Node Connection Strength Matrix-Based Graph Convolution Network for Traffic Flow Prediction

Jian Chen, Wei Wang, *Member, IEEE*, Keping Yu, Xiping Hu, Ming Cai, and Mohsen Guizani, *Fellow, IEEE*

***Abstract*—Traffic flow prediction plays an integral role in intelligent transport systems, helping to manage and control urban traffic and improving the operational efficiency of road networks. Although the current mainstream traffic flow prediction models have achieved good accuracy, they cannot effectively utilize the unique characteristics of the traffic network where the importance of a node in the traffic network is positively correlated with the traffic flow through the node. Actually, the historical traffic properties of nodes will have a great influence on the future. With this background, in this paper, we propose a node connection strength index by network representation learning to utilize the historical traffic attributes of nodes. Then, we design a graph convolution network based on the node connection strength matrix to predict the traffic flow of the node. A novel Dynamics Extractor is designed to learn the various characteristics of the traffic flow. Experimental results demonstrate that the proposed scheme has a better performance by comparison with baseline methods.**

***Index Terms*—Graph convolution network, network representation learning, node connection strength, traffic flow prediction.**

Manuscript received 19 October 2022; revised 20 January 2023; accepted 20 March 2023. Date of publication 6 April 2023; date of current version 19 September 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 52102400, in part by the National Key Research and Development Program of China under Grant 2020YFB1600400, in part by the Fundamental Research Funds for the Central Universities, Sun Yat-sen University under Grant 22dfx08, in part by the Sustainable Development Science and Technology Project of Shenzhen Science and Technology Innovation Commission under Grant KCXFZ20201221173411032, and in part by the Industry-University-Research Cooperation Project under Grant GP/026/2020 of Zhuhai City, Guangdong Province, China. The review of this article was coordinated by Prof. Jalel Ben-Othman. *(Corresponding authors: Wei Wang; Keping Yu; Xiping Hu.)*

Jian Chen is with the School of Intelligent Systems Engineering, Sun Yat-Sen University, Guangzhou 510006, China, and also with the Department of Engineering, Shenzhen MSU-BIT University, Shenzhen 518172, China (e-mail: chenj589@mail2.sysu.edu.cn).

Ming Cai is with the School of Intelligent Systems Engineering, Sun Yat-Sen University, Guangzhou 510006, China (e-mail: caiming@mail.sysu.edu.cn).

Wei Wang and Xiping Hu are with the School of Medical Technology, Beijing Institute of Technology, Beijing 100081, China, and also with the Department of Engineering, Shenzhen MSU-BIT University, Shenzhen 518172, China (e-mail: ehomewang@ieee.org; huxp@bit.edu.cn).

Keping Yu is with the Graduate School of Science and Engineering, Hosei University, Koganei 184-8584, Japan, and also with the RIKEN Center for Advanced Intelligence Project, RIKEN, Chuo 103-0027, Japan (e-mail: keping.yu@ieee.org).

Mohsen Guizani is with the Department of Machine Learning, Mohamed Bin Zayed University of Artificial Intelligence, Masdar 999041, UAE (e-mail: mguizani@ieee.org).

Digital Object Identifier 10.1109/TVT.2023.3265300

I. Introduction

With the advent of the era of intelligence, modern cities are gradually transforming into smart cities. The acceleration of urbanization and the rapid growth of the floating population have brought huge loads and pressures to the urban road traffic network. The smart city provides ideas for solving the problem brought about by urbanization [1]. Therefore, this also provides opportunities for the development of intelligent transportation systems (ITS).

Intelligent transportation systems can improve the operational efficiency of urban transport, ease traffic congestion, and improve traffic design, management, and safety. There are some specific research directions in the intelligent transportation system where we can analyze the corresponding traffic problem in these directions. The correlations between directions and problems are shown in Fig. 1.

As Fig. 1 illustrates, traffic flow prediction is the core component of an intelligent transportation system. Accurate traffic flow prediction can assist route planning and guide vehicle scheduling, which can reduce congestion, improve traffic safety, and even provide support for autonomous driving [2], [3], [4]. Therefore, completing the traffic flow prediction based on the traffic network is now a major research hotspot in this field.

A traffic network is a classical complex network structure, which has obvious characteristics. In addition to the spatial topological properties, a traffic network also has unique traffic properties. The traffic flow of nodes on the network has spatiotemporal correlations hidden in the traffic features such as flow, speed, and so on. Although the traffic flow prediction can be modeled as a typical time series forecasting problem, using time series models alone is not sufficient since the spatial dependency will be ignored. Network representation learning can help us capture the spatiotemporal dependency of traffic networks. To the best of our knowledge, many studies have used network representation learning to complete the traffic flow prediction task. Existing research has shown that the graph convolution method can effectively exploit the network properties of traffic networks. The traffic flow of traffic nodes is always associated with other nodes in the network [5]. But most of them cannot make full use of the historical traffic characteristics of traffic nodes, so the potential information of traffic data is not captured. This is one of the limitations of existing methods.

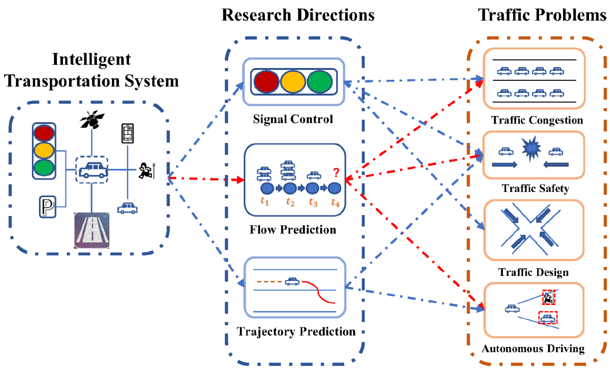


Fig. 1. Correlations between research directions and traffic problems. The number of cars represents the flow of the node in Flow Prediction.

Against this background, in this paper, we propose an efficient traffic flow prediction method by combining node embedding and graph convolution to make better use of the traffic characteristics of nodes. First of all, we learn the embedding vectors of traffic nodes and evaluate the importance of each node. Then, we propose a node connection strength index between nodes of the traffic network which can show the relationship between the traffic flow of nodes. After that, we design a deep-learning model based on the node connection strength matrix by combining a graph convolution and a time series model to complete the prediction task. To capture the spatiotemporal dependency, we also leverage the spatial-temporal network in the proposed model and a novel Dynamics Extractor is designed to learn the various characteristics of the traffic flow. We finally set up an experiment on a real-world dataset to verify the feasibility of the method, we proposed. Experimental results demonstrate that our proposed method can bring an obvious improvement in the traffic flow prediction task by comparison with several baseline methods.

The main contributions of this paper are as follows:

- We propose a method to evaluate the importance of nodes and design the node connection strength index according to the traffic flow relationship based on this.
- We design a deep learning model to predict the traffic flow of nodes, where a node connection matrix is used to replace the adjacency matrix as an improvement to utilize the information of the traffic data. We propose a dynamic extractor to extract the changing features of the traffic flow.
- We perform experiments on the real-world dataset to evaluate the performance of our proposed method. We also compare the evaluation metrics with baseline methods to prove the superiority of our method.

The remainder of this manuscript is organized as follows. Section II reviews the related work in this field. The method we are proposing is illustrated in detail in Section III. Experimental setup and results are in Sections IV and V. Finally, we make a conclusion of this work in Section VI.

II. Related Work

In this section, we review the related work from the perspectives of traffic flow prediction with deep learning and network representation learning.

A. Traffic Flow Prediction With Deep Learning

Traffic flow prediction is always a hot topic in ITS, and plenty of effective attempts have been devoted to completing this task. The methods proposed to predict short-term traffic flow can be divided into three kinds which will be reviewed in the following part: Statistical Learning Methods, Machine Learning Methods, and Deep Learning Methods. Furthermore, Deep Learning Methods are one of the most representative results, which are now the mainstream methods to predict short-term traffic flow.

