Journal of Advanced Transportation

Traffic Data Modeling with Graph Neural Networks

Lead Guest Editor: Yong Zhang
Guest Editors: Junbin Gao and Yan-Ming Shen

Traffic Data Modeling with Graph Neural Networks

Journal of Advanced Transportation

Traffic Data Modeling with Graph Neural Networks

Lead Guest Editor: Yong Zhang
Guest Editors: Junbin Gao and Yan-Ming Shen

Contents

Hindawi
Journal of Advanced Transportation
Volume 2023, Article ID 7152010, 13 pages
https://doi.org/10.1155/2023/7152010

Research Article

PGDRT: Prediction Demand Based on Graph Convolutional Network for Regional Demand-Responsive Transport

Eunkyeong Lee,[1] **Hosik Choi**,[1] **and Do-Gyeong Kim**[2]

[1]*Department of Urban Big Data Convergence, University of Seoul, Seoul 02504, Republic of Korea*
[2]*Department of Transportation Engineering & Graduate School, Department of Urban Big Data Convergence, University of Seoul, Seoul 02504, Republic of Korea*

Correspondence should be addressed to Do-Gyeong Kim; dokkang@uos.ac.kr

Received 3 June 2022; Accepted 13 September 2022; Published 5 January 2023

Academic Editor: Yanming Shen

To provide an efficient demand-responsive transport (DRT) service, we established a model for predicting regional movement demand that reflects spatiotemporal characteristics. DRT facilitates the movement of restricted passengers. However, passengers with restrictions are highly dependent on transportation services, and there are large fluctuations in travel demand based on the region, time, and intermittent demand constraints. Without regional demand predictions, the gaps between the desired boarding times of passengers and the actual boarding times are significantly increased, resulting in inefficient transportation services with minimal movement and maximum costs. Therefore, it is necessary to establish a regional demand generation prediction model that reflects temporal features for efficient demand response service operations. In this study, a graph convolutional network model that performs demand prediction using spatial and temporal information was developed. The proposed model considers a region's unique characteristics and the influence between regions through spatial information, such as the proximity between regions, convenience of transportation, and functional similarity. In addition, three types of temporal characteristics—adjacent visual characteristics, periodic characteristics, and representative characteristics—were defined to reflect past demand patterns. With the proposed demand forecasting model, measures can be taken, such as having empty vehicles move to areas where demand is expected or encouraging adjustment of the vehicle's rest time to avoid congestion. Thus, fast and efficient transportation satisfying the movement demand of passengers with restrictions can be achieved, resulting in sustainable transportation.

1. Introduction

The right to travel refers to citizens' right to move freely and safely. Because it is a fundamental right that is indispensable to human life, efforts to ensure the right to move continuously are needed [1]. Although transportation patterns have changed over the past few decades, mainstream passenger transport (e.g., buses and taxis) has not changed sufficiently to meet these changes. In particular, timed route methods, such as buses, incur fixed operating costs. If a passenger is not picked up, a loss occurs; if the passenger's demand changes, the utilization rate decreases, and eventually, the fixed cost increases. This leads to a vicious cycle that results in a decrease in use, because the service does not adequately satisfy the requirements of passenger travel. If this phenomenon persists, supply is concentrated on major routes, which can create barriers to passengers' travel rights. Moreover, socially disadvantaged people (elderly people, disabled people, residents of vulnerable areas, etc.) can experience severe isolation. Demand-responsive transport (DRT) services have emerged to solve this problem.

A DRT service refers to a transportation service that responds to the movement demand of passengers without a predetermined route or operation plan. It combines low fares, which are the advantages of buses running fixed routes, and convenient boarding and disembarking and

speed, which are the advantages of taxis. Therefore, relative to buses and taxis, DRT services achieve a tradeoff in terms of efficiency and cost. DRT services have the following advantages over fixed-route operations. First, the demand resolution is optimized. For DRT services, the driving distance of a fixed-route vehicle divided by the number of passengers onboard is approximately half that for a fixed-route operation. Additionally, DRT services have the advantage of efficient operational cost management. DRT services are economical because the fixed cost incurred when there is no demand is low. Another advantage of DRT services is their environmental superiority. They have a shorter tolerance distance than fixed-route vehicles. They are ecofriendly with regard to greenhouse-gas emissions and fuel consumption because they use small vehicles. Finally, passengers are highly satisfied with DRT services. DRT services operating in a door-to-door manner achieve higher levels of passenger satisfaction than fixed-route operations, where passengers must travel directly to the station [2].

DRT is applied to the movement of passengers with restrictions, e.g., in areas where demand is intermittent or transportation services are insufficient and vulnerable [3]. Real-time response to the travel demand is crucial for efficient DRT service operations, requiring a system and demand forecasting model to allocate requests to vehicles quickly and efficiently when passengers receive travel requests [4]. The demand forecasting field for mainstream passenger transport continues to improve with the development of deep-learning technologies such as long short-term memory (LSTM). For example, in [5], LSTM was utilized to predict future demand according to past demand through traffic card data analysis. However, DRT services are designed for the movement of passengers with restrictions; therefore, they exhibit a different demand pattern from general mainstream passenger transportation. Because the existing liquor passenger transportation model cannot be applied, a model that reflects the movement characteristics of passengers with restrictions is required.

It is crucial to consider the demand at previous times in the region, but it is also essential to reflect spatial characteristics. Each region has spatial characteristics, such as commercial districts and suburban areas [6, 7]. Because spatial characteristics affect temporal trends, spatiotemporal factors must be considered. In this study, three types of components that reflect spatial, temporal, and spatiotemporal characteristics were constructed and reflected in the model. Because DRT services are subject to spatiotemporal influences, the data are sparse. LSTM affiliation is not well suited for sparse data. To solve this problem, we used channel-wise attention and temporal means to alleviate the sparsity of the data to the greatest extent possible and then used ConvLSTM.

The main contributions of this study are as follows.

(i) First, we improved the interpretability of the model by identifying the cause of spatiotemporal demand and reflecting it in the model

(ii) Second, we used channel-wise attention and temporal means to maximize the demand for sparse demand response

(iii) Finally, a graph convolutional network (GCN) was used for the first time to reflect spatial factors in demand prediction according to the region of the DRT service

The remainder of this paper is organized as follows. Section 2 presents related research and basic deep-learning models related to DRT service demand prediction. Section 3 presents the proposed method. Section 4 presents the results of applying the proposed method to actual data. Section 5 presents conclusions and suggestions for further research.

2. Related Works and Preliminaries

This section introduces DRT service demand prediction research and deep-learning methods.

2.1. DRT Service Demand Prediction. Because demand prediction must precede the efficient operation of DRT services, many studies have recently been conducted using various methodologies. For example, in [8], after the entire region was divided into grids, the demand for a DRT service was predicted using a convolutional neural network (CNN), LSTM, and ConvLSTM, along with exogenous variables such as weather. In [9], an appropriate DRT type was identified by estimating the average number of people getting on and off at bus stops in a regular pattern identified through cluster classification of time-by-time boarding points for the efficient placement of DRT.

Recent studies focus on spatial dependence, traveler personal heterogeneous, sparse uncertainty, and demand prediction quality requirements. Reference [10] mentioned that variables representing factors related to the characteristics of service supply, demographic characteristics, land use, and accessibility should be discovered and fused to reflect the direct impact and ripple effect on demand. Their research uses a model structure (Attention, ConvLSTM) that can demonstrate demand patterns of call taxis for the disabled as a service supply characteristic. In addition, to reflect demographic characteristics, the administrative region, which is a division of a population-based area, was used as variables representing factors related to land use and accessibility were discovered and utilized as a functional similarity adjacency matrix of the GCN method. However, this paper is aimed at developing an optimal bus route rather than a DRT service. Call taxi for the disabled is a short-distance transportation service for people who cannot go to the appropriate stop due to severe disabilities. There is a separate long-distance customized bus service for the disabled in Seoul. Therefore, the use of the call taxi for the disabled is different. Reference [11] is a thesis that studies the error distribution rather than specific parameters, learning methods, and hyperparameter adjustments for a transportation demand prediction model for adequate public transportation (PT) operation. To build an accurate model, it is necessary to study the error distribution considered in the study. References [12, 13] utilized H-ConvLSTM that applies convolution based on a hexagonal shape rather than a

conventional pixel standard. We improved the performance by using the ensemble for postaggregation, like bagging. To reflect the interregional relationship between hexagons, they used the GCN additionally.

In particular, the traffic demand was predicted using call taxi data for people with disabilities in Seoul. In [14], the waiting times for disabled people in Seoul were predicted using SARIMA and LSTM and compared. In [15], the call taxi latency for the disabled was predicted using several hyperparameters of LSTM. However, in these studies, only past temporal characteristics were considered; spatial characteristics were omitted or reflected only in the Euclidean space. Furthermore, because the spatial relationship is not based only on the location in Euclidean distance, it is necessary to reflect various spatial structures based on non-Euclidean distance in the model.

2.2. Spatiotemporal Prediction. Demand prediction and urban traffic prediction fields, such as traffic volume prediction and congestion distribution estimation, exist in tasks that reflect spatiotemporal factors. Previous studies on urban traffic prediction can be classified into two categories according to the input data format. Grid-based inflow and outflow prediction is based on images, whereas graph-based traffic speed prediction is based on graphs.

