN, which incorporates spatiotemporal attention mechanisms and deep learning networks. Firstly, the spatiotemporal attention mechanism assigns attention weights to traffic sensor nodes, enabling the capture of spatiotemporal relationships. Dilation convolution is employed to process the temporal correlation in the data, mitigating concerns of gradient explosion and vanishing during the training of lengthy time series. Secondly, data containing spatiotemporal weight features undergoes input into the graph convolutional network, facilitating the capture of spatial dynamic correlations. The final prediction results are obtained through the utilization of the fully connected layer. Compared with the baseline model on two publicly available datasets, ARSTGCN showed certain advantages.

Keywords: Traffic Flow Prediction · Attention Mechanism · Graph Convolution

1 Introduction

With the continuous development of China's transportation industry and the sharp increase in vehicles, traffic congestion and frequent traffic accidents have emerged as critical challenges confronted by modern cities. To overcome these issues, experts have shifted their focus towards establishing an intelligent transportation system (ITS) [1]. The objective is to develop smart transportation and transportation information technology, enhance the efficiency and precision of traffic control, while optimizing traffic flow, with the ultimate aim of alleviating the prevailing traffic problems.

So far, research on traffic flow prediction has a history of over a decade. Initially, mathematical and physical methods were commonly employed for prediction. The most typical models included the Historical Average (HA) model [2], Vector Autoregressive (VAR) model [3], and Autoregressive Integrated Moving Average (ARIMA) model [4]. However, these models relied on linear data analysis, while traffic flow data is nonlinear and complex. Therefore, statistical-based prediction methods did not adequately fit traffic flow data. Some common machine learning methods used in the field of traffic

flow prediction include Support Vector Machine (SVM), K-Nearest Neighbors (KNN) algorithm, and Kalman filter model [5]. In the 1990s, SVM proposed by Vapnik et al. in the literature [6] has gained considerable attention. SVM is a supervised learning algorithm that separates samples into two categories by defining an optimal boundary. It was employed in traffic flow prediction to overcome the limitations of traditional statistical models in handling nonlinear problems. In literature [7], Zhang et al. proposed a short-term traffic flow prediction method based on Balanced Binary Tree K-Nearest Neighbor Non-parametric Regression. This method utilized clustering and a balanced binary tree structure to establish a case database, aiming to improve prediction accuracy and meet real-time requirements. These machine learning methods offered new avenues for enhancing the accuracy and reliability of traffic flow prediction. However, machine learning models needed to possess good generalization ability when predicting new traffic flow data. Yet, due to the complexity and dynamic nature of the traffic system, the generalization ability of the models was limited across different regions, time periods, or traffic scenarios.

In recent years, with the emergence of deep learning models, researchers have gradually employed deep neural network models for traffic flow prediction. Ma et al. [8] utilized Long Short-Term Memory (LSTM) neural networks for predicting traffic speeds, effectively capturing the temporal correlations in the data flow. Literature [9] combines the convolutional neural network (CNN) and LSTM to extract the spatiotemporal features from multiple perspectives, and the experimental results show that this method can effectively predict traffic information. Although these models have made significant progress in feature extraction, they still cannot represent the true non-Euclidean spatial road network structure [10]. Some scholars have studied the graph convolutional network (GCN). Li et al. [11] proposed the DCRNN model, combining the characteristics of diffusion graph convolution and recurrent neural network to capture the spatiotemporal dependence in traffic data. Yu et al. [12] designed the STGCN model using spatial graph convolution and temporal convolution to effectively model the spatiotemporal dependencies in traffic data and improve traffic prediction accuracy. Subsequently, Guo et al. [13] introduced the ASTGCN model by adding attention mechanisms to the STGCN model. Geng et al. [14] proposed the SMGCN model for predicting the demand of ride-hailing services. This model combines spatiotemporal features and multi-graph convolution operations, aiming to capture the spatiotemporal dependencies in the data related to ride-hailing service demand.