Statistical Learning Methods are always data-driven methods, such as the Autoregressive Integrated Moving Average model (ARIMA) and its variants [6], [7]. However, these methods only consider temporal information but not the spatial dependency of the traffic flow. Machine Learning Methods can perform better than Statistical Learning Methods in capturing the spatial-temporal dependence of data [8], [9], [10], [11]. Lin et al. [12] transform the traffic flow sequence into a vector and then combine support vector regression and K-nearest neighbors to get a better-predicted performance. Still, these methods always have no adequate performance to exploit the complexity and dynamics of traffic data. To make good use of the characteristic of traffic data, Deep Learning Methods are proposed soon.

Unlike the other two methods, Deep Learning Methods have an excellent performance to capture the nonlinear spatiotemporal characteristic of traffic data [5]. Deep Learning Methods can be divided into two types: the spatial dependency model and the time dependency model. The spatial dependency model defines a convolution network directly on the graph called graph convolutional network (GCN) [13]. GCN will aggregate the information from the central node and its neighbors to get a new representation of the central node. Therefore, GCN models are used in the traffic network to extract the influence between traffic nodes for traffic flow prediction. Time dependency models treat traffic flow data as time series data and try to capture its temporal dependency [9]. Recently, more researchers have combined the spatial dependency model and time dependency model to get better performance.

Zheng et al. [14] use LSTM [15] to capture the temporal characteristic of the traffic data and predict the flow. To extract the spatial and temporal characteristics of the traffic flow data, Cui et al. [16] combine GCN and RNN [17] and Wang et al. [18] proposed a spatial-temporal graph neural network. Zhang et al. [19] propose a spatial-temporal graph diffusion network that refines the resolution-aware region representations learned from the graph attention module and finally, gets the great encoding information of the traffic network. Guo et al. [20] also uses the graph attention mechanism to learn traffic network features better. Wang et al. [21] uses a deep convolutional neural network to extract the spatial feature and then uses LSTM to extract the temporal feature. Peng et al. [22] uses dynamic GCN with reinforcement learning to complete the short-term traffic flow prediction. Although they have achieved relatively good traffic flow prediction results, these methods fail to take full advantage of the inherent traffic flow properties of the traffic network and will cause information loss.

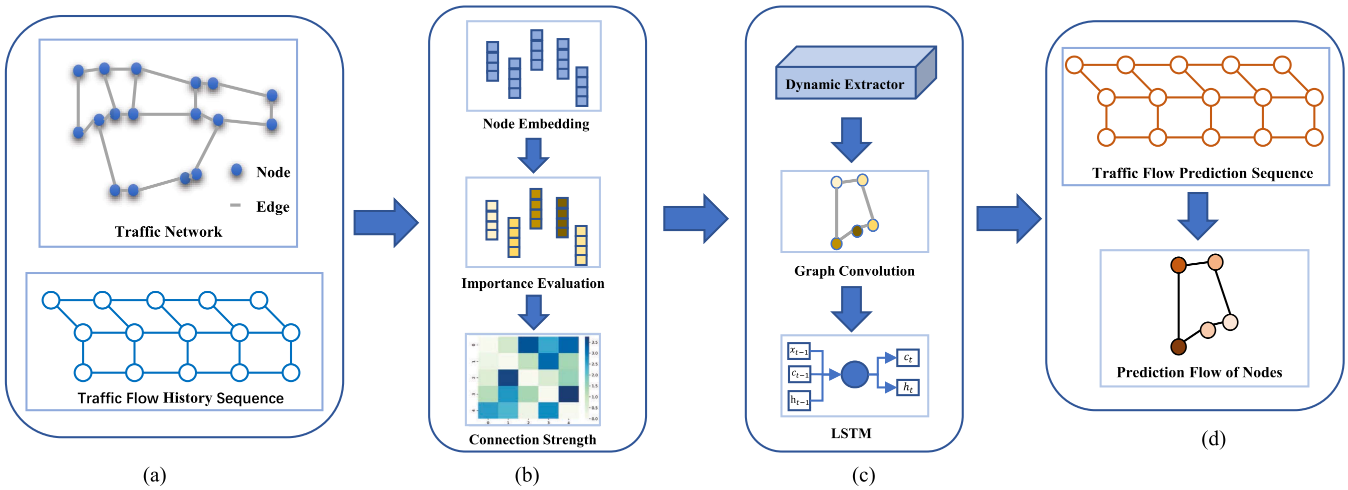


Fig. 2. Framework of NCSGCN.In the Node Embedding part, nodes are represented in dense vector form. In the Importance Evaluation part, different colors mean different importance. In the Connection Strength Matrix, the deeper the color is, the larger the strength is. (a) Data Input. (b) Learning Module. (c) Prediction Module. (d) Prediction Output.

B. Network Representation Learning

Network representation learning (NRL) is to learn effective features which can be used directly in follow-up specific tasks from the network data [23]. NRL can avoid complex feature engineering and algorithm derivation based on the original network data [24]. One of the main ideas of network representation learning is node embedding. Node embedding is to encode the information of nodes in the network and converts them into low-dimensional vectors called embedding vectors that contain information about the location of nodes in the network and local domain structure information [25].

The common way to get node embedding is to use an encoder-decoder framework [26]. In this framework, the process of learning can be summarized as follow: First, the encoder learns a low-dimensional embedding vector for each node in the network. Then the decoder reconstructs the neighborhood information of each node in the original network through the embedding vector. Inspired by natural language processing, random walk methods are proposed. Perozzi et al. [27] propose DeepWalk and Grover and Leskovec [28] propose node2vec, these two methods extract the node sequence by using random walks, encode the statistical information of the node sequence, and learn the vector representation of the vertices according to the co-occurrence relationship between the vertices and the vertices. Tang et al. [29] propose a method called LINE with an implicit way to learn the node embeddings.

The methods above are shallow embedding methods that will not take advantage of the properties of nodes and cannot effectively capture the similarity of the neighborhood structure. To overcome the disadvantages of shallow embedding, some researchers propose graph convolution. A simple graph convolution operation based on the graph adjacency matrix is equivalent to aggregating the information from the neighbor nodes, and when the self-connection matrix (namely the identity matrix I) is added, the information of the node itself is further considered [25]. By stacking simple graph convolution layers in multiple layers, more effective node embeddings can be learned by taking advantage of node attributes and neighborhood structures [30], [31]. NRL also shows its effect in learning the network in the traffic field [32], [33]. Therefore, GCN models have become the mainstream methods to predict traffic flow. Considering the superiority of node embedding and graph convolution in dealing with a complex network, we combine them in our proposed method to extract the spatial-temporal traffic characteristic for better prediction.

TABLE I
PARAMETERS OF NCSGCN

parameter	Definition
p	Hyperparameter about the probability to revisit a node
q	Hyperparameter about the search strategy
α	Hyperparameter about the bias towards network structure
β	Hyperparameter about the bias towards traffic attributes
$flow_v$	Traffic flow of node v
$u_{flow,k}$	Average flow of node cluster k
$impN$	Importance of traffic node
$impC$	Importance coefficient of traffic node cluster
NCS	Node connection strength index

III. PROPOSED METHOD

In this paper, we propose a novel graph convolutional network based on the node connection strength matrix called NCSGCN to predict the short-term flow of nodes in a traffic network. The framework of our proposed method can be seen in Fig. 2. With the data input, first of all, we will learn the traffic node embedding vectors. Then we will evaluate the node importance and construct the node connection strength matrix. Finally, we design a deep learning module by combining GCN with LSTM to predict the future traffic flow as an output.

The composition of the NCSGCN is discussed in detail below and the parameters used in the NCSGCN are listed in Table I.

A. Traffic Network Data

In order to construct the traffic network data, four kinds of data are needed [34]:

- **node** V: the set of intersections, sensors, and even the road segments.
- **edge** E: the link between nodes that can represent the real road between nodes.
- **weight** W: the weight set, which uses α_{ij} to represent the weight of edge from node i to node j. Common weights include distance, cost, and so on.
- **node features** X_t: a feature matrix that includes the traffic feature of nodes in the network such as flow, speed, and so on.