2.2.1. Grid-Based Inflow and Outflow Prediction. The demand forecasts for DRT services and taxis are highly similar [8]. Therefore, to predict the general taxi demand, the entire area is converted into an image set to a grid of a specific size and utilized. In [16], exogenous variables such as weather and weekend availability were added in a fully connected layer. The values before a certain point, such as the distant, near, recent of the grid, are learned through convolution. In [6], predetermined point-of-time values and point-of-interest (POI) characteristics were learned by a grid through convolution, such as the time, day, and week of the set grid, and combined through ResPlus to predict the regional taxi demand at the next time. Collecting exogenous variables that may be related to future demand can improve the predictive performance. Although weather, POI, or traffic flow was used in the foregoing studies, the performance improvement was insignificant relative to the increase in the number of parameters, because the improvement through exogenous variables was orthogonal to capturing complex spatiotemporal dependence in the data [7].

2.2.2. Graph-Based Traffic Speed Prediction. Graph structures—not images—are used to solve various urban problems. In contrast to the grid-based method, research is focused on solving various urban problems, such as predicting traffic speed, rather than predicting movement demand. For example, in [17], a GCN with three adjacency matrices was used. Spatial characteristics were adopted, along with a contextual gated recurrent neural network [14, 18, 19] and temporal characteristics with values prior to a certain point in time of closeness, period, and trend. In [20], a three-part model that predicts the travel demand at the next point in time was proposed. The first part is a long-term encoder for encoding the past moving demand. The second part is a short-term encoder for deriving next-step predictions from generated multistep predictions. The third part is an attention-based output module for modeling dynamic temporal and channel-wise information. In [21], ST-Conv block—a combination of temporal-gated convolution and spatial graph convolution—was used to predict the traffic speed at the next point in time. In this study, we predicted the demand for a DRT service using the graph-based method.

2.3. GCN. The GCN applies to graph $G = (V, A)$, where V refers to vertices and $A \in \mathbb{R}^{|V|\times|V|}$ is a matrix with edges expressing the relationships between the vertices. The GCN can extract a local feature from a non-Euclidean structure in another receptive field. For example, to utilize convolution in the graph structure, the Fourier transform [22] can be used. To share the basis of the Fourier transform, we compute the Laplacian matrix

$$L = I - D^{-1/2}AD^{-1/2}, \tag{1}$$

where D denotes the degree matrix. We denote X^l as the features of the l^{th} layer, α^k as trainable coefficients, L^k as the k-order multiplier of the graph Laplacian matrix, and σ as an activation function. The graph convolution operation [23] using a Laplacian matrix is defined as follows:

$$X_{l+1} = \sigma\left(\sum_{k=0}^{K-1} \alpha_k L^k X_l\right). \tag{2}$$

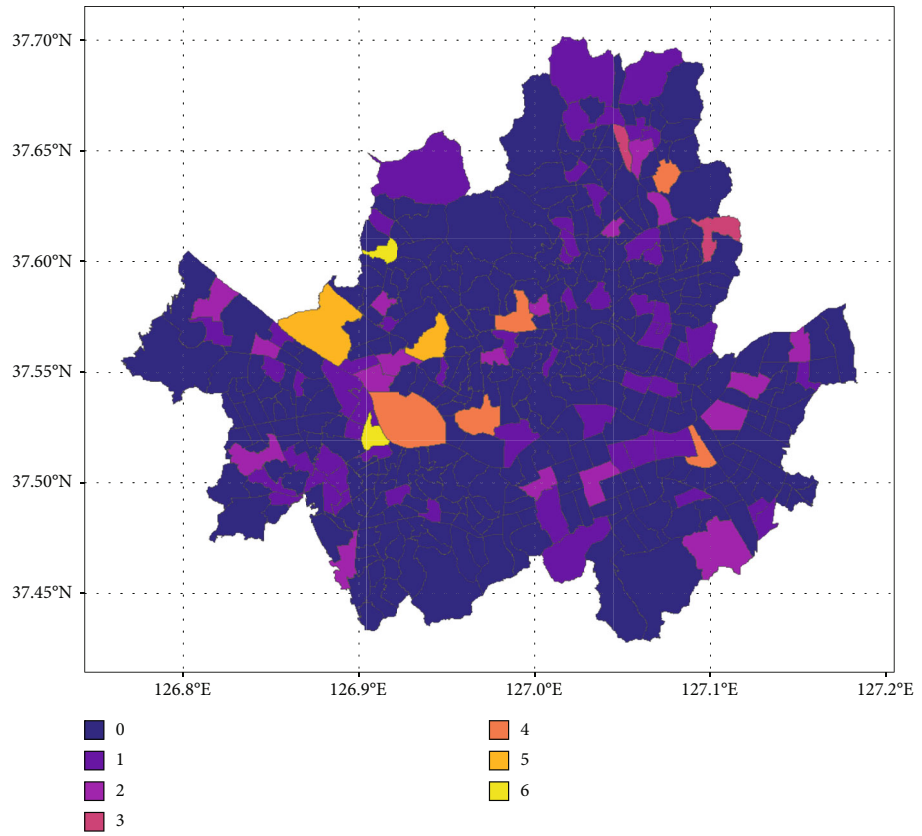
We learn the relationships between adjacent vertices by updating feature X through multiple layers. Moreover, because the GCN has the characteristics of learning weight sharing and local features, which are characteristics of the CNN, it is possible to obtain a node feature reflecting the connection information of the adjacent (hop) nodes of each node.

2.4. Channel-Wise Attention. Given an input $X \in \mathbb{R}^{W\times H\times C}$, channel-wise attention [17, 18] learns the weights for each channel to find and highlight the most important frame with larger weights. Here, H, W, and C refer to the height, width, and channel number of the image, respectively. The channel-wise attention is defined as follows. A summary of each channel

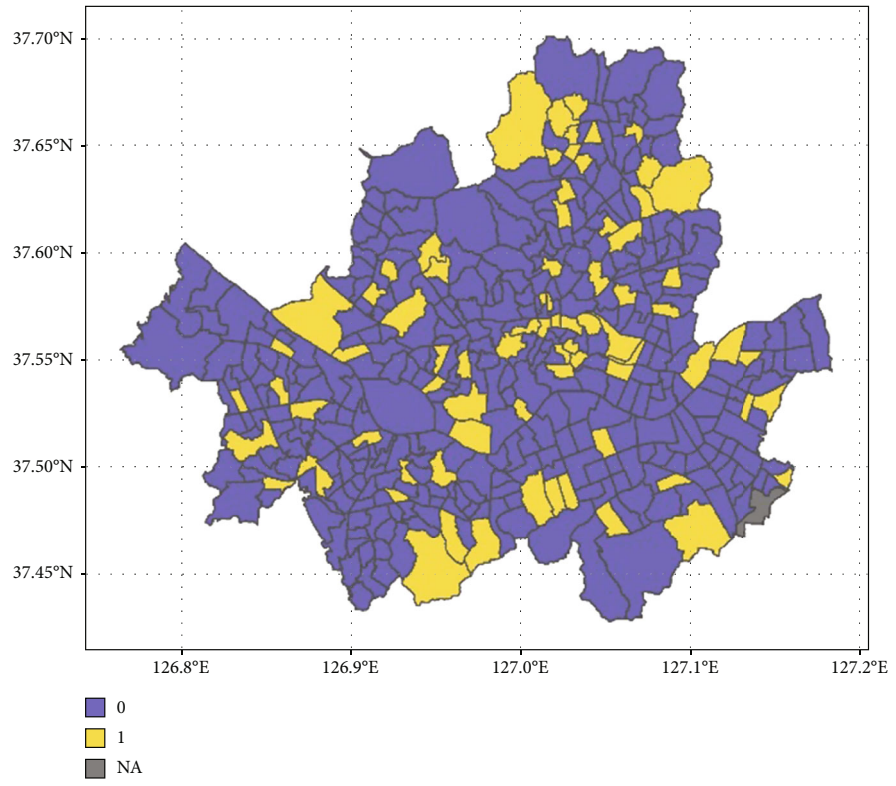
$$z_c = F_{\text{pool}}(\mathbf{X}_{:,:,c}) = \frac{1}{WH}\sum_{i=0}^{W}\sum_{j=0}^{H} X_{i,j,c} \text{ for } c = 1, \cdots, C \tag{3}$$

is obtained. Then, we obtain the attention $s = \sigma(\mathbf{W_2}\delta(\mathbf{W_1}\mathbf{z}))$. The algorithm learns to assign a large weight to the important channels. The attention value to the original input values is channel-wise as follows:

$$\tilde{\mathbf{X}}_{:,:,c} = \mathbf{X}_{:,:,\mathbf{c}} \odot \mathbf{s_c}, \text{ for } \mathbf{c} = \mathbf{1}, \cdots, \mathbf{C}. \tag{4}$$



(a)



(b)

Figure 1: Distributions of demands at 5 p.m. on November 1, 2019. (a) Distribution of demands (continuous) at 5 p.m. (b) Distribution of demands (0/1) at 5 p.m.