Although existing deep learning methods consider the temporal and spatial correlations, these methods still have shortcomings. Most of these methods rely on LSTM and GRU to capture temporal dependencies, which can easily lead to the problems of gradient explosion and vanishing when dealing with long time series. Additionally, some network models attempt to use stacked one-dimensional convolutions to mitigate the gradient explosion issue, leading to an increase in computational complexity.

To overcome these limitations, fully consider the spatiotemporal dynamic correlation, and further improve the accuracy of traffic flow prediction, this paper reduces the model complexity and improves the operating efficiency by reducing the number of network layers on the basis of the ASTGCN model. In addition, the dilated convolution

is introduced based on the existing baseline model to extract long-time dynamic correlations with fewer network layers and parameters to improve the prediction accuracy. Therefore, this paper proposes a new traffic flow prediction method called ARSTGCN.

2 ARSTGCN Model

2.1 Problem Definition

Traffic flow prediction refers to the use of traffic flow data collected by traffic sensors distributed on the road [15] to predict the future traffic flow of a certain location or area. That is, the number of vehicles passing through the location or area in a certain period of time in the future.

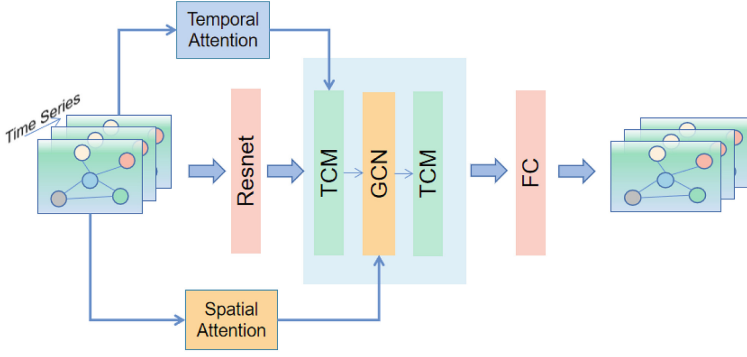
Definition 1 (Traffic Road Network G). The sensors in the road network constitute a topology diagram $G = (V, E, A)$, where V is the set of nodes, indicating the sensor nodes in the road network, the number of nodes is N , $V = \{v_1, v_2, v_3, \dots, v_n\}$; E is the set of edges, indicating the connectivity between the sensors; $A \in R^{N \times N}$ is the adjacency matrix constructed based on the distances between sensors, representing the connectivity between nodes.

Definition 2 (Graph Signal Matrix X). The traffic flow observed on graph G is represented as the graph signal $X \in R^{N \times P}$, where P represents the number of features of each node. The traffic flow prediction problem involves learning a mapping function $f(\cdot)$ for the traffic flow at a given graph G and historical T time period to predict future traffic information for T' . The mapping relationship is shown in Eq. (1).

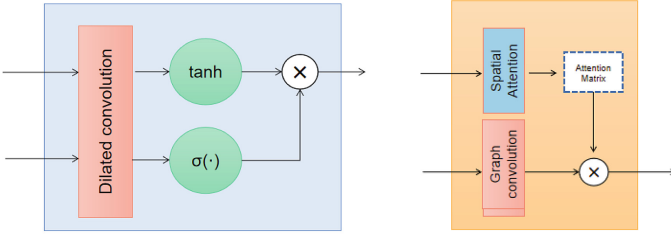
$$(X_{t+1}, X_{t+2}, \dots, X_{t+T'}) = f(G; (X_{t-T+1}, \dots, X_{t-1}, X_t)) \quad (1)$$

2.2 Model Framework

The overall framework of this paper is illustrated in Fig. 1, which consists of a residual layer, a spatiotemporal convolutional block, and a fully connected layer. The spatiotemporal convolutional block is composed of two temporal convolutions (TCM) and one graph convolution (GCN), and incorporates spatiotemporal attention mechanisms to extract features from both the temporal and spatial dimensions. The input traffic time series data first undergoes spatiotemporal attention mechanism to obtain the spatiotemporal correlation matrix. The TCM module introduces dilated convolutions to effectively handle long time series problems. The GCN module uses Chebyshev polynomial as the convolution kernel to reduce the computational complexity.



a. General Framework



b. TCM Block

c. GCN+Spatial Attention

Fig. 1. ARSTGCN model framework

2.3 Spatiotemporal Attention Mechanism

The spatiotemporal attention mechanism [16] is a mechanism used to handle spatiotemporal data. It can learn and capture the correlations between different locations and time points in spatiotemporal data, thereby extracting and expressing important spatiotemporal features.

