Spatio-Temporal Pyramid Networks for Traffic Forecasting

Jia Hu, Chu Wang, and Xianghong Lin(✉)

College of Computer Science and Engineering, Northwest Normal University, Lanzhou, China
linxh@nwnu.edu.cn

Abstract. Traffic flow forecasting is an important part of smart city construction. Accurate traffic flow forecasting helps traffic management agencies to make timely adjustments, thus improving pedestrian travel efficiency and road utilization. However, this work is challenging due to the dynamic stochastic factors affecting the variation of traffic data and the spatially hidden behavior. Existing approaches generally use attention mechanism or graph neural networks to model correlation in temporal and spatial terms, and despite some progress in performance, they still ignore a number of practical situations: (1) Anomalous data due to traffic accidents or traffic congestion can affect the accuracy of modeling in the current moment and further create potential optimization problems for model training. (2) According to the directedness of the road, the hiding behavior between nodes should also be unidirectional and dynamic. In this paper, we propose a dynamic graph network with a pyramid structure, named PYNet, and use it for traffic flow forecasting tasks. Specifically, first we propose the Pyramid Constructor for transforming multivariate time series into a pyramid network with a multilevel structure, where the higher the level, the larger the range of time scales represented. Second, we perform Trend-Aware Attention top-down in the pyramid network, which gradually enables the lower-level time series to learn their long-term dependence in multiples, and effectively reduces the impact of outliers. Furthermore, to fully capture the hidden behavior in the spatial dimension, we learn an adaptive unidirectional graph and perform forward and backward diffusion convolution on the graph. Experimental results on two types of datasets show that PYNet outperforms the state-of-the-art baseline.

Keywords: Traffic flow forecasting · Spatio-temporal data · Pyramid structure

1 Introduction

In recent years, many countries are focusing on the development of Intelligent Transportation Systems (ITS). Traffic flow forecasting, route planning and vehicle scheduling are important components of ITS, and they work together to

J. Hu and C. Wang—Equal contribution.

D. Koutra et al. (Eds.): ECML PKDD 2023, LNAI 14169, pp. 339–354, 2023.
https://doi.org/10.1007/978-3-031-43412-9_20

improve the transportation service system of cities. In these applications, route planning and vehicle scheduling are based on the traffic conditions of roads, so traffic flow forecasting is the cornerstone of ITS. In this paper, we use the historical traffic data of roads to forecast the future traffic conditions. Traffic data is a time series data, collected by sensors deployed in the traffic network at a fixed continuous period of time. Early researchers applied the classical time series models Vector Auto-Regression (VAR) [15], Autoregressive Integrated Moving Average model (ARIMA) [8] to forecast future traffic conditions, they are limited by the assumption of linearity and smoothness of the data, and traffic data are usually unsteady and nonlinear, so these methods perform poorly. Deep learning methods based on Recurrent Neural Networks (RNN) [3,4,6] are not subject to these limitations, therefore they are widely used to extract long and short term dependencies in time series. A limitation of these methods is the inability to model spatial correlations in traffic networks, and with the deeper understanding of the problem and the development of graph neural networks (GNN) [7], researchers have proposed a spatial-temporal forecasting framework based on graph neural networks [14,23,26], which construct traffic graphs by taking sensors deployed in traffic networks as nodes and road networks or node distances as edges, updating node characteristics through information transfer effects between nodes. The advantage of these GNN-based methods is that they can handle data with a non Euclidean structure, which makes up for the fact that CNN-based methods [27] can only handle data with a grid structure. While having shown the effectiveness of introducing the graph structure of data into a model, but there is still a lack of satisfactory progress in accurate and long term traffic forecasting, which is mainly due to the following two challenges:

First, unexpected events in the road such as traffic accidents can cause transient anomalies in the traffic data, which may pose potential optimization problems in the training of the model if they are ignored. For instance, most current studies use attention mechanism or CNN to model temporal correlation. The attention mechanism obtains the similarity between node pairs in the form of point-to-point, which will incorrectly update the node features if there exists anomalous data and further cause error accumulation. CNN updates node features by aggregating local contextual information, which can weaken the effect brought by outliers. Considering the multi-scale nature of time series and the design of convolution kernel size, it is difficult to solve this problem with a single convolution layer.

Second, roads in the traffic network are unidirectional, which means that the impacts from traffic conditions on upstream roads are transmitted to downstream roads in the future and continue to spread dynamically over time. The distance-based adjacency matrix defines this diffusion relationship based on the distance of the road network, ignoring the hidden spatial correlation in the traffic network. Therefore, we propose to learn a dynamic directed graph to maintain the hidden property of state transfer between nodes, and in addition, if the dataset further provides information on the structure and distance between nodes, we expect

the dynamic directed graph to easily incorporate this information to generate a more comprehensive representation of node embeddings and spatial matrices.

To solve the above challenges, we propose a new pyramid network for spatial-temporal forecasting, which we call PYNet, which mainly consists of three parts: Pyramid Constructor, Trend-Aware Attention and Diffusion Graph Convolution Network. Pyramid Constructor is based on CNN and is used to transform the input time series into a pyramid network with a multi-leveled structure, and can customize the time range of trend blocks in different levels (It means that the features of several consecutive time steps are aggregated). We then perform Trend-Aware Attention top-down, computing the similarity between trend blocks with different time scales in a local context, which allows not only the lower-level time series to receive several times the perceptual field, but also further attenuates the impact of outliers. In addition, we learn a dynamic directed graph that preserves the one way hidden relationship between nodes in the traffic network, and further, we describe this hidden relationship as a diffusion process of nodes over spatially and capture the potential spatial correlation by diffusion convolution. In summary, we summarize the contributions of this paper as follows:

- We propose a pyramid network for spatial-temporal forecasting tasks, named PYNet, which initializes the input time series into a pyramid network with a multi-leveled structure through the Pyramid Constructor. The trend blocks in the bottom-up levels of the pyramid represent progressively larger time ranges, and such time ranges are customizable.
- We perform Trend-Aware Attention and Diffusion Graph Convolution Network top-down in a pyramid network. The former computes the similarity between trend blocks in local context and gives several times the perceptual field to the lower-level trend blocks, which reduces the impact of outliers. The latter preserves the hidden spatial directed relations by performing diffusion GCN on the adaptive directed graph.
- We evaluate the performance of PYNet on four real-world datasets, and the experiments show that PYNet outperforms all the baseline.