After getting the data above, a traffic network can be constructed as $G = (V, E, W)$ and a feature matrix X_t. These data can show the topology structure and traffic flow properties of the traffic network, which makes them the basis for traffic flow prediction. An adjacency matrix A representing the topology structure of the network will be built based on the network data $G = (V, E, W)$. The adjacency matrix A is the basis of the GCN. But most existing methods use a 0-1 matrix to show whether there is a link between two nodes or use the reciprocal distance between nodes as the elements of the matrix. This can not fully use the unique attributes of the traffic network. So we design the node connection strength matrix based on the adjacency matrix as an improvement, which will be illustrated in the following part.

B. Learning Module

To construct the node connection strength matrix, first, we will complete the node embedding to get the representation vectors of nodes. Then, the representation vectors and the traffic attributes of nodes are used to complete the node clustering. Based on the clustering results, the importance of nodes is evaluated and the connection strength between nodes is calculated. Therefore, the learning module of NCSGCN can be divided into three parts: node embedding, importance evaluation, and connection strength matrix construction.

1) Node Embedding: To make good use of the traffic attributes of nodes, we choose the TraNode2vec [33] algorithm while learning the node embedding vectors. TraNode2vec is inspired by node2vec which has performed well in complex network learning. TraNode2vec is a second-order random walk method that depends on the current node and its previous node. In this way, a higher-order dependency between nodes can be established.

Based on considering the characteristics of static network structures like node2vec, TraNode2vec adds unique dynamic traffic flow characteristics of the traffic road network. In the process of sampling the nodes, TraNode2vec will use the traffic flow to correct the sampling probability. The node with more flow will be more likely sampled.

TraNode2vec inherits the hyperparameters p and q from node2vec. p decides the probability to visit a node visited in the walk and q decides the search strategy. A large p will take the walk away from the initial node, otherwise will make it focus on the neighborhood of the source node. A large q will bias the search strategy towards BFS, otherwise DFS. In addition, TraNode2vec also design two hyperparameters α and β to balance the static network structure and dynamic traffic attributes. The larger the α is, the sample probability will depend on the network topology structure. The β has the same effect on traffic attributes.

2) Importance Evaluation: As there is positive relativity between the flow of the node and the importance of the traffic node [35], [36], we add the traffic attributes when clustering the node. We use the node embedding vectors and the features of traffic nodes to cluster nodes. The purpose of clustering is to classify the nodes with similar importance into a cluster.

As the initial traffic features are time series data, we replace them with their average as the features of nodes. Then, we eliminate the influence of different feature dimensions by the standard normalization below:

$$x'_j = \frac{x_j - \mu_j}{\sigma_j} \tag{1}$$

where x'_j is the feature j, μ_j is the average of feature j, and σ_j is the standard deviation of feature j.

To have a good performance on clustering, the Agglomerative Hierarchical Clustering method (AGNES) [37] is used. The core idea of AGNES is to divide the dataset at different levels to form a tree-like clustering structure. It uses a bottom-up aggregation strategy to merge the two closest classes into a new class. Therefore, AGNES can perform well on datasets of arbitrary shapes (e.g., traffic network data).

After clustering, we divide the nodes into different clusters. Then we design a node importance index based on the idea that the more flow a node has, the more important it is. We set this index as $impN$, which can be calculated by (2):

$$impN_v = \frac{flow_v}{\mu_{flow,k}} \times impC_k, \quad v \in V \tag{2}$$

where $impN_v$ is the importance of node v, $flow_v$ is the flow of node v, $\mu_{flow,k}$ is the average flow of cluster k and $impC_k$ is the importance of cluster k. The cluster importance $impC_k$ is designed by (3):

$$impC_k = \frac{\mu_{flow,k}}{\mu_{flow}}, \quad k \in K \tag{3}$$

where μ_{flow} is the average flow of all nodes in V.

In the method we propose above, the importance of each node depends not only on its own flow but also on its category. So it can take into account the topology structure of the traffic network and the traffic flow characteristics of the node itself in this way.

3) Connection Strength Matrix Construction: Since the traffic flow passing through the more important node pairs will be larger, it can be known that the traffic flow of nodes tends to flow to the nodes with higher importance among the neighboring nodes. Based on this idea, we design the node connection strength (NCS) indicator to show the traffic flow relationship between nodes.

Using the edge E of the network, the NCS can be calculated by (4):

$$NCS_{A,B} = \frac{flow_B}{\sum_{k1} flow_{k1}} \times imp_A + \frac{flow_A}{\sum_{k2} flow_{k2}} \times imp_B \tag{4}$$

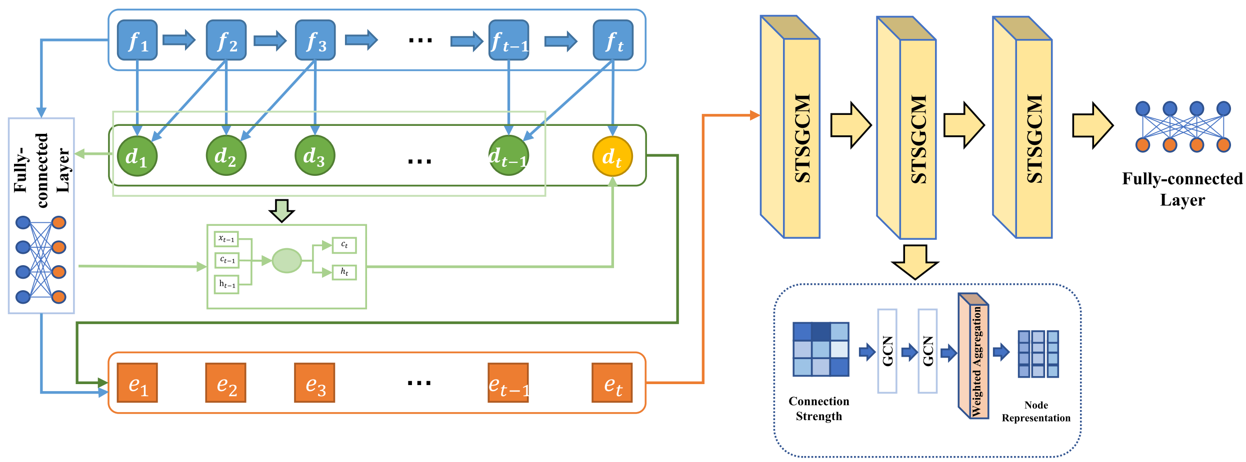


Fig. 3. Architecture of Predict Module, which combines a Dynamic Extractor, GCN module, and LSTM. f_t means the flow in the tth time step. d_t means the variation characteristics extracted in the t th time step. e_t means the encoding in the tth time step.

Algorithm 1: The connection strength matrix construction.

Require Traffic network $G = (V, E, W)$, node features X_t, clusters set K

Ensure Connection strength matrix S

$embedding = \textbf{TraNode2vec}(G, X)$;
$label_v = \textbf{AGNES}(embedding, X)$;
for k $\in K$ **do**
  $impC_k = \frac{\mu_{flow,k}}{\mu_{flow}}$;
end for
for v $\in V$ **do**
  $impN_v = \frac{flow_v}{\mu_{flow,k}} \times impC_k$;
end for
for e $\in E$ **do**
  $NCS_{v_1,v_2} = \frac{flow_{v_2}}{\sum_{k1} flow_{k1}} \times imp_{v_1} + \frac{flow_{v_1}}{\sum_{k2} flow_{k2}} \times imp_{v_2}$;
end for

where $NCS_{A,B}$ is the node connection strength between node A and node B, the ends of the edge, and $k1$, $k2$ respectively represent the neighbor nodes of node A and node B.

When the traffic network is constructed as an undirected graph, (4) can help us get the NCS of all nodes and the NCS we get has a symmetry which means that the nodes at both ends of the connection are equivalent. But if a directed graph is given, the NCS should be calculated correctly by (5):

$$NCS_{A,B} = \frac{flow_B}{\sum_{k1} flow_{k1}} * imp_A \quad (5)$$

which only considers the proportion of the traffic flow going to the node pointed to by the connection or edge to all the outgoing traffic flow from the source node.