The temporal attention is used to model the relationships between different time points. It helps us determine which time points are more important for the current task. In time series data, the temporal attention mechanism can learn the evolution and variation patterns of different time points, as well as their impact on specific tasks. The temporal correlation matrix is defined as Eq. (2).

$$I_t = V_t \cdot \sigma((X^T W_1) W_2 (W_3 X) + b_t) \quad (2)$$

In the equations, $V_t, b_t \in R^{T \times T}$, $W_1 \in R^N, W_2 \in R^{F \times N}, W_3 \in R^F$ are the learnable parameters. $X \in R^{N \times F \times T}$ represents all sequence data, where N is the number of nodes, F is the number of data types, and T is the length of time. σ denotes the activation function. Based on the temporal correlation matrix, we calculate the time attention matrix between

nodes i and j . such as the following Eq. (3).

$$I'_{t(i,j)} = \frac{\exp(I_{t(i,j)})}{\sum_{j=1}^N \exp(I_{t(i,j)})} \quad (3)$$

where $I_{t(i,j)}$ represents the element in the i -th row and j -th column of the temporal correlation matrix, which indicates the degree of temporal correlation between the two nodes. $I'_{t(i,j)}$ represents the calculated time attention matrix.

Similarly to the calculation of the time attention matrix, the spatial attention is used to model the relationships between different spatial locations. It helps us determine which spatial positions are more important for the current task. The equations for calculating the spatial correlation matrix and the spatial attention matrix are shown in (4) and (5) respectively. In these equations, $V_s, b_s \in \mathbb{R}^{N \times N}$, $U_1 \in \mathbb{R}^T$, $U_2 \in \mathbb{R}^{F \times T}$, $U_3 \in \mathbb{R}^F$ are all parameters to be learned, $X \in \mathbb{R}^{N \times F \times T}$ represents all sequence data. $I_{s(i,j)}$ denotes the element in the i -th row and j -th column of the spatial correlation matrix, which represents the degree of spatial correlation between the two nodes. $I'_{s(i,j)}$ represents the calculated spatial attention matrix.

$$I_s = V_s \cdot \sigma((XU_1)U_2(U_3X)^T + b_s) \quad (4)$$

$$I'_{s(i,j)} = \frac{\exp(I_{s(i,j)})}{\sum_{j=1}^N \exp(I_{s(i,j)})} \quad (5)$$

2.4 Time Convolution

Recurrent neural networks have significant advantages in handling time series problems. However, they can suffer from the issues of vanishing and exploding gradients when dealing with long time series data. In addition, when dealing with complex problems, it requires adding multiple layers of convolution to capture the long-term dependencies in the time series. Dilated convolution [17] has a major advantage in that it can increase the receptive field without introducing additional parameters and computational complexity. By introducing a dilation rate parameter, the receptive field of the convolution kernel can be expanded. This allows the network to capture both local detailed features and larger contextual information. Therefore, in this paper, dilated convolution is used to extract temporal dynamic correlations, and the network architecture is shown in Fig. 1b. Given the input time series data $X \in \mathbb{R}^{N \times T \times F}$, where N is the number of nodes, T is the time steps, and F is the number of features, the temporal convolution is defined as Eq. (6).

$$\text{TCM}(X) = g(\text{Conv}(X)) \odot \sigma(\text{Conv}(X)) \quad (6)$$

where, $g(\cdot)$ and $\sigma(\cdot)$ represent the activation functions tanh and sigmoid, respectively. $\text{Conv}(\cdot)$ denotes one-dimensional dilated convolution, and \odot represent Hadamard multiplication. After obtaining the temporal feature correlations through the temporal convolutional layer, the spatial features are learned using the graph convolutional layer.