2 Related Work

2.1 Traffic Forecasting

Traffic forecasting is an important component of intelligent transportation systems and has been widely studied in the last decades [10,14,23,26,27,29]. Earlier studies mainly used statistical methods, such as VAR [15], ARIMA [8], which rely on the assumption of linearity of the data and, without doubt, perform poorly when dealing with nonlinear traffic data. With the development of deep learning, recurrent neural networks [3,4,6], which ignore the smoothness assumption, have been successfully applied to time series modeling. To capture spatial correlations, [24,25,27,30] used CNNs to model spatial with regular grid structure, but were powerless for traffic networks with non-Euclidean spatial structure. With the

evolution of graph neural networks, it has become the best method to model the spatial correlation of traffic data, for example, DCRNN [14] uses diffusion GCN to capture the diffusion phenomenon of traffic flow in spatial terms and applies GRU to capture the temporal correlation. Graph WaveNet [23] modeled spatial and temporal correlations using GCNs and temporal convolution networks (TCNs), respectively, and [10,19,22,26] and other studies modeled spatial correlations based on GCNs. With the birth of Transformer [21], GMAN [29], ASTGCN [5], and ST-GRAT [17] introduced attention mechanisms into spatial-temporal modeling and further improved the forecasting accuracy.

If the spatial correlation of traffic networks is modeled using graph neural networks, then there is no doubt that the construction of the adjacency matrix is extremely important. DCRNN [14] computes the road network distance between sensors and uses it as a weight between nodes by means of a thresholded Gaussian kernel function. To react to hidden correlations in spatial, some works [16,23] proposed adaptive adjacency matrix to describe such potential spatial correlations and can be learned by end-to-end. Further, DGCRN [10], MTGNN [22] set the adaptive adjacency matrix as a directed graph, which means that a change in the state of one node leads to a change in the state of other nodes, which brings the learning of adjacency matrix to a new level. In addition, some studies have proposed a data-driven spatial heterogeneity graph based on adding connections between functionally similar regions, [9,12] proving its effectiveness, but it is static and still requires parameters to support training in the training of the model.

2.2 Graph Neural Network

The main idea of graph neural networks is to update node states through the information transfer effect between nodes, which has been a great success in dealing with spatial dependence between entities in a network and is now successfully applied to various tasks such as node classification [18] and link forecasting [31]. Various types of variants of GNN have been developed, such as GCN, Graph Attention Network (GAT), and there are two types of GCN, spatial GCN and spectral GCN. Spatial GCN on the neighboring nodes of the target node directly perform convolution filters, the spectral GCN defines the convolution in the spectral domain [13], which is firstly introduced in [1]. GAT introduces the attention mechanism into GNNs and uses node features to autonomously learn the weights between node pairs. Recently, spatial-temporal graph neural networks [2,28] have been introduced to traffic forecasting for capturing spatial-temporal correlations in traffic data, such as the STGNN, DGCRN replacing the fully connected layer in recurrent neural networks with GCNs, and STJGCN [28] constructing joint graph convolution layers between any two time steps. In addition, some works [29] learn the spatial embedding representation of each node by graph embedding methods such as node2Vec and deepWalk to further improve the efficiency of information transfer between node pairs in spatial.

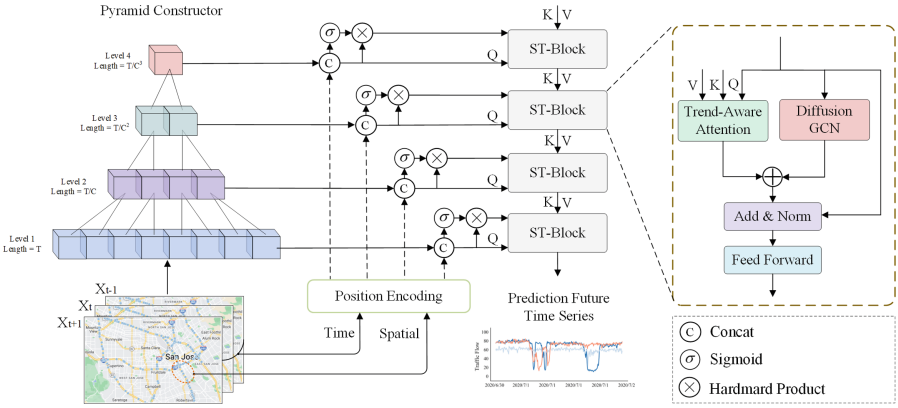

Fig. 1. The framework of PYNet.

3 Preliminary

We denote the traffic data recorded by N sensors at time t as traffic signals, C is the number of signals and the signals can be traffic volume, traffic speed, etc. The traffic forecasting problem aims to learn a function f that maps the traffic conditions at time step P of history to the next time step Q:

$$[\mathrm{X}_{t-P+1}, \mathrm{X}_{t-P+2}, \cdots, \mathrm{X}_t] \xrightarrow{f(\cdot)} [\mathrm{X}_{t+1}, \mathrm{X}_{t+2}, \cdots, \mathrm{X}_{t+Q}] \tag{1}$$

4 Methodology

In this section, we will introduce our proposed model in detail. The overall framework of our proposed model is shown in Fig. 1.