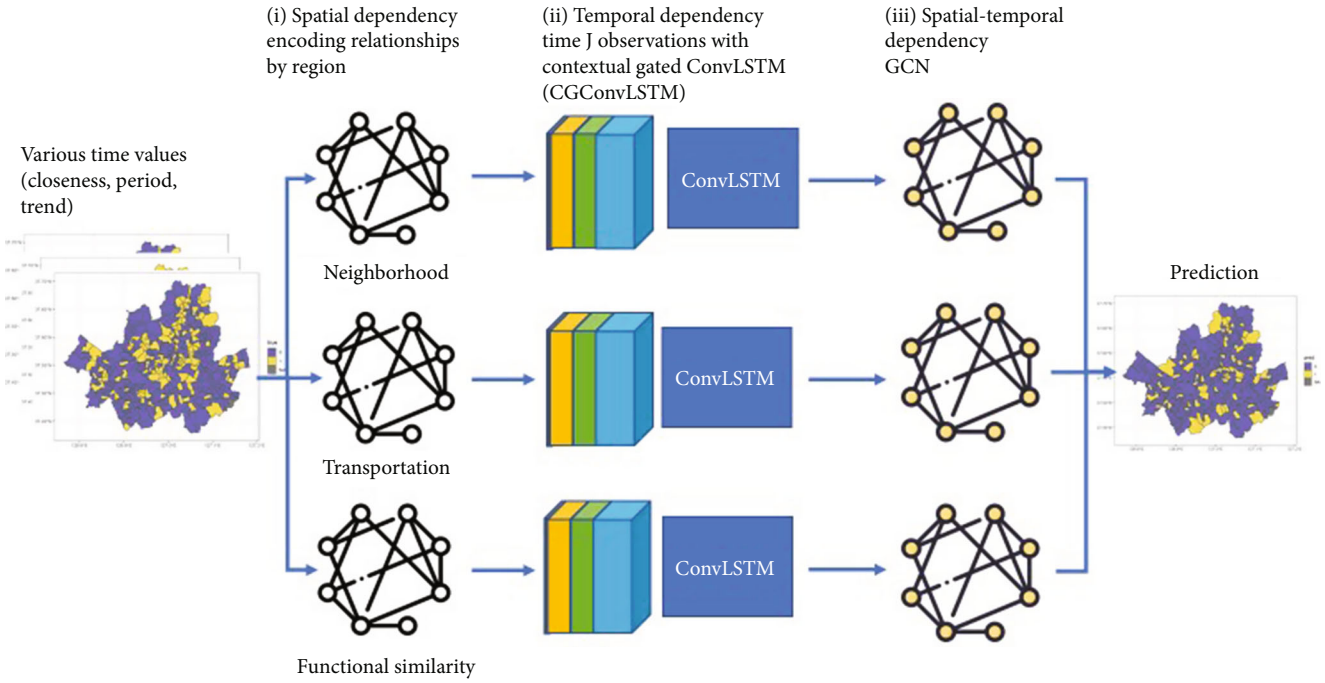


Figure 2: Model overview.

Require:

1: Past demands $\mathbf{X} = (x_i^{(\text{trend})}, x_i^{(\text{period})}, x_i^{(\text{closeness})})_{i=1}^{|V|-1}$

2: Future true demand $y_i = (x_i^{(t+1)})_{i=0}^{|V|-1}$

3: Adjacency matrix $\mathbf{A} = (A_N, A_T, A_F)$, degree matrix $\mathbf{D} = (D_N, D_T, D_F)$, hop: K

Ensure: future prediction demand $\widehat{y_i} = (\widehat{x}_i^{(t+1)})_{i=0}^{|V|-1}$

4: **While** training **do**

5: For all A do

6: (1) Spatial dependency: apply Chebyshev to each adjacency matrix $\mathbf{A}$

7: $\tilde{L} \longleftarrow$ rescale (normalize (L))

8: **For all K do**

9: $\mathbf{T}_{k+1} \longleftarrow$ Chebyshev $(\tilde{L}, \mathbf{T}_k)$

10: **End for**

11: (2) Temporal dependency: apply with contextual gating (CG) and ConvLSTM

12: $H_i \longleftarrow$ ConvLSTM$(CG(\mathbf{X}, \mathbf{T}_{k+1}))$

13: **End for**

14: (3) Spatial-temporal dependency: apply with FC (fully connected) and GCN (graph convolution network)

15: $y_i \longleftarrow \widehat{\text{FC}(\text{GCN}(Hi))}$

16: Compute loss: $L = \text{BCELoss}(\widehat{y_i}, y_i)$

17: **End while**

Algorithm 1: Training procedure of the proposed method.

Here, $\mathbf{F_{pool}}$ is a global average pooling operation, and $\mathbf{W_1}$ and $\mathbf{W_2}$ are the corresponding weights. δ and σ are nonlinear functions for each ReLU, i.e., rectified linear unit and sigmoid function.

3. Method

3.1. Description of Dataset. In this study, DRT service data of call taxis for the disabled in Seoul for two years (from 00:00 on January 1, 2018, to 24:00 on December 31, 2019) were used. The call taxis were primarily operated in Seoul but sometimes moved to areas adjacent to Seoul, depending on the passenger demand. However, we limited the spatial range to Seoul. Therefore, we included data from both departure and destination sets within Seoul. The call taxi data for the disabled included the following information. For each call, the variables were the type of call (regular reception, full-day reservation, and direct call), reception, hope, dispatch, boarding, departure, destination, departure coordinates, customer number, purpose of use, and number

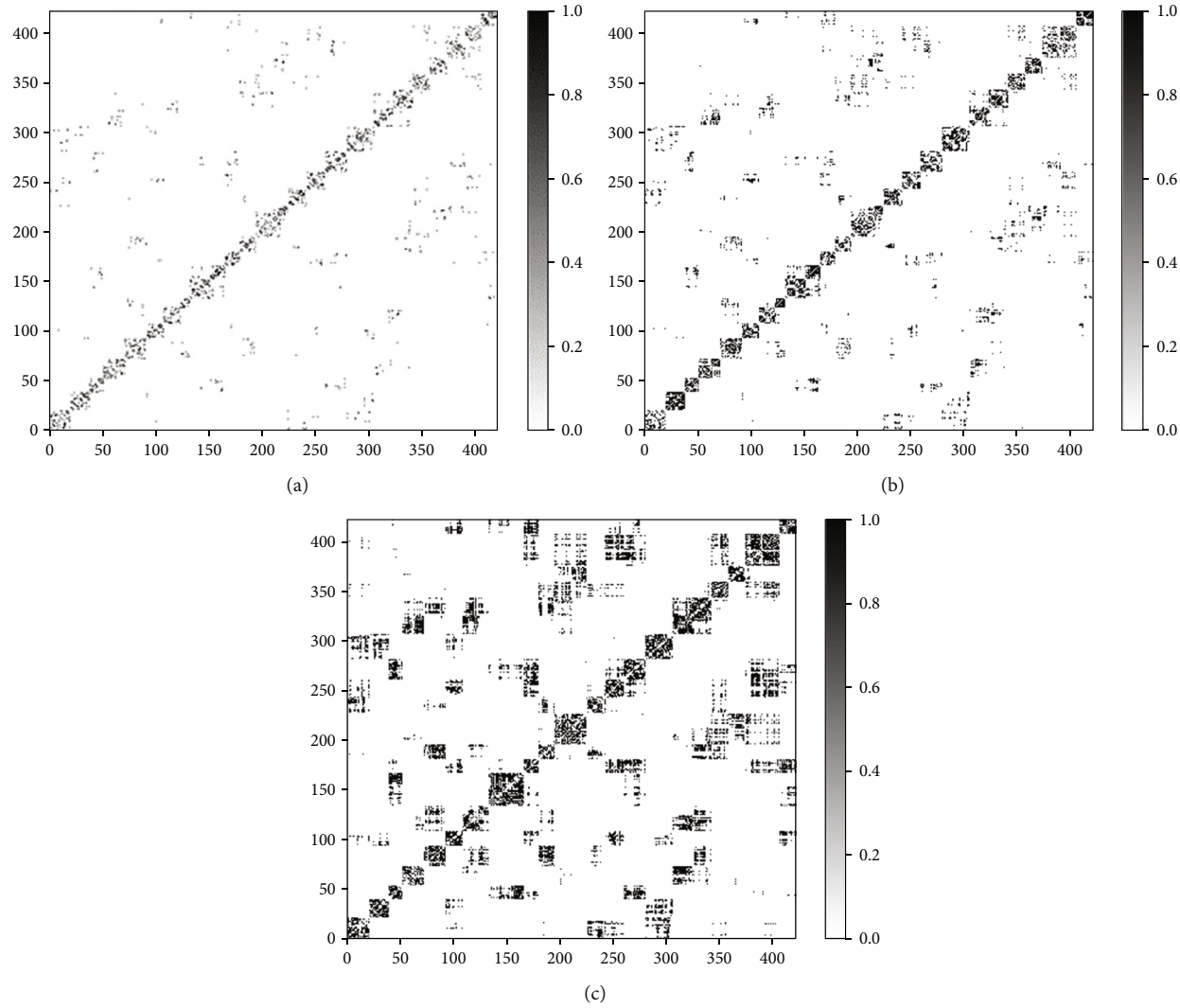


Figure 3: Adjacency matrices for neighborhood, transportation, and functional similarity. (a) Neighborhood adjacency. (b) Transportation adjacency. (c) Functional-similarity adjacency.

of boarding vehicles. Of the 424 administrative districts, Wirye-dong, which had no passenger demand in 2018 or 2019, was excluded. In addition, we excluded data corresponding to hours other than the primary operating hours. There were 1,699,614 data points within 11 h, including 7–17 h.