2.5 Graph Convolution

Graph convolution is mainly used to capture spatial dependencies among different nodes in a graph. Graph Convolutional Networks (GCNs) implement convolutional operations on topological graphs based on graph theory [18]. In graph convolution, each node has its own feature vector and is connected to its neighboring nodes to form an adjacency relationship. The goal of graph convolution is to update the features of each node by incorporating the feature information from its neighboring nodes. In graph convolution, the first step is to transform the adjacency matrix into a Laplacian matrix, which is defined as Eq. (7).

$$L = I_N - D^{-\frac{1}{2}}AD^{-\frac{1}{2}} \quad (7)$$

In the equation, I_N is the $N \times N$ identity matrix, A is the adjacency matrix of the graph G , and D is the degree matrix of the adjacency matrix A . The Laplacian matrix L is subjected to eigendecomposition, which decomposes it into the form $L = \beta\Lambda\beta^T$. Here, Λ is a diagonal matrix containing the eigenvalues, and β is a matrix containing the corresponding eigenvectors. The obtained parameters are used to perform graph convolution on the input time series sequence, and the equation is as follows.

$$g_\theta * x = g(L)x = \beta\Theta(\Lambda)\beta^T X_{in} \quad (8)$$

$g_\theta *$ is the graph convolution operator. Due to the computational complexity of Eq. (8), Hammond [19] et al. proposed that using Chebyshev polynomials to effectively solve this problem. Therefore, Eq. (8) can be approximated and simplified to Eq. (9).

$$g_\theta * x = g(L)x \approx \sum_{k=0}^{K-1} \theta_k T_k(\tilde{L})x \quad (9)$$

$$\tilde{L} = \frac{2L}{\lambda_{max} - I_N} \quad (10)$$

where, θ_k represents trainable parameters, $T_k(\tilde{L})$ are the coefficients of the Chebyshev polynomials. The expression for \tilde{L} is defined as Eq. (10). λ_{max} is the maximum eigenvalue of the Laplacian, and K is the size of the convolutional kernel. In order to effectively learn spatial-temporal correlations, attention is incorporated into the convolutional operation in this paper, as shown in Fig. 1c. Therefore, Eq. (9) is modified to Eq. (11).

$$g_\theta * x = g(L)x \approx \sum_{k=0}^{K-1} \theta_k \left(T_k(\tilde{L}) \odot I'_{s(i,j)} \right) x \quad (11)$$

3 Experiment and Result Analysis

3.1 Dataset

This experiment uses two publicly available datasets, namely PEMS4 and PEMS8. PEMS4 and PEMS8 are datasets from the Los Angeles area in California, USA. Table 1 provides detailed information about these datasets. The distinctive feature of

these two datasets is that they have a relatively fine granularity, with a statistical time interval of 5 min for each data group. In this experiment, one hour of traffic flow data is taken as the historical time period to predict the future traffic information for the next hour, with 12 data records considered as one time step. The datasets are divided into training, validation, and testing sets with a ratio of 6:2:2, respectively. The input data is processed using the Z-score method.

Table 1. Dataset description.

Datasets	Nodes	Timesteps	Time frame
PEMSD4	307	16992	2018.1.1—2018.2.28
PEMSD8	170	17856	2016.7.1—2016.8.31

3.2 Experimental Setting

The experiment was conducted in a Windows system environment, using the PyTorch framework. The computer processor used was Intel(R) Core(TM) i5-1135G7 @ 2.40 GH, with CUDA 10 and Python 3.7.

In the experiment, the Adam optimizer was employed, with a learning rate of 0.001. The batch size was set to 32. To measure the predictive performance of different methods, the chosen evaluation metrics were Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). Smaller values of these metrics indicate better prediction performance of the model. The formula are given as Eq. (12) and (13) below.