PYNet first takes multivariate time series and passes them through the Pyramid Constructor to obtain a pyramid network with a multi-level structure (the higher the level, the larger the range of time scales), and then adds learnable location codes to each level to facilitate labeling level structures with different scale information. Finally, the top-down stacked Spatial-Temporal Block (ST-Block), which consists of Trend-Aware Attention and Diffusion GCN, in the pyramid structure. Trend-Aware Attention uses both low level and high level features as common inputs, with the aim of enabling each trend block at the low level (aggregated by multiple time steps) to share the long term horizon represented at the high level. Diffusion GCN describes the behavior on spatial as a diffusion process of directed graphs and performs diffusion convolution operations on adaptive directed graphs.

4.1 Pyramid Constructor

Patterns in time series may evolve with time significantly due to various events, e.g. holidays and extreme weather, so whether an observed point is an anomaly,

change point or part of the patterns is highly dependent on its surrounding context. Hence, the independent time steps in the original time series cannot reflect the anomalous information of the data. In order to make full use of the contextual information and reduce the loss caused by data anomalies, we use Pyramid Constructor to obtain a pyramid network with a multi-level structure, which has two advantages: (1) Different levels of time scales can be customized, such that, bottom-up each trend block (i.e., features aggregated over several consecutive time steps) can be considered as hourly, daily and monthly features. (2) There is better fault tolerance in the face of anomalies. The higher the level of the hierarchy, the larger the range of time scales of the trend blocks, then the impact caused by the anomalies is limited.

Given the length T multivariate time series and a set of convolution layers $F^{CNN}(\cdot)$, then each level of the pyramid structure can be defined as:

$$\mathrm{X}_L = F_L^{CNN}(\mathrm{X}_{L-1}, \Theta_L) \in \mathbb{R}^{T_{L-1}/C_L \times N \times D} \tag{2}$$

We take the time series $\mathrm{X}_{L-1} \in \mathbb{R}^{T_{L-1} \times N \times D}$ at the L-1 level and pass it through the standard convolution layer $F_L^{CNN}(\cdot)$ to obtain the time series representation $\mathrm{X}_L \in \mathbb{R}^{T_{L-1}/C_L \times N \times D}$ at the Lth level, where Θ_L corresponds to the parameters of the convolution layer and C_L is the size and step size of the convolution kernel.

4.2 Trend-Aware Attention

In the traditional attention mechanism, the similarities between queries and keys are computed based on their point-wise values without fully leveraging local context information. Query-key matching agnostic of local context may confuse the self-attention module in terms of whether the observed value is an anomaly, change point or part of patterns, and bring underlying optimization issues. Thus, we perform top-down attention mechanisms between adjacent levels of the pyramid, which has two advantages: (1) Compute the similarity between query and key in a local context, which reduces the impact caused by anomalies. (2) Key and value have longer time range information than query, and the top-down attention mechanism will gradually make the lower-level time series learn its own C_L-fold long term dependence until the update of the original time series is completed.

Given the time series of two adjacent levels $\mathrm{X}_L \in \mathbb{R}^{T_L \times N \times D}$ and $\mathrm{X}_{L+1} \in \mathbb{R}^{T_{L+1} \times N \times D}$, which $T_{L+1} = T_L/C_L$. The operation of Trend-Aware Attention can be expressed as follows:

$$\mathrm{Q}_L^{(h)} = \mathrm{Softmax}\left(\frac{(\mathrm{X}_L^{(h)}\mathrm{W}_Q^{(h)})(\mathrm{X}_{L+1}^{(h)}\mathrm{W}_K^{(h)})^T}{\sqrt{d_h}} + \mathrm{W}_{adp}\right)(\mathrm{X}_{L+1}^{(h)}\mathrm{W}_V^{(h)}) \tag{3}$$

$$\mathrm{Q}_L = \mathrm{MLP}(\mathrm{Concat}(\mathrm{Q}_L^{(1)}, \mathrm{Q}_L^{(2)}, \ldots, \mathrm{Q}_L^{(H)})) \tag{4}$$

where $\mathrm{W}_Q^{(h)}$, $\mathrm{W}_K^{(h)}$, $\mathrm{W}_V^{(h)} \in \mathbb{R}^{d_h \times d_h}$ are learnable parameters. H is the number of attention heads. In addition, we adjust the inter level attention scores by a trainable parameter $\mathrm{W}_{adp} \in \mathbb{R}^{T_L \times T_{L-1}}$.

Trend-Aware Attention updates the lower level time series representation by the higher level time series, which helps to make the lower level time series learn longer time dependence. One drawback, however, is that time series at lower levels lose their inherent characteristics, which can make short term forecasting perform less well. To solve this problem, we compute Trend-Aware Attention and the self-attention of the current hierarchical time series synchronously in a parallel manner. The preference of self-attention for global information can impair the performance of short term forecasting, so we control the proportion of information flowing to the self-attention module at each time step by means of a selection gate:

$$\mathrm{V}_L = \mathrm{sigmoid}(\mathrm{MLP}(\mathrm{Concat}(\mathrm{X}_L,\mathrm{PE}_L))) \tag{5}$$

$$\mathrm{X}_L^S = \mathrm{V}_L \odot \mathrm{X}_L \tag{6}$$

We take the time series representation of layer L, X_L and the spatial-temporal position encoding PE_L (see Sect. 4.4 for details) of the concatenation as the input to the selection gate, and automatically learn the gate value of (0, 1) $\mathrm{V}_L \in \mathbb{R}^{T_L \times N \times D}$ by the sigmoid activation function. The symbol $\odot$ denotes the element-wise product, the attention module takes X_L^S as input and its operation can be expressed as:

$$\mathrm{S}_L^{(h)} = \mathrm{Softmax}(\frac{(\mathrm{X}_L^{S,(h)}\mathrm{U}_Q^{(h)})(\mathrm{X}_L^{S,(h)}\mathrm{U}_K^{(h)})^T}{\sqrt{d_h}})(\mathrm{X}_L^{S,(h)}\mathrm{U}_V^{(h)}) \tag{7}$$

$$\mathrm{S}_L = \mathrm{Concat}(\mathrm{S}_L^{(1)}, \mathrm{S}_L^{(2)}, \ldots, \mathrm{S}_L^{(H)}) \tag{8}$$

which $\mathrm{U}_Q^{(h)}$, $\mathrm{U}_K^{(h)}$, $\mathrm{U}_V^{(h)} \in \mathbb{R}^{d_h \times d_h}$ denotes learnable parameters. Finally, we model jointly the long-short-term temporal dependence by using the output of the Self-Attention module as a complement to Trend-Aware Attention:

$$\mathrm{B}_L = \mathrm{MLP}(\mathrm{Concat}(\mathrm{Q}_L, \mathrm{S}_L)) \tag{9}$$

where the MLP is a two-layer fully connected layer that weights and aggregates the feature representation of all attention heads. $\mathrm{B}_L \in \mathbb{R}^{T_L \times N \times D}$ is the final output representation of Trend-Aware Attention in the corresponding ST-Block. In the process of forward calculation, in order to avoid high computational cost, we can set the vector dimension of each of the two parts to $D/2$, and finally recover to D by performing concat operation on the channel by Eq. (9).

4.3 Diffusion Graph Convolution Network

In multivariate time series forecasting, the relationships between node pairs are not negligible, for example, traffic conditions on roads upstream of the traffic network produce impacts that are transmitted to downstream roads in the future, and weather conditions in adjacent regions are usually similar. Therefore, it is

necessary to consider these hidden spatial relationships. Existing studies usually construct the hidden relationships between node pairs through graphs, for instance, DCRNN computes the road network distance between pairs of nodes in the adjacency matrix using a threshold Gaussian kernel function. DSTAGNN calculates the similarity between different time series as the weights among node pairs by Wasserstein Distance. However, these approaches construct static or bi-directional graph-based structures, and we propose to learn a directed graph to preserve the property of state transfer between nodes (that is, a change in the state of one node leads to a change in the state of other nodes). It should be noted that the spatial structure in the traffic network includes both static and dynamic attributes, and for static attributes, it mainly refers to the inherent apriori knowledge of different correlations due to different road distances.

For dynamic attributes, let's take an example to help understand: due to the different attributes of different areas (apartment, school or industrial park), at 7 a.m., the correlation (A,B) between apartment A and school B is much greater than (B,A) due to students going to school, and at 6 p.m., (B,A) is much greater than (A,B) due to students leaving school. Therefore, in real traffic networks, there are hidden and uncertain relationships between different roads. If feature information is used to participate in the construction of the graph structure, the accuracy will be degraded during the testing process due to the different data and the accuracy deviation will be greater with time. Hence, we propose to learn the hidden graph structure in an adaptive manner and incorporate static attributes in an efficient way. It does not depend on the feature information at any moment and the graph structure is determined once the training of the model is completed.

First, we use thresholded Gaussian kernel function to measuring the proximity between different road pairs:

$$\mathrm{H}_{i,j} = \exp(-\frac{dist(v_i, v_j)^2}{\sigma^2}) \tag{10}$$

where $dist(v_i, v_j)$ represents the road network distance from node v_i to node v_j, σ is the standard deviation of distances, $\mathrm{H}_{i,j}$ denotes the edge weight between node v_i and node v_j.

Then, we obtain the embedding representation of each node by node2Vec:

$$\mathrm{N} = \mathrm{node2Vec}(\mathrm{H}) \tag{11}$$

$\mathrm{N} \in \mathbb{R}^{N \times D}$ is the embedding representation of the nodes in the spatial, taking the distance-based adjacency matrix H as input. The node2Vec algorithm makes nodes within the same region or nodes that have similar structural features represent similar. In particular, we randomly initialize two learnable node embedding matrices E_1,$\mathrm{E}_2 \in \mathbb{R}^{N \times D}$ and concate them with N on the channel:

$$\mathrm{M}_1 = \tanh(\alpha(\mathrm{linear}(\mathrm{Concat}(\mathrm{E}_1, \mathrm{N})))) \tag{12}$$

$$\mathrm{M}_2 = \tanh(\alpha(\mathrm{linear}(\mathrm{Concat}(\mathrm{E}_2, \mathrm{N})))) \tag{13}$$

M_1 and M_2 are the new node embedding representation containing learnable and static spatial information. Then, we regularize the adjacency matrix by subtraction terms and the ReLU activation function:

$$\mathrm{A} = \mathrm{ReLU}(\tanh(\alpha(\mathrm{M}_1\mathrm{M}_2^{\mathrm{T}} - \mathrm{M}_2\mathrm{M}_1^{\mathrm{T}}))) \tag{14}$$

which Θ_1,$\Theta_2 \in \mathbb{R}^{D\times D}$ are learnable parameters, α is a hyper-parameter for controlling the saturation rate of the activation function, Eq. (14) implements the asymmetric nature of the adjacency matrix.