The data contained one row of demand consisting of a three-dimensional matrix with 8030 rows and 423 (administrative districts) columns in 365 days × 2 (years) × 11 (time zones) by aggregating the number of demands by administrative district in the date-time period. The number of demand cases was continuous data; however, as mentioned previously, the number of demand cases had an extensive and intermittent distribution. Zero accounted for 62% of the cases (1,064,141 of 1,699,614), one accounted for 21%, and the others accounted for only 17%. The class imbalance problem was alleviated by treating multiple demands as one demand (0/1). For example, for 5 p.m. on November 1, 2019, the data exhibited a wide variety of demands, as shown in Figure 1(a). However, the number of demands was changed according to whether there was demand, as shown in Figure 1(b).

3.2. Proposed Method. The proposed method consists of three steps. The first step is to encode the spatial dependency, the second step is to use ConvLSTM [24] to reflect the temporal dependency, and the third step is to use a GCN [23] to reflect the temporal dependency. Figure 2 illustrates the overall process. Furthermore, pseudocode is presented in Algorithm 1.

3.3. Encoding Spatial Dependency. The proposed method utilizes several types of adjacency matrices to reflect the spatial dependency. The adjacency matrix A_N reflects the neighborhood between administrative districts.

$$A_{N,i,j} = \begin{cases} 1, v_i \text{ and } v_j \text{ are adjacent,} \\ 0, \text{otherwise} \end{cases} \quad (5)$$

Figure 3(a) shows a heat map of the adjacency matrix for adjacent connections between administrative districts. The second adjacency matrix A_T was designed to reflect the real

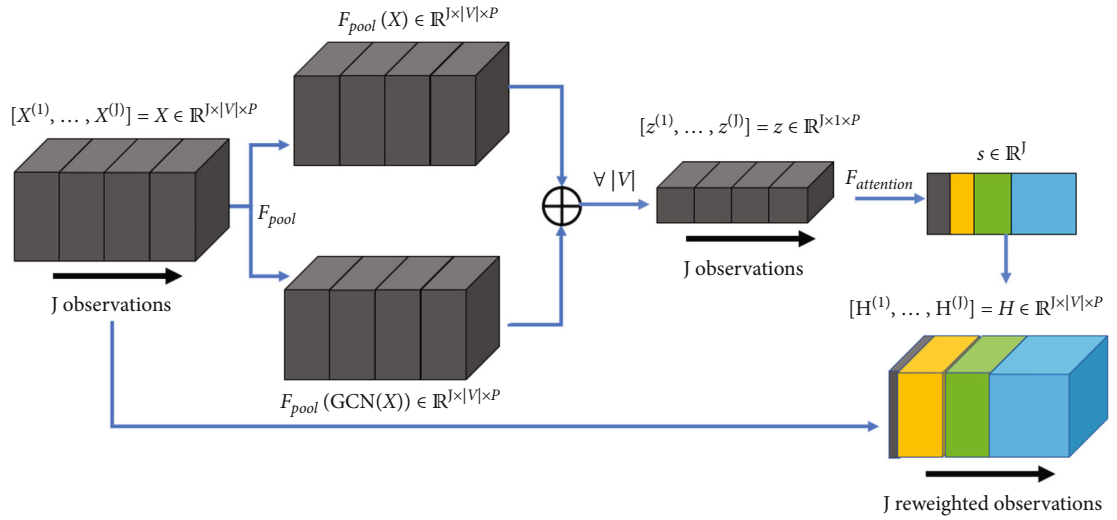


Figure 4: Contextual gating mechanism of the proposed method.

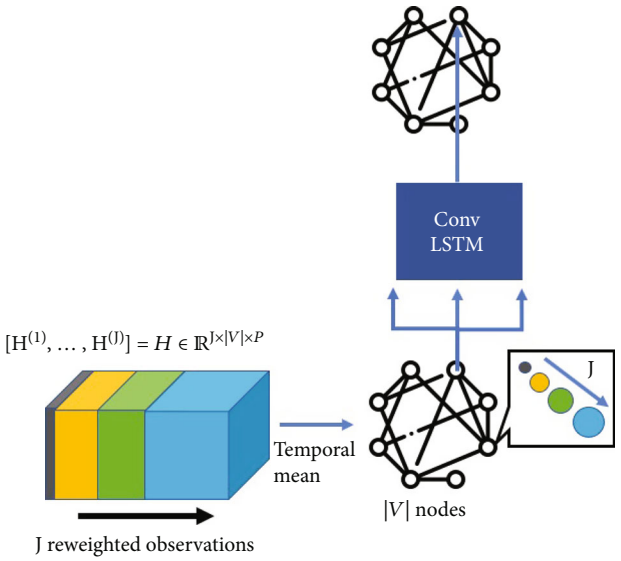


Figure 5: Contextual gating mechanism.

travel distance between the administrative districts.

$$A_{T,i,j} = \max\left(0, \mathrm{conn}\left(v_i, v_j\right) - A_{N,i,j}\right) \in \{0, 1\}. \quad (6)$$

Figure 3(b) shows a heat map of the adjacency matrix for the transportation convenience connection between administrative districts. According to the third adjacency matrix A_F, for administrative districts that are more functionally similar, the demand patterns are more similar.

$$A_{F,i,j} = I\left(\mathrm{sim}\left(P_{v_i}, P_{v_j}\right) > 0.9\right) - A_{T,i,j} - A_{N,i,j} \in \{0, 1\}. \quad (7)$$

Here, $\mathrm{sim}(\cdot)$ denotes cosine similarity. P is a vector of the medical location quotient (LQ), disability LQ, number of resident registration disabilities, and demand movements for each administrative district. Location quotient (LQ) measures the dispersion of a specific industry. We calculate the satisfaction of medical care and disability facilities in administrative districts by comparing them with Seoul city. It can be interpreted that the higher the coefficient, the higher the satisfaction of the owned facilities compared to other administrative districts, and vice versa—the lower the coefficient, the insufficient. LQ, a quantitative indicator, was used to compare the functional similarity between the two administrative districts. The adjacency matrix and normalized Laplacian matrix for the functional similarity between the two administrative districts were expressed in a heat map, as shown in Figure 3(c).

Chebyshev polynomials [25] were used to embed the configured adjacency matrix. We transformed the adjacency matrix into a Laplacian matrix as follows:

$$\tilde{L} = I - D^{-1/2} A D^{-1/2}, \quad (8)$$

where D is degree matrix, $\tilde{L}$ is normalized graph Laplcaian matrix, and I is identity matrix.

Using k-order Chebyshev polynomials [25],

$$f(A; \theta_i) = T_k\left(\tilde{L}\right) = 2x\, T_{k-1}\left(\tilde{L}\right) - T_{k-2}\left(\tilde{L}\right) \text{ with } T_0 = I, T_1 = \tilde{L}, \quad (9)$$

encoding.

3.4. Learning Temporal Dependency. Contextual gates and ConvLSTM deploy temporal dependencies. We use input values based on closeness, period, and trend. For closeness, we consider the demands from 1, 2, and 3 h in the past. The period is the same as that of 1, 2, and 3 d in the past. The trend is the demand a week in the past. As shown in Figure 4, contextual gating is performed.

We first compute GCN (X) applying GCN to the original value. In the model, GCN is applied as follows. Multigraph convolution is used, such as equation (10), to reflect spatial