Afterward, we design the connection strength matrix by using the NCS to replace the elements of the adjacency matrix. The connection strength matrix which is set as S contains more information about the transportation network and improves the effectiveness of the prediction module. The whole process of our learning module to construct the connection strength matrix S is shown in Algorithm 1.

Algorithm 1 requires the traffic network data $G = (V, E, W)$, node features data X_t, and clusters set K. The output is the Connection Strength Matrix S. Firstly, the node embedding can be obtained by the TraNode2vec. Then AGNES is used to get the cluster label of nodes. The importance of each cluster and each node is calculated. Finally the NCS_{v_1,v_2} can be obtained based on the importance and flow of each node. The Connection Strength Matrix S is constructed with the NCS.

C. Prediction Module

In the traffic flow prediction task, the traffic flow attributes of the traffic nodes and the neighbor nodes in the network have a spatiotemporal dependency. To capture this dependence in space and time, this paper designs a traffic flow prediction module based on the node connection strength matrix as Fig. 3 shows. Inspired by the Li et al. [38] which make use of the difference between each time step to improve the predicted performance, a Dynamics Extractor is designed to utilize the difference of traffic flow on each time horizon, and extract the variation characteristic of the traffic flow sequence.

With the input data, we use diff operation to get the variation of flow as (6):

$$d = [\Delta f_1, \Delta f_2, \ldots, \Delta f_{t-1}] \quad (6)$$

where Δf_{t-1} can be calculated as (7)

$$\Delta f_{t-1} = f_t - f_{t-1} \quad (7)$$

where f_t is the flow in tth time step. Since the flow sequence length is t, we can get $t-1$ variation. Then we use a Fully-connected layer to change the dimension and then use an LSTM to predict the tth variation. In order to capture the temporal dependency better, we first use the last hidden state of LSTM h_{t-1} as the output. Then we design a temporal attention mechanism [39] to improve it. The temporal attention mechanism can be calculated as (8).

$$\alpha_i = softmax(V^T \cdot tanh(W \cdot D_i)), i \in (1, t-1) \quad (8)$$

where $V \in R^{d\times 1}$ and $W \in R^{d\times 2\,d}$ are trainable parameters, and D_i is got by concatenate the h_{t-1} and d_i. After the attention weights generated, the output of Attention-LSTM $\hat{s}$ is computed as (9):

$$d_t = \alpha_1 d_1 + \cdots + \alpha_{t-1} d_{t-1} \tag{9}$$

Then we concatenate the output of f_i after going through the Fully-connected Layer and d_i to get the final encoding e_i, which is the input of graph convolution. In this way, the encoding e_i contains more temporal information than the original flow data f_i.

1) STSGCM: STSGCM is proposed by Song et al. [40]. To capture the spatial-temporal dependency of nodes, STSGCM constructs a local spatiotemporal graph. The local spatiotemporal graph will connect the nodes at adjacent time steps. In addition, a learnable temporal embedding representation matrix, as well as a spatial embedding representation matrix, are designed. After learning, these two matrices will contain important spatiotemporal information about node properties. So they are used to correct the input feature data X by broadcast operations for matrices. Each STSGCM includes multiple simple graph convolution layers to extract the feature of data effectively. In order to utilize the information of the local spatial-temporal graph better, we design the Weighted Aggregation as Fig. 4. We first use a max aggregation to get the maximum signal s_0. Then we use an attention mechanism to improve the performance. Similar to the attention-LSTM above, we concatenate the local graph signal with s_0 to get S_i. Then the calculation of the attention mechanism to get the aggregated signal is shown as (10) and (11):

$$\alpha_i = softmax(V^T \cdot tanh(W \cdot S_i)), i \in (t-1, t+1) \tag{10}$$

$$\hat{S}_0 = \alpha_{t-1} d_{t-1} + \alpha_t d_t + \alpha_{t+1} d_{t+1} \tag{11}$$

where W and V is trainable parameters.

2) Fully-Connected Layer: There are two fully-connected layers used in our model. Both of them are used to transform the data dimensions. The first one transforms the input data into the dimensions required by the convolution module. The other one transforms the output data of LSTM into the predicted flow result in the form of the time series we want as the final output of the model.

3) LSTM: LSTM is an improved temporal RNN that solves the problem of gradient disappearance and gradient explosion during long sequence training. LSTM has shown its superiority in sequence prediction tasks, and since traffic flow prediction is a typical sequence prediction task we combine it with STSGCM to complete the model.

4) Activation Function: Activation functions can bring nonlinearity into neural networks. In this paper, we choose $GELU$ [41] as the activation of the graph convolution and Fully-connected layer. $GELU$ is a variant of the nonlinear activation function $ReLU$. $GELU$ adds an idea of random regularization in the process of activation to solve the problem that $ReLU$ function lacks random factors. $GELU$ can retain some negative information and has shown better and more effective results in practice. In addition, we choose $tanh$ as the activation function of LSTM and attention mechanism.

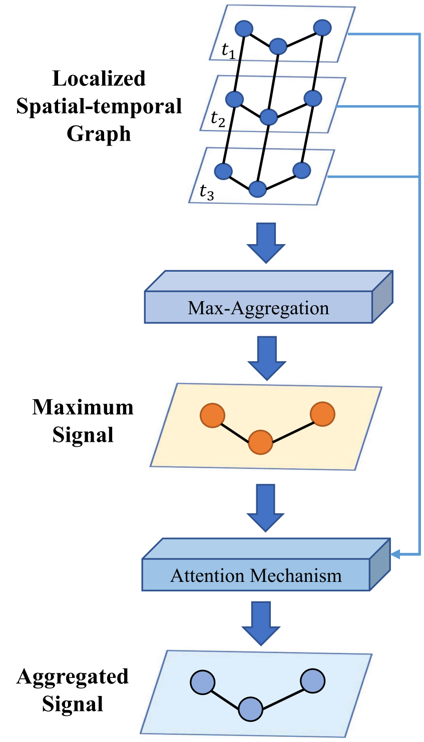


Fig. 4. Architecture of the Weighted Aggregation.

5) NCS Matrix: While the process of GCN, a common method is to use the 0-1 adjacency matrix A to complete the operation. In this paper, we use the NCS matrix S to replace A, to improve the representation power of the traffic network, and make better use of the traffic attributes of nodes. So the operation of a GCN layer in NCSGCN is defined by (12):

$$h^{(l)} = GCN(h^{(l-1)}) = \sigma(Sh^{(l-1)}W + b) \tag{12}$$

where W and b is the learnable weight matrix of the GCN layer, $\sigma()$ is the activation function and $h^{(l)}$ is the output of the l th GCN layer.

IV. Experimental Setup

To evaluate the performance of NCSGCN, we set up an experiment on a real-world highway dataset. And we will compare the results with some baseline methods by using some evaluation metrics.

A. Dataset

In this experiment, we use the open data collected from the Caltrans Performance Measurement System (PeMS) called PeMSD8. PeMSD8 contains 1979 sensor nodes on 8 roads and the features of nodes from July 1, 2016, to August 31, 2016. Nodes that are more than 3.5 miles away from any other node are considered redundant and removed. The features include each node's traffic flow, temporal occupancy, and speed. Our work is to predict the traffic flow of each node in the future. The dataset also provides a topological relationship and distance between nodes, which means a real road to connect the nodes.

Since the flow data has been aggregated to 5 minutes, each hour includes 12 data points. We split the dataset in 6:2:2 ratio

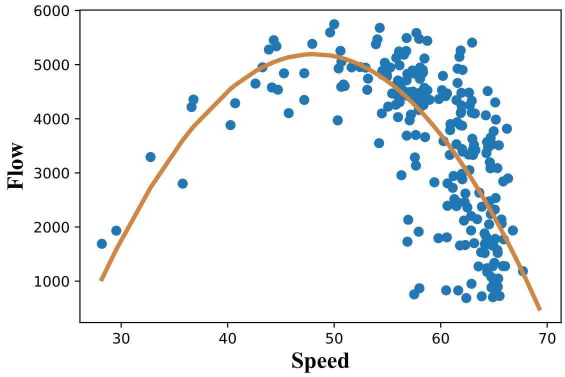


Fig. 5. Flow-Speed basic graph of example node.

to get the training set, valid set, and testing set. To ensure the test data is only used while testing, we only use the training set data to calculate the NCS matrix in the learning module.