We characterize the state transfer between nodes as a spatial diffusion process of nodes, and this Markovian stochastic process converges to a smooth distribution after K time steps by performing a random wander on the graph. Given the graph signal $\mathrm{X}_L \in \mathbb{R}^{T_L\times N\times D}$ and adjacency matrix $\mathrm{A} \in \mathbb{R}^{N\times N}$ at the Lth level, we describe the diffusion graph convolution as:

$$\mathrm{Z}_L = \sum_{k=0}^{K} (\mathrm{D}_O^{-1}\mathrm{A})^k \mathrm{X}_L \mathrm{W}_{Ok} + (\mathrm{D}_I^{-1}\mathrm{A}^{\mathrm{T}})^k \mathrm{X}_L \mathrm{W}_{Ik} \tag{15}$$

In the case of directed graphs, the diffusion process has two directions, outflow and inflow, and the corresponding state transfer matrix for both are $\mathrm{D}_O^{-1}\mathrm{A}$ and $\mathrm{D}_I^{-1}\mathrm{A}^{\mathrm{T}}$, respectively. Where D_O and D_I are the degree matrix of the corresponding matrix, $\mathrm{W}_{Ok}, \mathrm{W}_{Ik} \in \mathbb{R}^{D\times D}$ are the learnable parameter, and $\mathrm{Z}_L \in \mathbb{R}^{T_L\times N\times D}$ is the output of the diffusion graph convolution layer in the ST-Block corresponding to the Lth level.

Then, we aggregate the outputs of the Trend-Aware Attention and diffusion graph convolution layer, either by summing or concatting over the channels. We select $\mathrm{SUM}(\cdot)$ as the aggregator function which is differentiable and maintains high representational capacity:

$$\mathrm{Y}_L = \mathrm{Agg}(\mathrm{Q}_L, \mathrm{Z}_L) = \mathrm{Q}_L + \mathrm{Z}_L \tag{16}$$

Finally, we add residual connectivity and BatchNorm to Y_L and obtain the output of ST-Block by an MLP containing two layers of fully connected neural networks:

$$\mathrm{Y}_L^{out} = \mathrm{MLP}(\mathrm{BatchNorm}(\mathrm{Agg}(\mathrm{Y}_L, \mathrm{X}_L))) \tag{17}$$

$\mathrm{Y}_L^{out} \in \mathbb{R}^{T_L\times N\times D}$ is the output of the ST-Block corresponding to the Lth level.

4.4 Position Encoding

Considering that the pyramid performs Trend-Aware Attention between adjacent levels, and that the sequential relationships of adjacent levels lose their relevance to each other. To solve this problem, we add location codes for the different levels, which are aggregations of temporal and spatial codes (the aggregation function uses $\mathrm{SUM}(\cdot)$). Temporal encoding is one-hot encoding and concat separately for day-of-week and time-of-day of each time step. In spatial, we randomly initialize a vector representation for each node, both of which have the same number of

channels after passing through the fully connected neural network. For example, for node v_i on time step t_j, its position encoding is defined as:

$$PE^{v_i,t_j} = \text{Agg}(\text{MLP}(\text{onehot}(t_j)), \text{MLP}(\text{emb}(v_i))) \tag{18}$$

For the Lth level in the pyramid, the position encoding is defined as:

$$PE_L^{v_i,L_j} = \text{Agg}(\text{MLP}(\sum_{u=j\times p_L}^{u=(j+1)\times p_L-1} \text{onehot}(t_u)), \text{MLP}(\text{emb}(v_i))) \tag{19}$$

In the Lth ($L > 1$) level, each trend block (aggregated by multiple consecutive time steps) represents a time horizon, as exemplified by Eq. (12), L_j is the jth trend block in the Lth hierarchy, p_L is the length of time of each trend block in the Lth level. We sum the one-hot encoding corresponding to successive time steps and set the maximum value to 1.

The position encoding preserves the correlation between level, effectively modeling long term dependence while better preserving similar information when performing Trend-Aware Attention.

5 Experiment

5.1 Dataset

To evaluate the model performance, we conducted extensive experiments on four traffic flow datasets [18], namely PEMS03, PEMS04, PEMS07 and PEMS08 datasets, which were collected on California freeways.

5.2 Baseline Methods

(1) VAR [15] which is a traditional time series model that captures the pairwise relationship of time series; (2) ARIMA [8] which is a classical time series model; (3) STGCN that models spatial and temporal correlations using GCN and CNN, respectively; (4) DCRNN [14] that captures spatial-temporal correlation using GRU and diffusion graph convolution network, respectively; (5) Graph WaveNet [23] that combines adaptive graph convolution and dilated casual convolution to capture spatial-temporal correlations; (6) ASTGCN [5] that is based on spatial-temporal attention and model spatial and temporal correlations by GCN and CNN; (7) STSGCN [19] that constructs a local spatio-temporal graph and captures local spatio-temporal correlations by spatio-temporal synchronous graph convolution; (8) AGCRN [20] that uses adaptive graphs to describe spatial correlation and GRU to model temporal correlation; (9) Z-GCNETS [11] that models spatial and temporal correlation using graph convolution and GRU; (10) GMAN [29] that captures spatio-temporal correlations by attention and designs a transformation layer to reduce error propagation; (11) DSTAGNN [9] that was designed to describe regions with similar functions.

5.3 Experimental Settings

The dataset is divided into training, validation and test sets in the ratio of 6:2:2, and they are normalized with Z-Score. Following the standard benchmark setting for the domain, we use data from 12 consecutive historical time steps to forecast traffic data from 12 consecutive future time steps, with an interval of 5 min between two consecutive time steps. We use Adam optimizer as the models' optimizer with initial learning rate set to 0.01, BathSize to 128, attention head to 8, vector dimension to 64, and Pyramid Constructor with convolution kernel and three convolution layers with step size [2, 2, 3]. We use mean absolute error (MAE), root mean squared error (RMSE), and mean absolute percentage error (MAPE) as the evaluation metric and MAE as the loss function.

Table 1. .