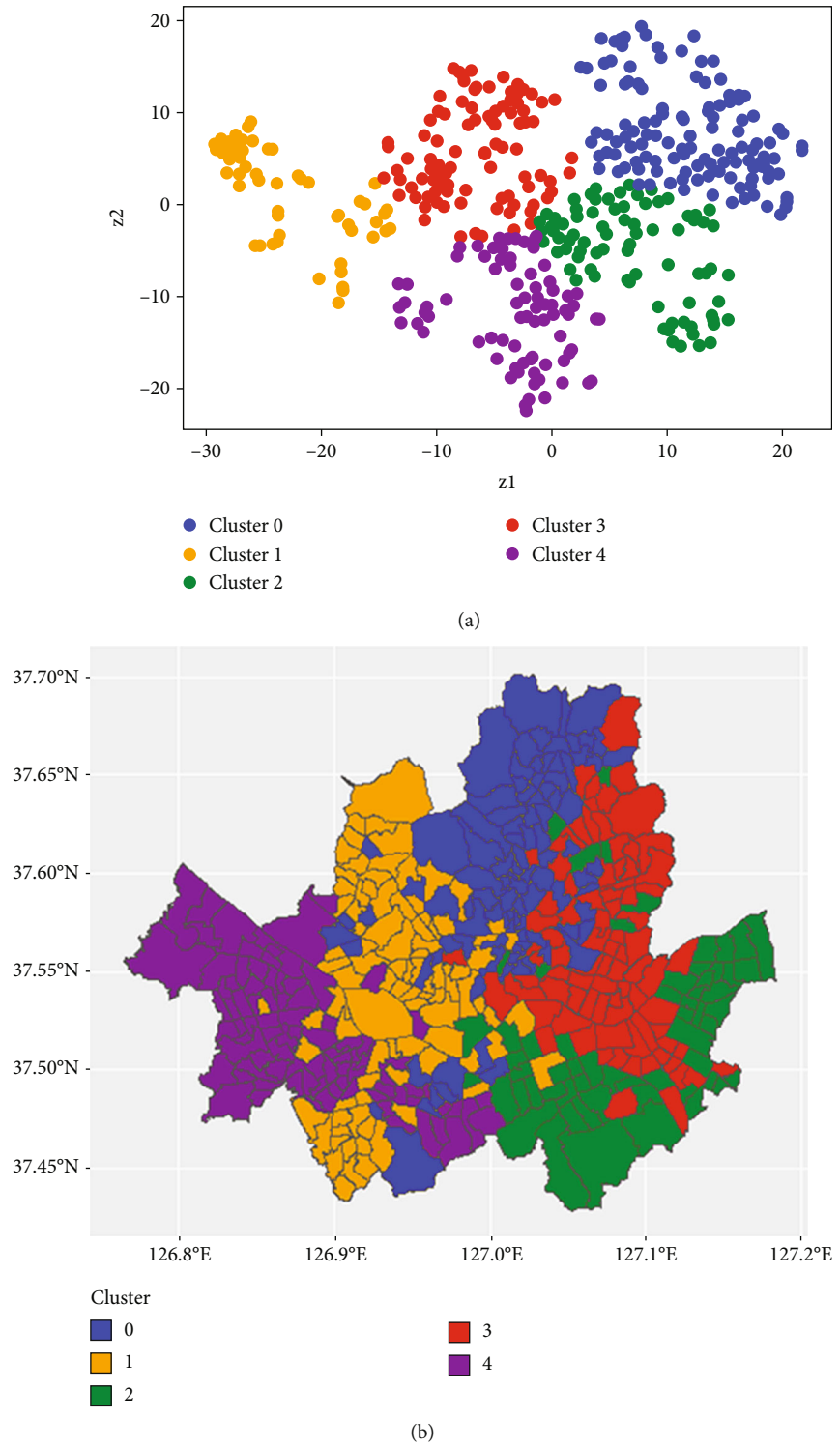


FIGURE 6: Results for the functional similarity adjacency matrix obtained using t-SNE: (a) t-SNE results for the functional similarity; (b) t-SNE results for the Seoul map.

dependency by utilizing several graphs configured. Multi-graph convolution is used to reflect spatial dependency. $X_l \in \mathbb{R}^{|V|\times P_l}$, $X_{l+1} \in \mathbb{R}^{|V|\times P_{l+1}}$ is feature vectors of region V layer in l and $l+1$.σ is activation function and $\sqcup$ is aggregation function, where is sum. $\mathbb{A}$ is a set of graphs, and $f(A;\theta_i)$ $\in \mathbb{R}^{|V|\times|V|}$ is the aggregation matrix of other samples. If $W_l \in \mathbb{R}^{P_l\times P_{l+1}}$ is the feature transformation matrix, X_{l+1} is updated to

$$X_{l+1} = \sigma\left(\sqcup_{A\in\mathbb{A}} f(A;\theta_i) X_l W_l\right). \tag{10}$$

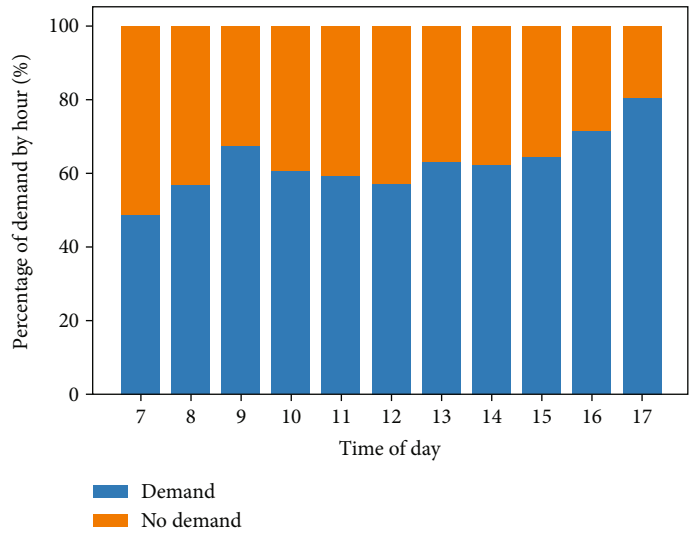


Figure 7: Percentage of demand occurrences by hour.

Table 1: Mean feature vectors by clusters.

Cluster	Medical LQ	Disabled LQ	Garage	Return home	Treatment	Rehabilitation	Religion	Commute	Shopping	Business
0	1.57	1.07	0.08	1259.99	616.59	416.56	78.55	148.14	12.31	0.33
1	1.46	0.96	0.11	1075.18	814.89	474.19	119.67	138.13	11.78	0.33
2	1.22	0.94	0.07	1218.28	553.75	454.73	90.22	89.83	10.15	0.23
3	1.53	1.00	0.09	1200.80	1200.80	717.32	59.14	141.73	21.75	0.55
4	1.61	1.08	0.10	1319.93	1319.93	405.94	68.83	97.03	12.47	0.49

Then, we apply global average pooling to all regions.

$$z^{(j)} = \frac{1}{|V|}\sum_{i=1}^{|V|}\left(F_{\text{pool}}\left(X_i^{(j)}\right) + F_{\text{pool}}\left(\text{GCN}\left(X_i^{(j)}\right)\right)\right), \text{for } j = 1, \cdots, J. \tag{11}$$

Let σ be a sigmoid function and δ be the GeLU, i.e., Ga linear unit function. Equation (11) produces the following summary:

$$\mathbf{s} = \left(s^{(1)}, s^{(2)} \cdots, s^{(J)}\right) = \sigma(\mathbf{W_2}\delta(\mathbf{W_1}\mathbf{z})), \tag{12}$$

for each of the temporary observation periods. We multiplied the calculated summary by the original value.

$$\tilde{X}^{(j)} = X^{(j)} \odot s^{(j)}, \text{for } j = 1, \cdots, J. \tag{13}$$

Through the contextual gating mechanism, we obtain reweighted observations with weights over time.

However, the LSTM architecture may not be well learned from sparse data. To resolve this, we applied ConvLSTM after the temporal mean, as shown in Figure 5. For each of the three inputs, the temporal mean is as follows:

$$\tilde{X}^{(j)} = \left(\text{mean}\left(\tilde{X}^{(\text{closeness})}\right), \text{mean}\left(\tilde{X}^{(\text{period})}\right), \text{mean}\left(\tilde{X}^{(\text{trend})}\right)\right). \tag{14}$$

We learn the temporal characteristics of each region using ConvLSTM in the temporal mean reweighted observations. Owing to intermittent demand, we convert sparse data into dense data. Therefore, we average features for closeness, period, and trend and then apply ConvLSTM. Across all regions, ConvLSTM is applied to the values of the reweighted observations. This results in a single vector that aggregates the learned spatiotemporal information.

$$H_i = \text{convlstm}\left(\tilde{X}^{(1)}, \cdots, \tilde{X}^{(j)}\right), \text{for } i = 1, \cdots, |V|. \tag{15}$$

Finally, a multigraph GCN is applied to the result of the ConvLSTM to learn spatiotemporal characteristics simultaneously. We then apply a fully connected layer for aggregation.

$$\hat{y}_i = \text{FC}((\text{GCN}(H_i)). \tag{16}$$

4. Spatiotemporal Characteristics of DRT Service

In the proposed method, the regional demand for DRT services is predicted via graph-based deep learning using the spatiotemporal characteristics of the demand in the past

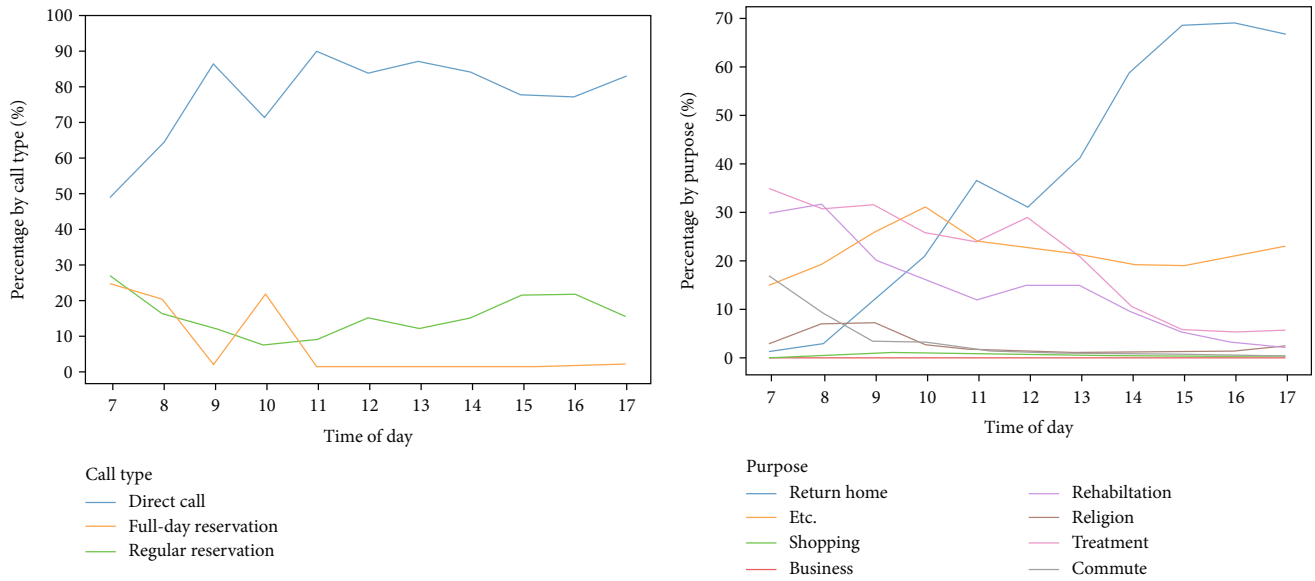


Figure 8: Percentage of demand for different (a) purposes of use and (b) call types based on hours.

two years. Therefore, it is necessary to investigate the cause of the presence or absence of demand. Accessibility to the DRT service is influenced mainly by time and space, as shown in Figures 6 and 7. In this section, the factors that affect the demand for transportation services are identified through temporal and spatial characteristic analyses.