(1) Mean Absolute Error:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i| \quad (12)$$

(2) Root Mean Square Error:

$$\text{RMSE} = \left(\frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i|^2 \right)^{\frac{1}{2}} \quad (13)$$

In the equations, y_i represents the true traffic flow and \hat{y}_i represents the predicted traffic flow for the i -th sample, where n represents the number of samples.

3.3 Baselines

This study compares the proposed model with six benchmark models for traffic flow prediction.

(1) Historical Average: This is a simple traffic flow prediction model that uses the historical average to predict future traffic flow.

- (2) LSTM: LSTM is a type of recurrent neural network model that can capture long-term dependencies in time series data. It is used for predicting future traffic flow.
- (3) T-GCN: T-GCN is a traffic flow prediction model that combines temporal information and graph convolution. It can effectively learn the spatiotemporal relationships and evolution patterns of traffic flow for future predictions.
- (4) STGCN: STGCN is a spatial-temporal method for traffic flow prediction. It leverages graph convolutional operations to model the spatial and temporal dependencies in traffic data.
- (5) ASTGCN: ASTGCN is a traffic flow prediction model that utilizes attention mechanisms along with spatial and temporal graph convolutional networks.
- (6) Graph WaveNet [20]: Graph WaveNet is a traffic flow prediction model based on graph convolutional neural networks and WaveNet.

3.4 Experimental Results

Table 2 displays the prediction results of various models for a 60-min prediction on the PEMS4 and PEMS8 datasets. The benchmark experimental results in this study referenced the experimental data from ASTGCN [13]. The performance of the ARSTGCN model showed improvement on both datasets. Compared to ASTGCN, the proposed model improved the MAE and RMSE metrics by 1.7%, 2.4%, 8.02%, and 3.6% on PEMS4 and PEMS8, respectively. Compared to STGCN, the proposed model achieved an improvement of 10.1%, 10.1%, 6.5%, and 1.2% on PEMS4 and PEMS8 for the MAE and RMSE metrics, respectively.

Table 2. Performance comparison of different methods for one-hour traffic prediction on PEMS4 and PEMS8

Dataset	Metric	HA	LSTM	T-GCN	STGCN	Graph WaveNet	ASTGCN	ARSTGCN
PEMS4	MAE	36.76	36.76	28.04	25.15	25.45	23.00	22.59
	RMSE	54.14	45.82	41.21	38.29	39.70	35.28	34.39
PEMS8	MAE	29.52	23.18	24.01	18.88	19.13	19.19	17.65
	RMSE	44.03	36.96	33.98	27.87	31.05	29.20	27.52

3.5 Run Time Comparison

In the experiment, the computational time of the proposed ARSTGCN model was compared with the benchmark models STGCN and ASTGCN on the two datasets. Since STGCN and ASTGCN involve stacking multiple layers in their network, it increases the computational complexity. In this study, some complex network layers were reduced in the proposed model. As shown in Fig. 2, it can be observed that compared to STGCN and ASTGCN models, the ARSTGCN model had the least computational time on both datasets, resulting in significant speed improvements.

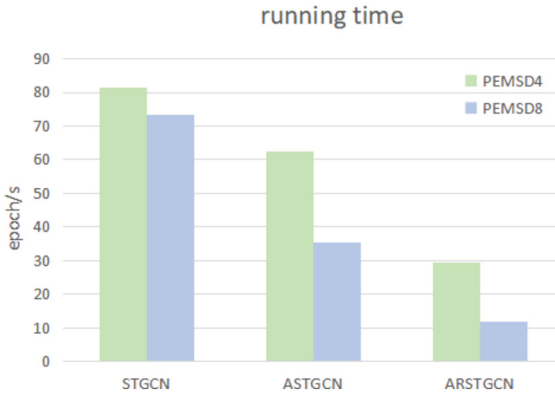


Fig. 2. Comparison of running times on dataset PEMSD4 and PEMSD8

4 Conclusion

This paper introduces the spatiotemporal attention mechanism to capture the temporal and spatial correlation respectively, reducing the defect of common graph convolution in extracting data features. Additionally, the paper introduces dilated convolutions to the existing benchmark models, which can increase the range of receptive fields without introducing extra parameters and computational complexity. This effectively solves the issues of gradient vanishing and explosion that often occur in long time series. This model is optimized in terms of network complexity, which not only improves accuracy but also greatly reduces time cost operations.