Model	PEMS03			PEMS04			PEMS07			PEMS08		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
VAR	23.65	38.26	24.51%	24.54	38.61	17.24%	50.22	75.63	32.22%	19.19	29.81	13.10%
ARIMA	35.41	47.59	33.78%	33.73	48.80	24.18%	38.17	59.27	19.46%	31.09	44.32	22.73%
FC-LSTM	21.33	35.11	23.33%	26.77	40.65	18.23%	29.98	45.94	13.20%	23.09	35.17	14.99%
STGCN	17.55	30.42	17.34%	21.16	34.89	13.83%	25.33	39.34	11.21%	17.50	27.09	11.29%
DCRNN	17.99	30.31	18.34%	21.22	33.44	14.17%	25.22	38.61	11.82%	16.82	26.36	10.92%
GraphWaveNet	19.12	32.77	18.89%	24.89	39.66	17.29%	26.39	41.50	11.97%	18.28	30.05	12.15%
ASTGCN	17.34	29.56	17.21%	22.93	35.22	16.56%	24.01	37.87	10.73%	18.25	28.06	11.64%
STSGCN	17.48	29.21	16.78%	21.19	33.65	13.90%	24.26	39.03	10.21%	17.13	26.80	10.96%
AGCRN	15.98	28.25	15.23%	19.83	32.26	12.97%	22.37	36.55	9.12%	15.95	25.22	10.09%
STFGNN	16.77	28.34	16.30%	20.48	32.51	16.77%	23.46	36.60	9.21%	16.94	26.25	10.60%
Z-GCNETS	16.64	28.15	16.39%	19.50	31.61	12.78%	21.77	35.17	9.25%	15.76	25.11	10.01%
GMAN	15.52	26.53	15.19%	19.25	30.85	13.00%	20.68	33.56	9.31%	14.87	24.06	9.77%
DSTAGNN	15.57	27.21	**14.68%**	19.30	31.46	12.70%	21.42	34.51	9.01%	15.67	24.77	9.94%
PYNet	**14.94**	**25.27**	14.94%	**18.46**	**30.36**	**12.46%**	**19.61**	**32.85**	**8.36%**	**14.03**	**23.84**	**9.39%**
improve	**3.73%**	**3.62%**	-	**4.10%**	**1.59%**	**1.89%**	**5.17%**	**2.11%**	**7.21%**	**5.65%**	**0.91%**	**3.89%**

5.4 Experiment Results

Table 1 shows the performance of PYNet and the thirteen baselines on the four datasets, and we report the average error of the one-hour ahead forecasting. As can be seen, PYNet achieves state-of-the-art performance on four datasets, and in terms of MAE, PYNet improves the state-of-the-art results by 2.51%, 4.16%, 5.08% and 5.51%, respectively. In addition, we observed that: (1) The performance of VAR and ARIMA is poor; they rely on the assumption of linearity in the data, while traffic data has dynamic non-linear feature. (2) GNN-based deep learning methods (STGCN, DCRNN, AGCRN, GraphWaveNet, DSTAGNN) take spatial correlation into account and usually have better forecasting performance. However, the semantic information contained in the graph structure may be imperfect or even biased, which limits the expressive power of the graph model. (3) The models based on the attention mechanism,

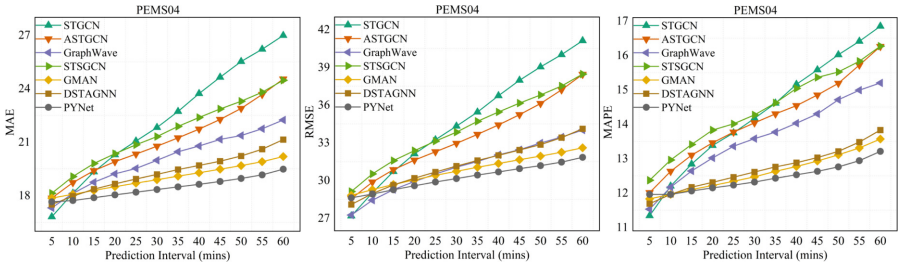

Fig. 2. Forecasting performance comparison at each horizon on the PEMS04 dataset.

ASTGCN and GMAN, perform better in long-term forecasting, but the insensitivity of attention to local information leads to poorer short-term forecasting performance.

Compared with the above methods, PYNet introduces a pyramid structure to learn the multi-scale representation of time series, which can effectively model long and short term dependence. In moreover, we add corresponding scale position encoding for each level in the pyramid to record the relative position relationship and retain the correlation between levels. We perform Trend-Aware Attention and diffusion GCN top-down in a pyramid network, where the former gradually causes the lower-level time series to learn several times their own long term dependence until the update of the initial time series is completed. The latter performs diffusion convolution operations on directed graphs to preserve the properties of state transfer between nodes. Considering these features, PYNet consistently outperforms other methods.

To investigate the specific performance of PYNet on short-medium term and medium-long term forecasting, we plot the error curves of the seven models on one-hour ahead forecasting in Fig. 2. We observe that STGCN and Graph WaveNet have the best short term (0min-10min) forecasting performance, and PYNet performs best when there is a medium-long term (>10 min) forecasting demand, and the error curve of PYNet grows the slowest with increasing time step, while the gap with other models gradually increases, which indicates that PYNet has strong stability while maintaining high performance.

5.5 Ablation Study

To verify the effectiveness of the individual components in PYNet, we made the following variants of PYNet: (1) PYNet-NC: We use the average pooling layer to construct the pyramid network. (2) PYNet-NT: We removed the Trend-Aware Attention from ST-Block. (3) PYNet-NS: We removed the self attention. (4) PYNet-ND: We remove the diffusion GCN from the ST-Block.