4.1. Analysis of Spatial Characteristics. To visually validate the spatial dependency embedded vectors of the functional similarity adjacency matrix, we used t-distributed stochastic neighbor embedding (t-SNE) [26] over a low-dimensional space. Then, we applied k-means clustering [27] to the lower dimensions. We performed dimension reduction with t-SNE for visualization and observed five clusters, as shown in Figure 6. Table 1 presents the mean feature vectors for each group.

Group 0 shows the residential area with the most passengers boarding to commute. Meanwhile, there is a moderate demand for the rest of the purposes. In the case of group 1, the number of garages is relatively large, and it is a residential area where people board the most for returning home and religious purposes. In the case of group 2, the medical LQ and disability LQ are low, and they do not board well for business work and treatment purposes. In the case of group 3, many people used DRT service for returning home, rehabilitation, and shopping, and the pursuit of work was relatively high. Finally, in the case of group 4, the medical LQ and disability LQ are high, and the residential area tends to have the highest purpose of returning home.

4.2. Analysis of Temporal Characteristics. As shown in Figure 7, aggregating the demand status for the two years by the hour revealed that 7 a.m. was the most in demand and shows a decreasing trend at 8 a.m. and 9 a.m. However, it increases again from 10 a.m. and then to decrease to 20% from 1 p.m. to 5 p.m. Because of this characteristic, it is crucial to predict the demand in the period when the demand is plummeting, as most administrative districts exhibited a demand of >50% at 7 a.m. These results are attributed to the purpose of passenger use.

Table 2: Comparison of various methods.

Model	Accuracy	Precision	Recall	$F1$ score
HA	65.18	36.74	55.29	44.15
Logistic regression	68.04	30.92	66.07	42.13
XGboost [28]	69.82	42.47	65.19	51.43
Our method	78.13	75.41	62.26	68.21

Figure 8 shows the usage purpose pattern: the number of people returning home increased by 12 p.m., and the demand for treatment, rehabilitation, and commuting/work increased in the morning. In the case of movement for this purpose, because the movement is often constant, it is possible to predict the demand position using this pattern. A functionally similar adjacency matrix can explain this pattern.

According to the ratio of call types by time, direct calls and full-day reservations were inversely proportional in the case of full-day reservations. Therefore, we infer that disabled call taxis operate regularly. We make three policy suggestions. First, the demand should be checked on the previous date by expanding the operating time zone of the full-day reservation. Currently, the service is only operated at 7 a.m., 8 a.m., and 10 a.m. However, the demand should be predicted by expanding the operating hours or establishing a system that can be flexibly received the reservation before anytime. Second, movement should be encouraged by utilizing measures such as deploying additional temporal vehicles at 7 a.m., when the demand is the highest. Third, maximum movement should be achieved at the minimum cost by avoiding and adjusting the driver's rest time between 10 a.m. and 12 p.m., when the demand increases again.

Table 3: Effect of adding components to the spatial correlation modeling on the performance.

Component	Accuracy	Precision	Recall	$F1$ score
Neighborhood	77.89	76.50	59.65	67.03
Neighborhood+transportation	77.75	75.54	60.56	67.22
Neighborhood+transportation+functional	78.13	75.41	62.26	68.21

Table 4: Effects of temporal correlation modeling.

Temporal	Accuracy	Precision	Recall	$F1$ score
Average pooling	76.60	71.46	63.11	67.02
Max pooling	77.56	72.71	64.72	68.49
LSTM	77.45	73.40	62.99	67.80
ConvLSTM	78.13	75.41	62.26	68.21

Table 5: Effects of time combinations.

J	(# closeness, # period, # temporal)	Accuracy	Precision	Recall	$F1$ score
7	(3, 3, 1)	78.13	75.41	62.26	68.21
5	(2, 2, 1)	77.83	74.07	63.31	68.27
3	(1, 1, 1)	77.12	73.14	62.08	67.16
2	(0, 1, 1)	70.02	62.17	52.76	57.08
2	(1, 0, 1)	70.24	62.69	52.48	57.13
2	(1, 1, 0)	71.25	65.15	51.40	57.46

Table 6: Measures according to K.

K	Accuracy	Precision	Recall	$F1$ score
2	77.95	73.96	64.02	68.63
3	78.13	75.41	62.26	68.21
4	77.87	73.38	64.77	68.81

4.3. Model Performance Comparison. In this section, we compare the two aforementioned models. Let $\hat{y}_i = \Pr(\mathbf{X_i})$ be the conditional probability given an input x_i. For a loss of observation, we used the binary-cross entropy loss.

$$\mathscr{L}_{\mathrm{BCE}} = -\frac{1}{n}\sum_{i=1}^{n}[y_i \cdot \log(\hat{y}_i) + (1-y_i)\cdot \log(1-\hat{y}_i)]. \quad (17)$$

The training dataset included data from January 1, 2018, to October 31, 2019. Twenty percent of the data were used for the validation. Data from November 1, 2019, to December 31, 2019, were used as test data. To maintain chronological order, the data were not shuffled. ConvLSTM had four hidden sizes and three layers, and the GCN had 64 hidden sizes.

The performance of the proposed method was compared with that of other methods, and the results are presented in Table 2. Compared with the existing time series and classification model, the proposed method achieved significantly better performance. In contrast to the other methodologies, previous time zones, e.g., the closeness, period, and trend, were input as data configurations, and the characteristics of each administrative district (medical LQ, disability LQ, etc.) were added. Three adjacency matrices were used, and the results of the experiment are presented in Table 3. The first row presents the results obtained using only the neighborhood adjacency matrix. The second row presents the results obtained using two transportation adjacency matrices: the neighborhood and transportation adjacency matrices. The third row presents the results obtained using all three functional adjacency matrices, i.e., neighborhood, transportation, and functional similarity.

Table 7: Mean waiting time depending on whether there is a vacant vehicle that exists or not (min).

Time	Exist	Nonexist	Difference (nonexist-exist)
7	49.23	57.96	8.73
8	46.58	54.34	7.76
9	42.14	40.32	-1.82
10	26.95	48.24	21.29
11	28.81	44.96	16.15
12	30.16	42.07	11.91
13	32.91	35.63	2.72
14	33.77	34.78	1.01
15	31.74	29.38	-2.36
16	34.50	30.36	-4.14
17	28.17	34.55	6.38
Total	33.91	49.71	15.8

As shown, the method exhibited the best performance when all three adjacency matrices were used. However, in the case of the second row, the performance was inferior to that achieved using only the neighborhood adjacency matrix.

Table 4 presents the performance with respect to the type of temporal correlation. ConvLSTM outperformed vanilla LSTM, which did not reflect the spatial information. Also, max pooling shows lower performance.

The performance differences for different combinations of closeness, period, and trend are presented in Table 5. As time was used more, performance increased. In the case of call taxi data for the disabled, the demand is very intermittent, so the less time is used, the greater the sparse value will be affected. In addition, in the case of the demand a week ago, the actual past information is excessively required; therefore, the demand was fixed to 1. The performance difference when using the performance difference according to the use of K is presented in Table 6.

In the GCN, problems such as oversmoothing occur as the number of layers K increases excessively [29]. Similarly, in this study, when K increased by four or more, the

performance was degraded. Finally, in the case of Seoul, if it is influenced by too many hops, the performance is degraded, reflecting irrelevant administrative district relationships.

At the time of demand generation, we investigate the average difference time in waiting time between the case where the empty car is waiting and the case where there is no waiting. Table 7 shows the mean waiting time depending on whether there is a vacant vehicle that exists or not. When an empty car is on standby, we expect that we could reduce waiting time by about 16 minutes on average.

5. Conclusions

The proposed method can resolve unequal waiting times between regions by predicting the demand location for efficient operation of DRT services, which can support minimum cost–maximum movement. The objective of this study was to reduce the waiting time by efficiently rearranging nearby empty cars by predicting the regional demand for Seoul's call taxi service for the disabled, which has intermittent call characteristics. After configuring various subgraphs, the GCN was used to reflect the spatial characteristics between regions, and the model was constructed using the temporal mean and ConvLSTM to reflect temporal characteristics. Using various subgraphs from real data analysis showed alleviated results in terms of accuracy and interpretation. We expect improved convenience of movement and satisfaction with public transportation by reducing the waiting time. In addition, DRT services can replace public buses, increase the efficiency of subsidies for various types of public transportation, and generate profits and labor inducement effects for transportation companies, revitalizing the local economy and increasing the sharing rate of public transportation.

Data Availability

Data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

Acknowledgments

This study was supported by the Basic Study and Interdisciplinary R&D Foundation Fund of the University of Seoul (2021). The authors express their gratitude for this support.