We also analyze the dataset to dig out the traffic characteristics of the data. We choose one node of the network and calculate the traffic of the node per hour. Then we calculate the average speed for an hour and draw the Flow-Speed basic graph as Fig. 5 illustrates.

We can see that when the speed is smaller than the critical speed at which the flow is the maximum, the flow increases with speed. And when the speed is larger than the critical speed, the flow decreases as the speed increases. That's because only there is less flow on the road can the cars have a higher speed. This correlation between speed and flow is what we want to utilize in our model by the NCS index.

To analyze the temporal characteristics of the traffic flow better, we plot the flow of a node over time in Fig. 6.

Fig. 6(a) shows the flow of a node in a week and Fig. 6(b) shows the flow in a day. As illustrated, the flow has a significant periodicity within a week and distinct peaks in a day. These are the temporal characteristics we want to capture by the model we proposed.

B. Baseline Motheds

We choose six baseline methods to complete the comparison:

- **STSGCN** [40]: The Spatial-Temporal Synchronous Graph Convolutional Networks extracts the spatiotemporal correlation between traffic nodes at the same time but not respectively to dig the spatiotemporal information of data. So STSGCN has a good performance on traffic flow prediction tasks.
- **Graph WaveNet** [42]: We use GWN to represent this model in this paper. GWN combines graph convolution with dilated random convolution, proposes a new model framework to capture the spatiotemporal dependencies of network nodes, and has the ability to deal with long sequence data.
- **ASTGCN** [20]: Attention Based Spatial-temporal Graph Convolutional Network builds a spatiotemporal attention mechanism that learns the dynamic spatiotemporal correlation of traffic data. In this way, ASTGCN can effectively utilize the spatiotemporal correlation to achieve accurate flow prediction.

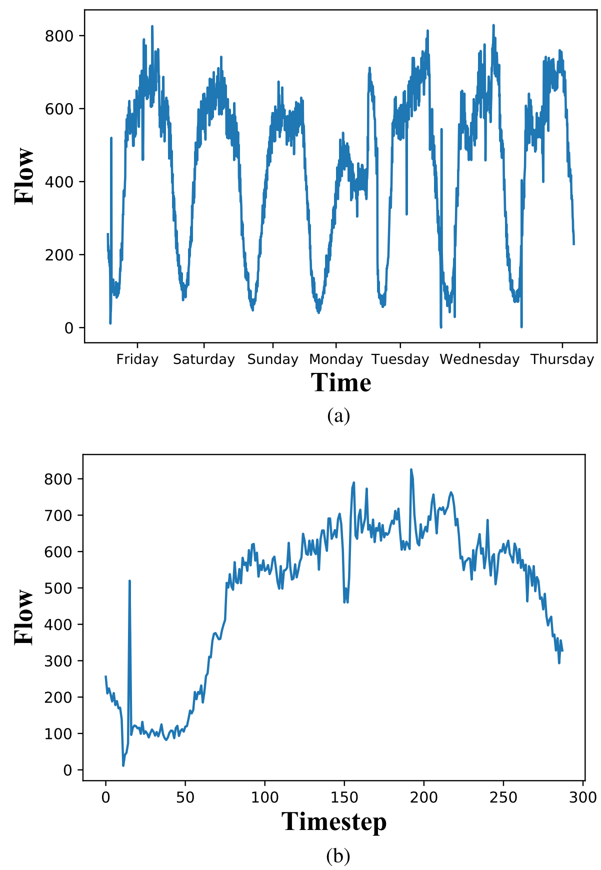


Fig. 6. Flow-Time Curve of example node. (a) A Week. (b) A Day.

- **MSTGCN** [20]: The Multi-Component Spatial-Temporal Graph Convolution Networks is the simplified version of ASTGCN. MSTGCN removes the attention mechanism from ASTGCN and keeps other components unchanged.
- **GconvGRU** [43]: GconvGRU combines graph convolutions with a variant of LSTMs called gated recurrent units for time series forecasting tasks.

C. Evaluation Metrics

To compare the accuracy of different methods, we use the MAE, MAPE, and RMSE as the evaluation metrics. These metrics are defined below:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \widehat{y_i}| \tag{13}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}|\frac{y_i - \widehat{y_i}}{y_i}| \tag{14}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2} \tag{15}$$

where y_i is the real data and $\widehat{y_i}$ is the predicted data.

We calculate these metrics based on the prediction results of different methods and make a visualization showing the results.

TABLE II
HYPERPARAMETERS OF TRANODE2VEC

hyperparameters	p	q	α	β
value	8	0.125	1.5	2

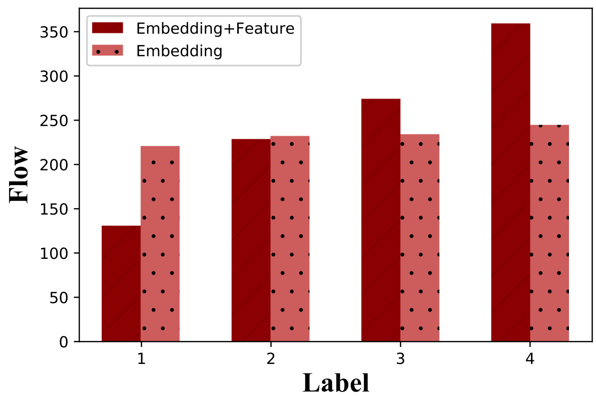
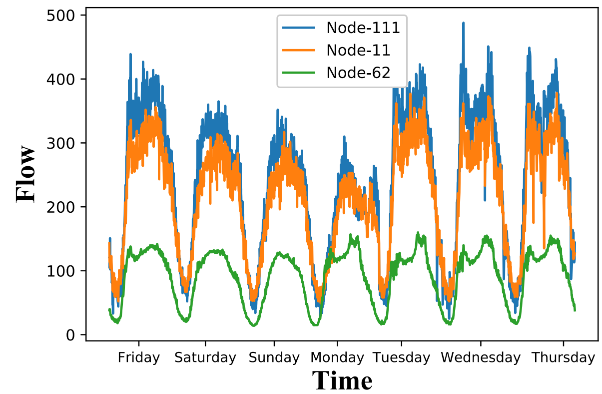


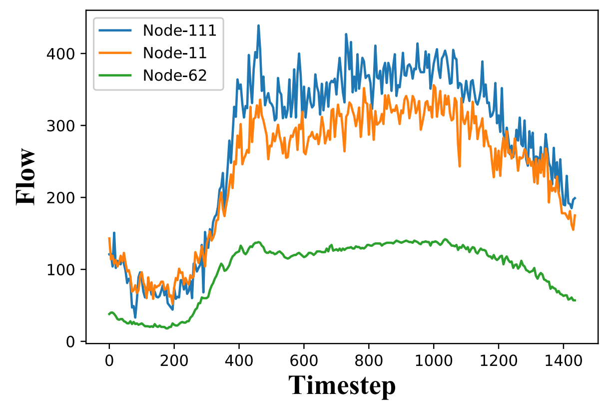
Fig. 7. Average flow of each cluster.

TABLE III
INFORMATION OF EXAMPLE NODES

node i	μ_{flow}	imp	$NCS_{i,111}$
111(source node)	249.11	1.07423	0
11(neighbor node)	220.38	0.95033	1.21283
62(neighbor node)	91.81	0.39587	0.728939



(a)



(b)

Fig. 8. Flow-Time Curve of example node and the neighbor nodes. (a) Week. (b) Day.

V. EXPERIMENTAL RESULTS

In this section, we first show the detailed results of our proposed method NCSGCN, then we compare the evaluation metrics with baseline methods.

A. Proposed Method

The hyperparameters of TraNode2vec used in the experiment are set as Table II shows.

1) Clustering: We compare the clustering effect before and after adding traffic flow features while clustering by calculating the average of the flow of each cluster in Fig. 7.