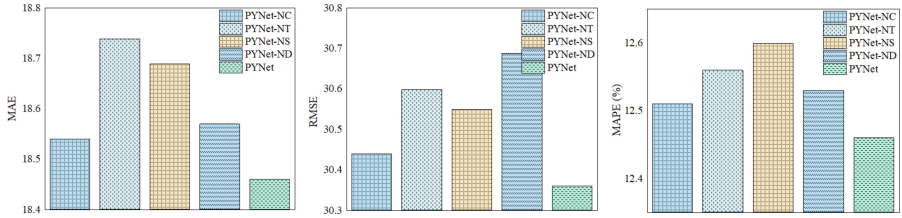

Fig. 3. Ablation study on PEMS04.

Figure 3 shows the average performance of PYNet and the four variants on the PEMS04 dataset. We observe that (1) The pyramid network constructed by CNN has better results compared to the average pooling layer because the convolution layer can weigh the importance of each time step in the window better than the pooling layer. (2) The performance of PYNet-NT decreases dramatically after removing the Trend-Aware Attention. This is because the Trend-Aware Attenion acts as a connection between two levels, and after removing it, the model cannot learn the correlation between the pyramid levels. (3) The self attention module complements the trend attention module with the aim of improving short term forecasting performance. When self attention is removed, PYNet-NS has the worst short term forecasting performance, which implies that self attention, as a complement to Trend-Aware Attenion, can effectively improve the performance of forecasting. (4) After removing the diffusion GCN, the model cannot capture the spatial correlation in the traffic network, and therefore the performance of PYNet-NS decreases.

5.6 Long Term Forecasting Performance

Long term (i.e., one hour or more) forecasting of traffic flow or traffic speed in a traffic network is challenging. The number of sensors deployed in a traffic network as nodes on a graph is usually huge, and if the model includes similarity calculation of nodes, the time complexity grows quadratically with the number of nodes, and secondly, the long term traffic conditions are difficult to forecast accurately due to the non-stationarity factor in the time series.

PYNet is based on a pyramid structure, which can effectively model correlations between time series with different time scales and has advantages in modeling long term temporal dependence. Therefore, to evaluate the performance of PYNet in long term forecasting, we forecast the future traffic data for 30, 60, 90 and 120 min on the PEMS04, and the results are shown in Table 2. We observe that PYNet improves the state-of-the-art baseline from 3.10% to 7.13% in MAE on the PEMS04 dataset, and as the time step, the gap further increases, which further demonstrates the performance of PYNet on long term traffic flow forecasting.

Table 2. Long term forecasting performance of different models on PEMS04.

Model	Metrics	30 min	60 min	90 min	120 min	Average
ASTGCN	MAE	22.08 ± 0.28	25.51 ± 0.69	29.32 ± 1.17	34.04 ± 1.42	26.01 ± 0.75
	RMSE	34.47 ± 0.42	39.35 ± 1.10	44.95 ± 1.87	51.60 ± 2.20	40.64 ± 1.28
	MAPE (%)	14.70 ± 0.10	16.84 ± 0.19	19.28 ± 0.28	22.49 ± 0.31	17.22 ± 0.19
STSGCN	MAE	21.66 ± 0.36	24.04 ± 0.41	26.70 ± 0.52	29.07 ± 0.64	24.35 ± 0.47
	RMSE	34.56 ± 0.75	37.98 ± 0.72	41.91 ± 0.75	45.45 ± 0.90	38.46 ± 0.79
	MAPE (%)	14.44 ± 0.13	15.76 ± 0.11	17.50 ± 0.18	18.92 ± 0.15	16.13 ± 0.20
GMAN	MAE	20.50 ± 0.01	21.02 ± 0.04	21.55 ± 0.08	22.29 ± 0.05	21.08 ± 0.05
	RMSE	33.21 ± 0.42	34.18 ± 0.48	35.09 ± 0.56	36.13 ± 0.54	34.24 ± 0.49
	MAPE (%)	15.06 ± 0.52	15.37 ± 0.57	15.78 ± 0.66	16.54 ± 0.76	15.48 ± 0.60
DSTAGNN	MAE	19.36 ± 0.04	20.69 ± 0.08	21.69 ± 0.03	22.91 ± 0.15	20.60 ± 0.02
	RMSE	31.36 ± 0.17	33.65 ± 0.27	35.29 ± 0.22	36.81 ± 0.04	33.47 ± 0.15
	MAPE (%)	12.88 ± 0.02	13.54 ± 0.03	14.22 ± 0.01	15.04 ± 0.05	13.58 ± 0.02
PYNet	MAE	$\mathbf{18.76 \pm 0.02}$	$\mathbf{19.35 \pm 0.03}$	$\mathbf{19.88 \pm 0.03}$	$\mathbf{20.70 \pm 0.04}$	$\mathbf{19.38 \pm 0.02}$
	RMSE	$\mathbf{30.72 \pm 0.08}$	$\mathbf{31.86 \pm 0.07}$	$\mathbf{32.79 \pm 0.08}$	$\mathbf{33.91 \pm 0.07}$	$\mathbf{31.84 \pm 0.07}$
	MAPE (%)	$\mathbf{12.46 \pm 0.24}$	$\mathbf{12.79 \pm 0.29}$	$\mathbf{13.15 \pm 0.26}$	$\mathbf{13.82 \pm 0.31}$	$\mathbf{12.86 \pm 0.25}$
Improve	MAE	**3.10%**	**6.48%**	**7.75%**	**7.13%**	**5.92%**
	RMSE	**2.04%**	**5.32%**	**6.55%**	**6.14%**	**4.87%**
	MAPE	**3.26%**	**5.54%**	**7.52%**	**8.11%**	**5.30%**

6 Conclusion

In this paper, we propose a pyramid network for traffic forecasting tasks, namely PYNet, where the Pyramid Constructor initialize a pyramid network with a multi-level structure through a set of convolution layers. Then we perform Trend-Aware Attention in the pyramid network top-down between adjacent levels to compute the attention matrix in local context, which not only reduces the impact of anomalies in the data, but also allows the trend blocks in the lower levels of the time series to benefit from their own multiplicity of perceptual fields. In spatial dimension, we learn an adaptive unidirection graph that maintains the properties of state transfer between nodes by a random walk process over spatially. Finally the effectiveness of PYNet was verified by experiments on four traffic flow datasets.