References

[1] J.-H. Son, D.-G. Kim, E. Lee, and H. Choi, "Investigating the spatiotemporal imbalance of accessibility to demand responsive transit (drt) service for people with disabilities: explanatory case study in South Korea," *Journal of Advanced Transportation*, vol. 2022, Article ID 6806947, 9 pages, 2022.

[2] F. M. Coutinho, N. van Oort, Z. Christoforou, M. J. Alonso-González, O. Cats, and S. Hoogendoorn, "Impacts of replacing a fixed public transport line by a demand responsive transport system: Case study of a rural area in Amsterdam," *Research in Transportation Economics*, vol. 83, article 100910, 2020.

[3] K. M. Nahiduzzaman, T. Campisi, A. M. Shotorbani, K. Assi, K. Hewage, and R. Sadiq, "Influence of Socio-Cultural Attributes on Stigmatizing Public Transport in Saudi Arabia," *Sustainability*, vol. 13, no. 21, article 12075, 2021.

[4] S. Jain, N. Ronald, R. Thompson, and S. Winter, "Predicting susceptibility to use demand responsive transport using demographic and trip characteristics of the population," *Travel Behaviour and Society*, vol. 6, pp. 44–56, 2017.

[5] F. Toqué, M. Khouadjia, E. Come, M. Trepanier, and L. Oukhellou, "Short & long term forecasting of multimodal transport passenger flows with machine learning methods," in *In2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 560–566, Yokohama, Japan, 2017.

[6] Z. Lin, J. Feng, Z. Lu, Y. Li, and D. Jin, "Deepstn+: context-aware spatial- temporal neural network for crowd flow prediction in metropolis," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 1020–1027, 2019.

[7] D. Lee, S. Jung, Y. Cheon, D. Kim, and S. You, "Demand forecast- ing from spatiotemporal data with graph networks and temporal-guided embedding," 2019, http://arxiv.org/abs/1905.10709.

[8] E. Chandakas, "On demand forecasting of demand-responsive paratransit services with prior reservations," *Transportation Research Part C: Emerging Technologies*, vol. 120, article 102817, 2020.

[9] J. Choi, J. Song, M. Kang, and K. Hwang, *A clustering analysis on the machine learning-based bus routes types and bus stations for adoption of demand responsive transport*, Korean Society Of Transportation, 2021.

[10] J. Wang, T. Yamamoto, and K. Liu, "Spatial dependence and spillover effects in customized bus demand: empirical evidence using spatial dynamic panel models," *Transport Policy*, vol. 105, pp. 166–180, 2021.

[11] I. Peled, K. Lee, J. Yu, J. Dauwels, and F. C. Pereira, "On the quality requirements of demand prediction for dynamic public transport," *Communications in Trans- portation Research*, vol. 1, article 100008, 2021.

[12] Z. Chen, K. Liu, J. Wang, and T. Yamamoto, "H-convlstm-based bagging learning approach for ride-hailing demand prediction considering imbalance problems and sparse uncertainty," *Transportation Research Part C: Emerging Technologies*, vol. 140, p. 103709, 2022.

[13] Z. Chen, K. Liu, and T. Feng, "Examine the prediction error of ride-hailing travel demands with various ignored sparse demand effects," *Journal of Advanced Transportation*, vol. 2022, Article ID 7690309, 11 pages, 2022.

[14] G. Han, S. Ha, J. J. Hong, and C. Lee, *A study on the application of demand forecasting for call taxi for the disabled in Seoul using deep learning*, Korean Society Of Transportation, 2017.

[15] D. Hong, G. Han, S. Ha, and C. Lee, *Optimum hyperparameter selection of lstm model for call taxi waiting time for persons with disabilities in Korea*, Korean Society Of Transportation, 2018.

[16] J. Zhang, Z. Yu, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," 2016, http://arxiv.org/abs/1610.00081.

[17] G. Xu, Y. Li, L. Wang et al., "Spa- tiotemporal multi-graph convolution network for ride-hailing demand forecasting," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 3656–3663, 2019.

[18] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," 2017, http://arxiv.org/abs/1709.01507.

[19] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, and T.-S. Chua, "SCA-CNN: spatial and channel-wise attention in convolutional networks for image captioning," 2016, http://arxiv.org/abs/1611.05594.

[20] L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Q. Z. Sheng, "Stg2seq: spatial- temporal graph to sequence model for multi-step passenger demand forecasting," 2019, http://arxiv.org/abs/1905.10069.

[21] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional neural net- work: a deep learning framework for traffic forecasting," 2017, http://arxiv.org/abs/1709.04875.

[22] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, 2011.

[23] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, http://arxiv.org/abs/1609.02907.

[24] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional LSTM network: a machine learning approach for precipitation nowcasting," 2015, http://arxiv.org/abs/1506.04214.

[25] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," *Advances In Neural Information Processing Systems*, vol. 29, 2016.

[26] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.

[27] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[28] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," *InProceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, vol. 13, pp. 785–794, 2016.

[29] U. Alon and E. Yahav, "On the bottleneck of graph neural networks and its practical implications," 2020, http://arxiv.org/abs/2006.05205.

Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 2806183, 17 pages
https://doi.org/10.1155/2022/2806183

Research Article

ST-AGRNN: A Spatio-Temporal Attention-Gated Recurrent Neural Network for Traffic State Forecasting

Jian Yang[1,2] **Jinhong Li**[1,2] **Lu Wei**[1] **Lei Gao**[1,2] **and Fuqi Mao**[2]

[1]*Beijing Key Lab of Urban Road Traffic Intelligent Technology, North China University of Technology, Beijing 100144, China*
[2]*School of Computer Science and Technology, North China University of Technology, Beijing 100144, China*

Correspondence should be addressed to Jian Yang; yanj200045@163.com

Received 2 June 2022; Revised 26 July 2022; Accepted 13 September 2022; Published 3 October 2022

Academic Editor: Yanming Shen

Accurate traffic state prediction plays an important role in traffic guidance, travel planning, etc. Due to the existence of complex spatio-temporal relationships, there are some challenges in forecasting. Firstly, in terms of spatial correlation, some models only consider the road network structure information, and ignore the relative location relationships between nodes. Secondly, some models ignore the different impacts of nodes in the global road network on traffic. To solve these problems, we propose a new traffic state-forecasting model, namely, spatio-temporal attention-gated recurrent neural network (ST-AGRNN). In the proposed model, structure-based and location-based localized spatial features are obtained simultaneously by Graph Convolutional Networks (GCNs) and DeepWalk. The localized temporal features are obtained by gated recurrent unit (GRU). The attention-based approach is used to obtain global spatio-temporal features. Experimental validation is performed with two real-world public datasets, and the results show that the ST-AGRNN model outperforms the state-of-the-art methods.

1. Introduction

Traffic congestion is a common problem faced by almost all major cities. Because of traffic congestion, a lot of manpower and material resources are wasted every year. Accurate and real-time traffic state prediction is the basis to solve the problem of traffic congestion. On the one hand, people can plan their trips in advance through traffic-state information. On the other hand, traffic managers also conduct effective traffic guidance and management through traffic state prediction information. At the same time, traffic prediction is a typical spatio-temporal problem, and the inherent nonlinearity and complexity of traffic affect the accuracy of prediction. Therefore, integrated consideration of temporal and spatial characteristics is necessary for traffic state prediction.

Taking the spatio-temporal correlation in Figure 1 as an example, there are localized spatio-temporal correlations and global spatio-temporal correlations. Each node will have influence on the traffic of its neighbors because it is physically connected with its neighbors and belongs to the relationship between upstream and downstream, which is spatial dependence. At the same time, each node will also affect itself at the next time step, which is temporal dependence. These are localized spatio-temporal correlations. In addition, a busy intersection has influence on the traffic of the entire region, which is the global spatio-temporal correlation in the road network. Obtaining this correlation is crucial to spatio-temporal data prediction.

In previous studies, various deep learning approaches were used to model spatio-temporal correlations, including stacked autoencoders (SAEs) [1], recurrent neural networks (RNNs) [2], generative adversarial networks (GANs) [3], transformer [4, 5], convolutional neural networks (CNNs) [6], and Spatio-Temporal Graph Convolutional Networks (STGCN) [7]. The SAEs acquire spatial and temporal correlations through unsupervised learning. The RNNs extract temporal features through the gate mechanism. The GANs extract spatio-temporal features through generators and discriminators and the transformer model spatial and temporal dependencies through encoder–decoder architecture. The

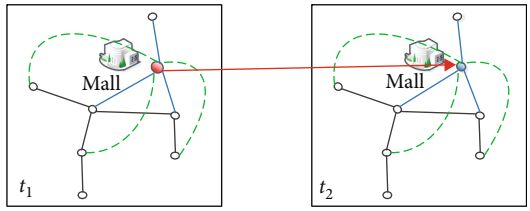


Figure 1: The influence of nodes in spatio-temporal correlations networks. The solid blue lines represent node spatio correlations. The red arrow represents the node temporal correlations. The green dash lines represent the global spatio-temporal correlations.

CNNs and GCNs obtain spatial features through convolution operation. However, these methods only capture localized spatio-temporal correlations.