The legend which shows ‘Embedding+Feature’ in Fig. 7 means adding traffic features while clustering and the other one means clustering only with embedding vectors. We can see that using traffic features can make the difference between different clusters more pronounced which means nodes with similar traffic is grouped into the same category. Clustering only by node embedding vectors performs much worse and cannot make a good distinction in traffic flow. Since using traffic features of nodes help us divide different important nodes better, the $impC$ of each cluster could be calculated more accurately.

2) NCS Evaluation: We choose a node as the example to show the meaning of NCS and analyze the correlation between it and its neighborhoods. The information of nodes we choose is shown in Table III.

We can see that the node with the largest imp has the largest flow. Among its neighbor nodes, node 11 with larger traffic flow has a much larger $NCS_{11,111}$ than that node 62 with less traffic flow. Then we show the flow curve with the time of these nodes in Fig. 8.

As Fig. 8 illustrates, the trend of the flow of node 11 has a greater similarity with node 111 than node 62. The traffic flow from the source node is more towards node 11 than node 62. Fig. 8 suggests that the NCS index is designed to depict and mine this traffic flow correlation between connected nodes.

3) Ablation Study: We predict the next an hour which includes 12 horizons of the traffic flow by using the history data. To analyze the effect of the node connection strength matrix we designed, we also use a 0-1 adjacency matrix and distance matrix to complete the prediction. Finally, we calculate the evaluation metrics of these three methods and compare them in Fig. 9

In Fig. 9, the legend show ‘Strength’ means the method based on the connection strength matrix, the legend shows ’Connectivity’ means the method based on the 0-1 adjacency matrix and the last one means the method based on the reciprocal of the distance matrix. Fig. 9(a) shows the evaluation metrics on the different horizons and Fig. 9(b) shows the average error of each method. We can see that though the error increases as the horizon increases, the method based on the connection strength

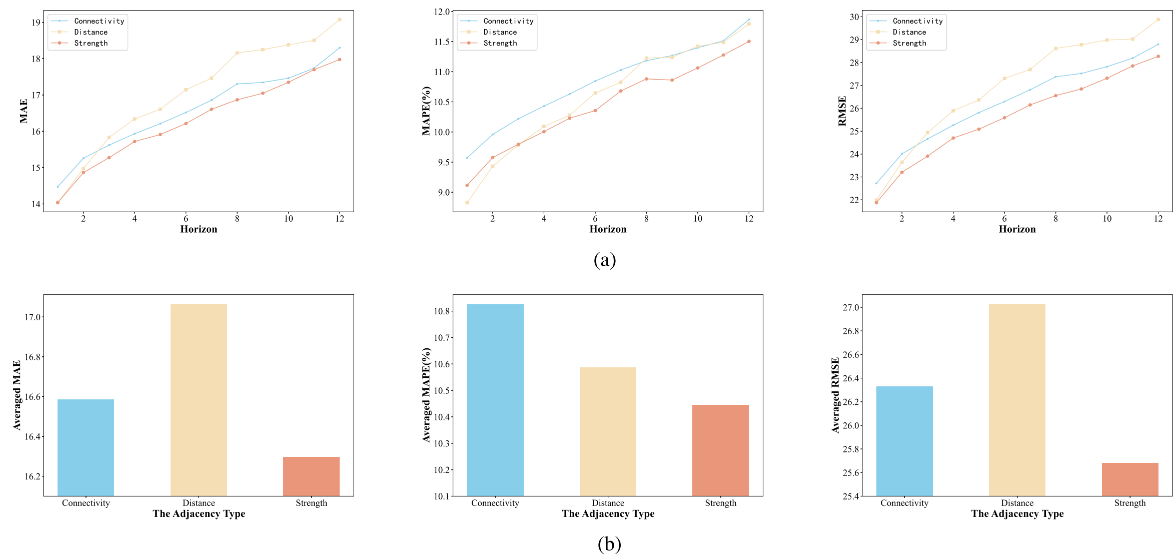


Fig. 9. Accuracy of methods based on different matrices. (a) Accuracy of each horizon. (b) Accuracy of each matrix.

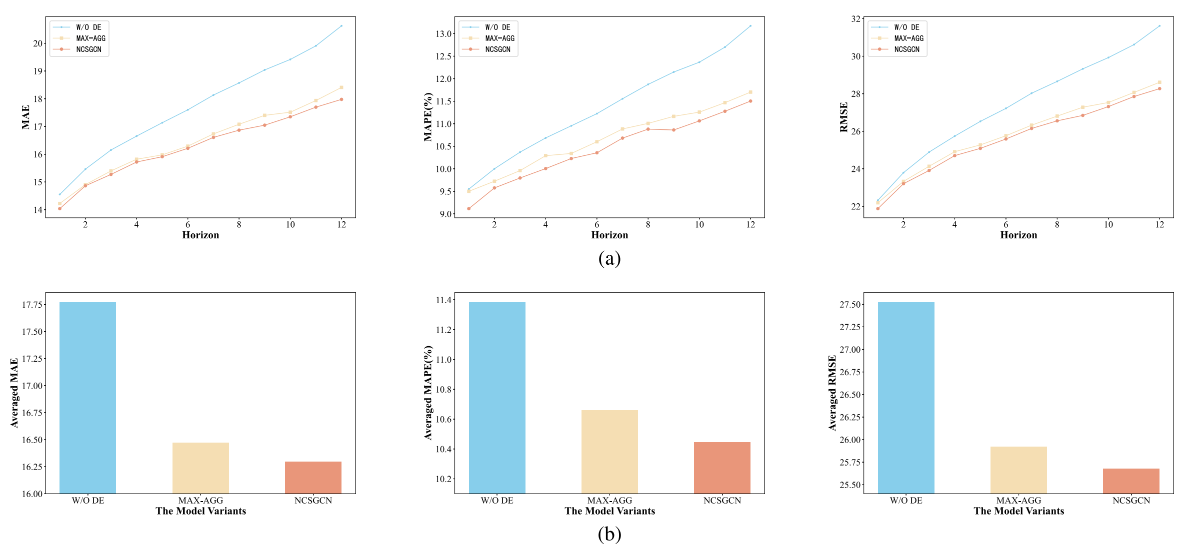


Fig. 10. Comparison of different variants of NCSGCN. (a) Accuracy of each horizon. (b) Mean accuracy of each variant.

matrix has smaller MAE and RMSE on each horizon than the methods based on other matrices. Though the method based on the reciprocal of the distance matrix has a smaller MAPE on the first two horizons, it cannot complete the long-term prediction better than our methods.

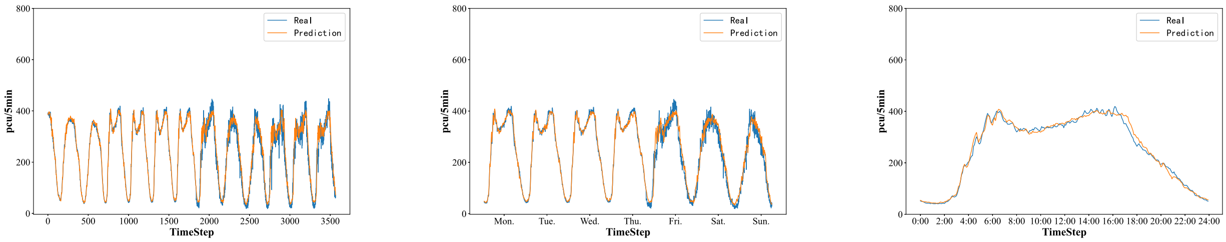
As Fig. 9(b) illustrates, the average error of the method based on the connection strength matrix has an obvious superiority over the others. It means that no matter which metric we consider, the connection strength matrix can bring a significant improvement to prediction results. This provides evidence for the significance of the node connection strength matrix in the traffic flow prediction task.

To verify the effect of the components we proposed, two variants of our model are designed: the NCSGCN with MAX-AGG which uses the max-pooling, and the NCSGCN without DE which removes the Dynamics Extractor. The comparison result is shown in Fig. 10.