References

1. Tanimoto, J., An, X.: Improvement of traffic flux with introduction of a new lane-change protocol supported by intelligent traffic system. *Chaos, Solitons Fractals: Interdisc. J. Nonlinear Sci. Nonequilibrium Complex Phenomena* **122**, 1–5 (2019)
2. Smith, B.L., Williams, B.M., Oswald, R.K.: Comparison of parametric and nonparametric models for traffic flow forecasting. *Transp. Res. Part C* **10C**(4), 303–321 (2002)
3. Zivot, E., Wang, J.H.: Vector autoregressive models for multivariate time series. In: *Modeling Financial Time Series with S-PLUS®*, pp. 385–429 (2006). https://doi.org/10.1007/978-0-387-32348-0_11
4. Ahmed, M.S., Cook, A.R.: Analysis of freeway traffic time-series data by using box-jenkins techniques. *Transp. Res. Rec.* **722**, 1–9 (1979)
5. Okutani, I., Stephanedes, Y.J.: Dynamic prediction of traffic volume through Kalman filtering theory. *Transp. Res. Part B: Methodol.* **18**(1), 1–11 (1984)
6. Vapnik, Vladimir. *The nature of statistical learning theory*. Springer science & business media, 1999
7. Xiaoli, Z., Guoguang, H., Huapu, L.: Short-term traffic flow prediction method based on k-neighborhood nonparametric regression. *J. Syst. Eng.* **24**(02), 178–183 (2009)
8. Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y.: Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transp. Res. Part C* **54**, 187–197 (2015)

9. Yao, H., Wu, F., Ke, J., et al.: Deep multi-view spatial-temporal network for taxi demand prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1 (2018)
10. Bruna, J., Zaremba, W., Szlam, A., et al.: Spectral networks and locally connected networks on graphs. In: 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, 14–16 April, 2014, Conference Track Proceedings, pp. 1–10 (2014)
11. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: data-driven traffic forecasting. In: Proceedings of the 6th International Conference on Learning Representations, pp. 1295–1302 (2018)
12. Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence, pp. 3634–3640 (2018)
13. Guo, S., Lin, Y., Feng, N., Song, C., Wan, H.: Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the 33th AAAI Conference on Artificial Intelligence, pp. 922–929 (2019)
14. Geng, X., Li, Y., Wang, L.: Spatiotemporal multi-graph convolution network for ride hailing demand forecasting. In: Proceedings of the 33th AAAI Conference on Artificial Intelligence, vol. 33, pp. 3656–3663 (2019)
15. Tedjopurnomo, D.A., Bao, Z.F., Zheng, B.H., et al.: A survey on modern deep neural network for traffic prediction: trends, methods and challenges. *IEEE Trans. Knowl. Data Eng.* **34**(4), 1544–1561 (2022)
16. Li, Y., et al.: Transformer with sparse attention mechanism for industrial time series forecasting. *J. Phys. Conf. Ser.* **2026**(1), 012036 (2021)
17. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. In: 4th International Conference on Learning Representations, San Juan, ICLR, pp. 1–13 (2016)
18. Kipe, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, Toulon:ICLR, pp. 1–14 (2017)
19. David, A., Hammond, K., Van Der Broek, P., Remi Gribonval, C.: Wavelets on graphs via spectral graph theory. *Appl. Comput. Harmonic Anal.* **30**(2), 129–150 (2011)
20. Wu, Z., Pan, S., Long, G., et al.: Graph WaveNet for deep spatial-temporal graph modeling. *CoRR*, abs/1906.00121 (2019)