Acknowledgment. This research was supported by the National Natural Science Foundation of China (Grant no. 62266040), the Key Research and Development Project of Gansu Province (Grant no. 20YF8GA049), the Youth Science and Technology Fund Project of Gansu Province (Grant no. 20JR10RA097), the Industrial Support Plan Project for Colleges and Universities in Gansu Province (Grant no. 2022CYZC-13), and the Lanzhou Municipal Science and Technology Project (Grant no. 2019-1-34).

Ethical Considerations. The work studied in this paper can contribute not only towards more accurate predictions of traffic forecasts, but also to more efficient traffic scheduling. Our work also provides better planning methods for pedestrians, helping

people to save time and, on an environmental level, to save energy. This research is primarily based on a Pyramid learning structure to more effectively and deeply mine the underlying features of traffic data. This prediction model also addresses to some extent the prediction problems of other spatio-temporal series, such as water quality prediction and weather prediction. There are no ethical implications for this work such as possible use in policing or military related applications.

References

1. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203 (2013)
2. Chen, W., Chen, L., Xie, Y., Cao, W., Gao, Y., Feng, X.: Multi-range attentive bicomponent graph convolutional network for traffic forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 3529–3536 (2020)
3. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
4. Connor, J.T., Martin, R.D., Atlas, L.E.: Recurrent neural networks and robust time series prediction. IEEE Trans. Neural Networks **5**(2), 240–254 (1994)
5. Guo, S., Lin, Y., Wan, H., Li, X., Cong, G.: Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting. IEEE Trans. Knowl. Data Eng. **34**(11), 5415–5428 (2021)
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
7. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
8. Kumar, S.V., Vanajakshi, L.: Short-term traffic flow prediction using seasonal ARIMA model with limited input data. Eur. Transp. Res. Rev. **7**(3), 1–9 (2015)
9. Lan, S., Ma, Y., Huang, W., Wang, W., Yang, H., Li, P.: DSTAGNN: dynamic spatial-temporal aware graph neural network for traffic flow forecasting. In: International Conference on Machine Learning, pp. 11906–11917. PMLR (2022)
10. Li, F., et al.: Dynamic graph convolutional recurrent network for traffic prediction: benchmark and solution. ACM Trans. Knowl. Discov. Data **17**(1), 1–21 (2023)
11. Li, M., Zhu, Z.: Spatial-temporal fusion graph neural networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 4189–4196 (2021)
12. Li, P., Fang, J., Chao, P., Zhao, P., Liu, A., Zhao, L.: JS-STDGN: a spatial-temporal dynamic graph network using JS-Graph for traffic prediction. In: Bhattacharya, A., et al. (eds.) Database Systems for Advanced Applications: 27th International Conference, DASFAA 2022, Virtual Event, 11–14 April 2022, Proceedings, Part I, pp. 191–206. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-00123-9_15
13. Li, Q., Han, Z., Wu, X.M.: Deeper insights into graph convolutional networks for semi-supervised learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
14. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: data-driven traffic forecasting. arXiv preprint arXiv:1707.01926 (2017)
15. Lu, Z., Zhou, C., Wu, J., Jiang, H., Cui, S.: Integrating granger causality and vector auto-regression for traffic prediction of large-scale WLANs. KSII Trans. Internet Inf. Syst. **10**(1), 136–151 (2016)

16. Oreshkin, B.N., Amini, A., Coyle, L., Coates, M.: FC-GAGA: fully connected gated graph architecture for spatio-temporal traffic forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 9233–9241 (2021)
17. Park, C., et al.: ST-GRAT: a novel spatio-temporal graph attention networks for accurately forecasting dynamically changing road speed. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 1215–1224 (2020)
18. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014)
19. 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)
20. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, vol. 27 (2014)
21. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
22. Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., Zhang, C.: 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, pp. 753–763 (2020)
23. Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C.: Graph WaveNet for deep spatial-temporal graph modeling. arXiv preprint arXiv:1906.00121 (2019)
24. Yao, H., Tang, X., Wei, H., Zheng, G., Li, Z.: Revisiting spatial-temporal similarity: a deep learning framework for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 5668–5675 (2019)
25. Yao, H., et al.: Deep multi-view spatial-temporal network for taxi demand prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
26. Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. arXiv preprint arXiv:1709.04875 (2017)
27. Zhang, J., Zheng, Y., Qi, D.: Deep spatio-temporal residual networks for citywide crowd flows prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31 (2017)
28. Zheng, C., et al.: Spatio-temporal joint graph convolutional networks for traffic forecasting. IEEE Trans. Knowl. Data Eng. **17**, 691–703 (2023)
29. Zheng, C., Fan, X., Wang, C., Qi, J.: GMAN: a graph multi-attention network for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 1234–1241 (2020)
30. Zheng, C., Fan, X., Wen, C., Chen, L., Wang, C., Li, J.: DeepSTD: mining spatio-temporal disturbances of multiple context factors for citywide traffic flow prediction. IEEE Trans. Intell. Transp. Syst. **21**(9), 3744–3755 (2019)
31. Zhu, D., Cui, P., Wang, D., Zhu, W.: Deep variational network embedding in Wasserstein space. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2827–2836 (2018)