Fig. 10(a) shows the evaluation metrics on each horizon and Fig. 10(b) shows the averaged errors of each variant. As Fig. 10 illustrates, Dynamics Extractor can bring an obvious improvement to the performance of the model. It means that Dynamics Extractor successfully captures the variation characteristics of traffic flow. Using Weighted Aggregation instead of Max-Aggregation can also bring an improvement. It means that our model can better capture the information of the local spatial-temporal graph.

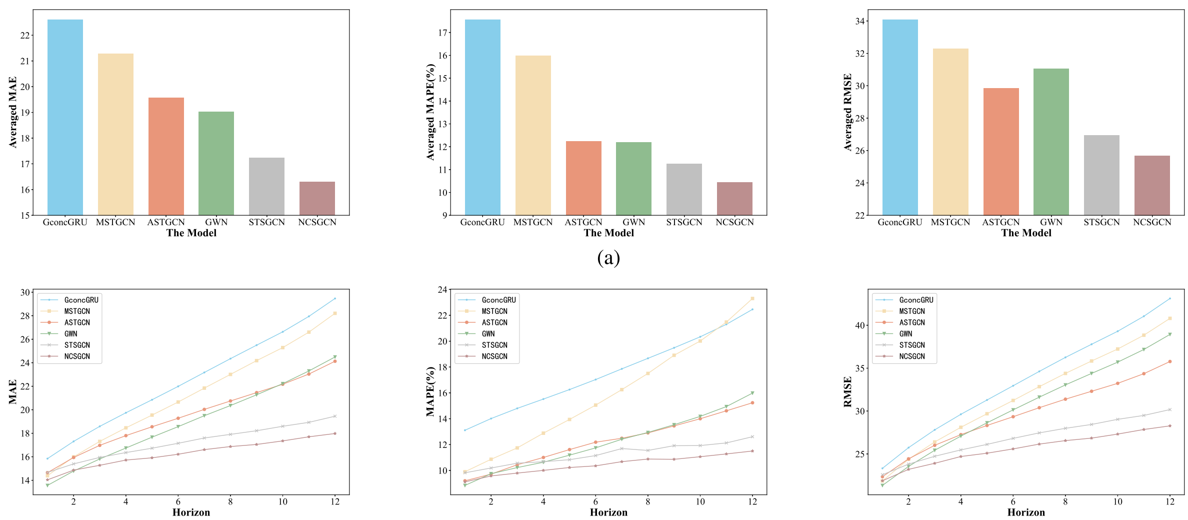
4) Prediction: In order to show the prediction accuracy more directly, we take the prediction flow on the testing set of the 100th node on the road network as an example of our result to complete curve fitting with real data in Fig. 11.

As Fig. 11(a) and (b) illustrate, there is still a little gap between the predicted value of the model and the real value at the peak and low peaks of traffic flow. But from the perspective of the basic trend of traffic flow changes, the predicted situation of traffic



(a) (b) (c)

Fig. 11. Comparison between prediction result and real flow. (a) All prediction. (b) Week. (c) Day.



(a)

(b)

Fig. 12. Accuracy of methods based on different matrices. (a) Average accuracy of each method. (b) Accuracy of each horizon.

TABLE IV
PERFORMANCE COMPARISON OF DIFFERENT MODELS

	MAE	MAPE	RMSE
NCSGCN	**16.30**	**0.1045**	**25.68**
STSGCN	17.24	0.1125	26.94
GWN	19.02	0.1220	31.04
ASTGCN	19.56	0.1224	29.85
MSTGCN	21.29	0.1599	32.30
GconvGRU	22.61	0.1757	34.09

flow is consistent with the changing trend of the real situation over time.

In Fig. 11(c), during the forecast period of one day, the forecast curve is smoother, but the actual flow value fluctuates. From the perspective of traffic theory, this fluctuation is caused by the randomness of vehicle arrivals. The random arrival of vehicles causes the flow rate to increase or decrease monotonically over time. It can only capture the overall change trend and periodicity of the flow rate.

B. Contrastive Analysis

In order to show the improvement for the traffic flow prediction task brought by NCSGCN, we complete the evaluation of baseline methods and make a comparison. The difference between all methods can be seen in Table IV.

Table IV shows the average error of different methods. Fig. 12 shows the error gap between different methods in a more intuitive representation way.

Fig. 12(a) can show the difference in the accuracy of each method more directly and Fig. 12(b) shows the error of each horizon. We can see that only the GWN model can have a better performance than NCSGCN in the first horizon, but it lost good performance on long-term prediction. NCSGCN has the best stability and is always at the bottom. This means that NCGSCN has the best ability on capturing the spatial-temporal dependency of traffic flow data. So NCSGCN has a better prediction effect on the whole in traffic flow prediction task and the smallest average error than baseline methods. And the STSGCN which uses the same $STSGCM$ to capture the spatial-temporal dependency is the second-best method in the comparison. The experiment results also suggest that the $STSGCM$ has good performance in capturing the spatial-temporal dependency of traffic flow data.

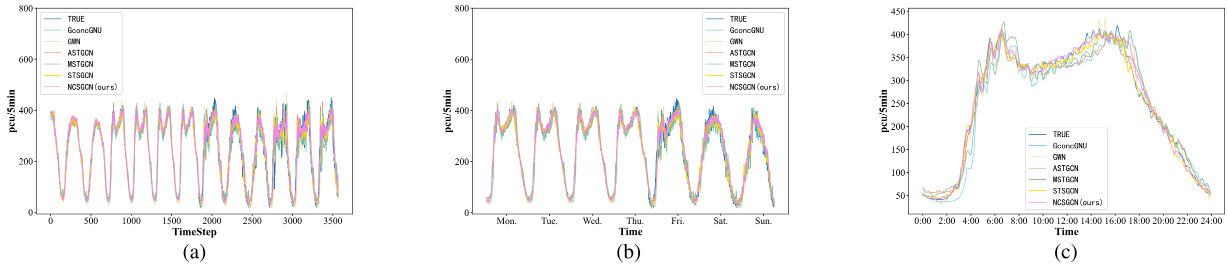


Fig. 13. Comparison of the prediction results with different models. (a) All prediction. (b) Week. (c) Day.

To show the difference between the prediction result and real flow data better, we choose one node as an example to compare the fitting curve in Fig. 13 and analyze the result.

As Fig. 13 illustrates, all the methods can capture the periodicity of flow. However, the coincidence degree of each method and the true value curve are different, the specific data and disturbance are also slightly different, and the fitting degree to the peak value is also different. The accumulation of these small differences over time and nodes causes the overall prediction accuracy gap between the models to become apparent. The visualization of predicted results shows that our proposed model has superiority over existing methods.

VI. Conclusion

Traffic data always contains spatial and temporal dependency, critical information to predict the future flow. Some models capture spatial dependency and temporal dependency respectively. But the experimental results show that models such as NCSGCN, STSGCN, and ASTGCN model can capture the spatial and temporal dependence of traffic network data, and utilize them to complete prediction tasks of future traffic flow, so these methods also have better prediction accuracy than other baseline models.

In this paper, we design the node connection strength matrix to make full use of the traffic features of nodes by combining node embedding and cluster algorithm. As the most important innovation of this paper, the node connection strength matrix can provide more traffic characteristics of the traffic network than the 0-1 adjacency matrix and distance matrix. Then we design the NCSGCN framework with a novel Dynamics Extractor by combining STSGCM with LSTM to get the prediction flow and complete the traffic flow prediction task. The effect of the Dynamics Extractor and the Weighted Aggregation we designed have been verified in experiments. Although we have achieved a predicted result with a smaller error, there is also some room for improvement. From Figs. 11 and 13, we can see the peek period characteristics of traffic flow should be considered. Some other traffic characteristics can be considered which will provide more potential features of traffic flow data such as turning, cart rate, and so on. We will further refine the proposed method in these aspects.

References

[1] W. Wang et al., "Realizing the potential of the Internet of Things for smart tourism with 5G and AI," *IEEE Netw.*, vol. 34, no. 6, pp. 295–301, Nov./Dec. 2020.

[2] X. Yin, G. Wu, J. Wei, Y. Shen, H. Qi, and B. Yin, "Deep learning on traffic prediction: Methods, analysis and future directions," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 4927–4943, Jun. 2022.

[3] A. Boukerche and J. Wang, "Machine learning-based traffic prediction models for intelligent transportation systems," *Comput. Netw.*, vol. 181, 2020, Art. no. 107530.

[4] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, and A. Souza, "Self-driving cars: A survey," *Expert Syst. Appl.*, vol. 165, 2021, Art. no 113816.

[5] N. G. Polson and V. O. Sokolov, "Deep learning for short-term traffic flow prediction," *Transp. Res. Part C. Emerg. Technol.*, vol. 79, pp. 1–17, Jun. 2017.

[6] I. M. Wagner-Muns, I. G. Guardiola, V. A. Samaranayke, and W. I. Kayani, "A functional data analysis approach to traffic volume forecasting," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 3, pp. 878–888, Mar. 2018.

[7] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, 2003.

[8] X. Yin, G. Wu, J. Wei, Y. Shen, H. Qi, and B. Yin, "Deep learning on traffic prediction: Methods, analysis, and future directions," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, 2020, Art. no. 4927.

[9] W. Li et al., "A general framework for unmet demand prediction in on-demand transport services," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 8, pp. 2820–2830, Aug. 2019.

[10] Z. Diao et al., "A hybrid model for short-term traffic volume prediction in massive transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 3, pp. 935–946, Mar. 2019.

[11] P. Duan, G. Mao, W. Liang, and D. Zhang, "A unified spatio-temporal model for short-term traffic flow prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 9, pp. 3212–3223, Sep. 2019.

[12] G. Lin, A. Lin, and D. Gu, "Using support vector regression and k-nearest neighbors for short-term traffic flow prediction based on maximal information coefficient," *Inf. Sci.*, vol. 608, pp. 517–531, 2022.

[13] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, vol. 29.

[14] Z. Zheng, W. Chen, X. Wu, P. Chen, and J. Liu, "LSTM network: A deep learning approach for short-term traffic forecast," *IET Intell. Transport Syst.*, vol. 11, no. 2, pp. 68–75, 2017.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, "Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4883–4894, Nov. 2020.

[17] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci. USA*, vol. 79, no. 8, pp. 2554–2558, 1982.

[18] X. Wang et al., "Traffic flow prediction via spatial temporal graph neural network," in *Proc. Web Conf.*, 2020, pp. 1082–1092.

[19] X. Zhang et al., "Traffic flow forecasting with spatial-temporal graph diffusion network," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, pp. 15008–15015.

[20] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, pp. 922–929.

[21] K. Wang, C. Ma, Y. Qiao, X. Lu, W. Hao, and S. Dong, "A hybrid deep learning model with 1DCNN-LSTM-attention networks for short-term traffic flow prediction," *Phys. A: Stat. Mech. Appl.*, vol. 583, 2021, Art. no. 126293.

[22] H. Peng et al., "Dynamic graph convolutional network for long-term traffic flow prediction with reinforcement learning," *Inf. Sci.*, vol. 578, pp. 401–416, 2021.

[23] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," 2017, *arXiv:1709.05584*.

[24] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network representation learning: A survey," *IEEE Trans. Big Data*, vol. 6, no. 1, pp. 3–28, Mar. 2020.

[25] W. L. Hamilton, "Graph representation learning," *Synth. Lectures Artif. Intell. Mach. Learn.*, vol. 14, no. 3, pp. 1–159, 2020.

[26] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2017, vol. 30.

[27] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 701–710.

[28] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 855–864.

[29] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 1067–1077.

[30] N. ThomasKipf, and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.

[31] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016, *arXiv:1611.07308*.

[32] W. Wang et al., "Vehicle trajectory clustering based on dynamic representation learning of internet of vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3567–3576, Jun. 2021.

[33] X. Huang, J. Chen, M. Cai, W. Wang, and X. Hu, "Traffic node importance evaluation based on clustering in represented transportation networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 16622–16631, Sep. 2022.

[34] J. Ye, J. Zhao, K. Ye, and C. Xu, "How to build a graph-based deep learning architecture in traffic domain: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 5, pp. 3904–3924, May 2022.

[35] H. Mo and Y. Deng, "Identifying node importance based on evidence theory in complex networks," *Physica A: Stat. Mech. Appl.*, vol. 529, 2019, Art. no. 121538.

[36] L. Wang, M. Huang, and L. Gao, "Methods of importance evaluation of traffic network node based on clustering algorithms," *J. Transport Inf. Saf.*, vol. 38, no. 2, pp. 80–88, 2020d.

[37] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: An overview," *Wiley Interdiscipl. Rev.: Data Mining Knowl. Discov.*, vol. 2, no. 1, pp. 86–97, 2012.

[38] X. Li, X. Ying, and M. C. Chuah, "Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving," 2019, *arXiv:1907.07792*.

[39] A. Vaswani et al., "Attention is all you need," in *Proc. Int. Adv. Conf. Neural Inf. Process. Syst.*, 2017, vol. 30.

[40] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, pp. 914–921.

[41] D. Hendrycks and K. Gimpel, "Gaussian error linear units (GELUS)," 2016, *arXiv:1606.08415*.

[42] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," 2019, *arXiv:1906.00121*.

[43] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson, "Structured sequence modeling with graph convolutional recurrent networks," in *Proc. Int. Conf. Neural Inf. Process.*, 2018, pp. 362–373.

Jian Chen is currently working toward the bachelor's degree in transportation engineering with the School of Intelligent Engineering, Sun Yat-sen University, Shenzhen, China. His research interests include transportation network analysis and intelligent transportation.

Wei Wang (Member, IEEE) received the Ph.D. degree in software engineering from the Dalian University of Technology, Dalian, China, in 2018. He is currently a Professor with the School of Medical Technology, Beijing Institute of Technology, Beijing, China, and also with the Department of Engineering, Shenzhen MSU-BIT University. He had been the UM Macao Research Fellow with the University of Macau, Macau. His research interests include computational social science, data mining, Internet of Things, and artificial intelligence.

Keping Yu received the M.E. and Ph.D. degrees from Waseda University, Tokyo, Japan, in 2012 and 2016, respectively. He was a Research Associate, Junior Researcher, and Researcher with the Global Information and Telecommunication Institute, Waseda University, Tokyo, Japan, from 2015 to 2019, from 2019 to 2020, and from 2020 to 2022, respectively. He is currently an Associate Professor with the Graduate School of Science and Engineering, Hosei University, Tokyo, and a Visiting Scientist with the RIKEN Center for Advanced Intelligence Project, RIKEN, Saitama, Japan. His main research interests include Internet of Things, artificial intelligence, blockchain, and information security.

Xiping Hu received the Ph.D. degree from the University of British Columbia, Vancouver, BC, Canada. He is currently a Professor with the School of Medical Technology, Beijing Institute of Technology, Beijing, China, and also with the Department of Engineering, Shenzhen MSU-BIT University, Shenzhen, China. He is the co-Founder and a Chief Scientist of Erudite Education Group Limited, Hong Kong, a leading language learning mobile application company with over 100 million users, and listed as a top two language education platform globally. His research interests include mobile cyberphysical systems, crowdsensing, social networks, and cloud computing.

Ming Cai received the B.E. and Ph.D. degrees from Sun Yat-sen University, Guangzhou, China, in 1999 and 2004, respectively. He is currently a full-time Professor with the School of Intelligent Systems Engineering, Sun Yat-sen University, Guangzhou, China. His main research interests include traffic Big Data, traffic environmental engineering and intelligent transportation system.

Mohsen Guizani (Fellow, IEEE) received the B.S. (with Distinction), M.S., and Ph.D. degrees in electrical and computer engineering from Syracuse University, Syracuse, NY, USA. He is currently a Professor with the Machine Learning Department, Mohamed Bin Zayed University of Artificial Intelligence (MBZUAI), Abu Dhabi, UAE. He has authored nine books and more than 800 publications. His research interests include wireless communications and mobile computing, applied machine learning, cloud computing, security, and its application to healthcare systems.