Recently, attention mechanisms have received increasing attention. Because they are effective in identifying the relevance of inputs in prediction, components with high relevance are given greater attention. They are successfully applied in many fields, such as natural language processing (NLP) [8], computer vision (CV) [9, 10], and speech recognition [11]. Attention-based traffic forecasting has also developed rapidly in recent years. For example, attention temporal graph convolutional network (A3T-GCN) [12] uses attention mechanism to obtain global temporal and spatial correlations. However, it ignores location-based localized spatial information.

To obtain complex localized and global spatio-temporal correlations, we propose a novel deep learning architecture—spatio-temporal attention-gated recurrent neural network (ST-AGRNN)—for traffic state prediction. To fully exploit the localized spatio-temporal correlations, ST-AGRNN learns structure-aware graph embedding information through a GCN, and obtains position-aware information through DeepWalk. To tackle temporal dependencies, a gated recurrent unit (GRU) is used. Finally, in order to fully exploit the global spatio-temporal correlations, the attention mechanism is used to obtain spatio-temporal correlations about the networks.

The main contributions of this work are as follows:

(i) we propose a new localized spatial feature extraction method by combining DeepWalk with a GCN, where DeepWalk obtains position-aware information and the GCN obtains structure-aware graph embedding information

(ii) Traffic state is a time series data. The current traffic state will affect the traffic state at the next time step. GRU is used to obtain localized temporal correlation between traffic data

(iii) Attention mechanisms are introduced to obtain global spatio-temporal correlations about networks. Different nodes have different impacts on the traffic state, and the attention mechanism can obtain the weight of nodes from the historical traffic state, representing the global spatio-temporal correlations of network

(iv) Our experiments applying ST-AGRNN to traffic state prediction show that ST-AGRNN outperforms 12 state-of-the-art methods in terms of both accuracy and robustness on two benchmark datasets

2. Literature Review

2.1. Traffic State Forecasting. Time series data modeling and prediction are widely used in many fields [13, 14]. Traffic state data is a typical time series data. There are two main categories in traffic forecasting: statistical methods and machine learning methods. Statistical methods include autoregressive integrated moving average (ARIMA), the Kalman filter (KF), Markov chains, exponential smoothing (ES), and Bayesian networks. In the 1970s, Ahmed and Cook [15] used ARIMA to predict short-term traffic flow. Hamed et al. [16] later applied a simple ARIMA model to predict traffic volumes in urban arterials. Subsequently, various variants of ARIMA have emerged [17–19]. Kalman filtering excels in regression problems. Guo et al. [20] applied an adaptive Kalman filtering model to predict short-term traffic flow. Hinsbergen et al. [21] used a localized extended Kalman filter (L-EKF) to estimate traffic states. In addition, traffic prediction methods based on Markov chains, exponential smoothing (ES), and Bayesian networks also perform well. For example, Qi et al. [22] proposed a hidden Markov model (HMM) to achieve short-term freeway traffic prediction during peak periods. Chan et al. [23] employed the hybrid exponential smoothing method and the Levenberg–Marquardt (LM) algorithm for short-term traffic flow forecasting. Wang et al. [24] used an improved Bayesian combination method (BCM) for short-term traffic flow prediction.

Statistical methods have some disadvantages, such as the inability to deal with nonlinear relationships between data. Machine learning methods, on the other hand, are more flexible. Machine learning methods are mainly divided into classical machine learning and deep learning.

Commonly used classical machine learning approaches include k-nearest neighbors (KNN), support-vector machine (SVM), random forest (RF), and decision tree (DT) methods. Cai et al. [25] proposed an improved KNN model to achieve short-term traffic multistep forecasting. Xu et al. [26] used kernel k-nearest neighbors (kernel-KNN) to predict road traffic states in time series. Cong et al. [27] presented a traffic flow prediction model based on the least squares support-vector machine, and automatically determined the least squares support-vector machine model with two parameters at the appropriate value by FOA. Xu et al. [28] used genetic programming (GP) and random forest (RF) techniques to achieve real-time crash prediction on freeways. Crosby et al. [29] proposed a spatially intensive decision tree for the prediction of traffic flow across the entire UK road network. Although classical machine learning methods are effective in identifying nonlinear relationships in traffic states, they still have many drawbacks, e.g., KNN models have low prediction accuracy for rare

categories and require high computational complexity when there are many features. It is difficult to choose a suitable kernel function by applying the SVM model. The random forests do not perform very well on high-dimensional sparse data. In addition, decision trees are prone to overfitting.

In order to solve the above problems, deep learning has been developed rapidly in recent years. The key to traffic prediction is to learn the temporal dependence and spatial dependence, where the methods to learn the temporal dependence are mainly recurrent neural networks (RNNs) and their variants long short-term memory (LSTM) and gated recurrent units (GRUs). Nejadettehad et al. [30] used three kinds of recurrent neural networks to predict short-term traffic flow. Van et al. [31] used recurrent neural networks to predict freeway travel time. Tian et al. [32] took advantage of LSTM to dynamically determine the optimal time lags to predict short-term traffic flow. Fu et al. [33] used LSTM and GRU methods to predict short-term traffic flow. These models consider the temporal dependence but ignore the spatial dependence in the road network. Therefore, they cannot accurately predict changes in the traffic state. Obtaining the temporal and spatial dependence is a prerequisite for accurate traffic prediction. There are also many models for the learning of spatial features. For example, Lv et al. [34] proposed a stacked autoencoder model to inherently learn the spatial and temporal correlations for traffic flow prediction. Yuan et al. [35] proposed a novel variable-wise weighted stacked autoencoder (VW-SAE) for hierarchical, layer-by-layer output-related feature representation. Ma et al. [36] proposed a convolutional neural network (CNN)-based model to learn traffic as images and predict large-scale, network-wide traffic speed. Wu et al. [37] proposed a model called CLTFP, which combines CNN and LSTM, to forecast future traffic flow. Jo et al. [38] adopted a convolutional neural network (CNN) to deal with map images representing traffic states and the model adopts images for both the input and the output of a CNN model to predict traffic speeds.

Although the above methods can handle spatial dependencies in traffic, CNNs are more suitable for Euclidean spatial structures such as pictures, and grids. Meanwhile, traffic road networks are complex networks, and the neighboring nodes are not fixed. Thus, the spatial features of the road network cannot be fully obtained by CNNs. In recent years, graph-based convolution operations have developed rapidly [39], and have become suitable for learning the structural features of graph types. He et al. [40] used LDA and GCN to tackle road link speed prediction. Li et al. [41] proposed a DCRNN model for obtaining spatio-temporal dependence in traffic flow forecasting; the model uses diffusion convolution to learn spatial dependence and a GRU to learn temporal dependence. Wu et al. [42] learned an adaptive dependency matrix via node embedding to obtain spatial dependency and temporal dependency through stacked dilated 1D convolution. Huang et al. [43] proposed a new graph attention network, cosAtt, to obtain spatial features through cosAtt and GCN and temporal features through a GLU. Roy et al. [44] consider important daily patterns and present-day patterns from traffic data in addition to spatio-temporal characteristics to improve the accuracy of predictions. However, these methods only consider the spatial features based on structure-aware graph embedding information, without considering the location information, so they cannot effectively obtain the spatial features.

2.2. Attention Mechanism. The attention mechanism has been a hot topic of neural network research in recent years, and it has been remarkable in neural machine translation, image captioning, time series prediction etc. The attention mechanism originates from the study of human vision, which determines which part of the input needs to be attended to and allocates processing resources to the important parts. Bahdanau et al. [45] proposed the use of an attention mechanism in the decoder to decide which part of the input sentence should be attended to. Xu et al. [46] introduced the application of soft and hard attention mechanisms to image captioning. Li et al. [47] proposed convolutional self-attention further improves Transformer' performance to achieve time series forecasting. Daiya et al. [48] proposed a multimodal deep learning architecture for stock movement prediction. Zhou et al. [49] used ProbSparse self-attention mechanism and distilling operation to handle quadratic time complexity and memory usage. In the area of traffic state prediction, prediction methods based on attention mechanisms are also developing rapidly. Park et al. [50] proposed the use of temporal attention, spatial attention and spatial sentinel vectors to obtain temporal and spatial dependencies. Wang et al. [51] proposed a novel spatial temporal graph neural network model for traffic flow prediction, and a learnable positional attention mechanism is applied in the model to aggregate information from adjacent roads. Guo et al. [52] proposed a novel attention-based spatio-temporal graph convolutional network (ASTGCN) to model recent, daily, and weekly dependencies.

Inspired by the above study, considering traffic location information and spatio-temporal characteristics, we learned both location- and structure-based information to obtain localized spatial features, learned localized temporal features through a GRU and, finally, considered the global spatio-temporal features of traffic networks through the attention mechanism.

3. Methodology

3.1. Data Processing. Given a speed sequence of data $T_0, T_1, T_2, \cdots, T_n$ with a length of n, the time interval is 5 minutes. To predict the future 15 minutes of data, for example, the input sample construction process of the model is shown in Figure 2. The input data of sample 1 is $\{T_0, T_1, T_2, \cdots, T_{11}\}$, and the label data is $\{T_{12}, T_{13}, T_{14}\}$. The input data of sample 2 is $\{T_1, T_2, T_3, \cdots, T_{12}\}$, and the label data is $\{T_{13}, T_{14}, T_{15}\}$. And so on, to obtain the entire input sample matrix. If predicting the next 30 minutes of data, the method is similar, i.e., the input data of sample 1 is unchanged, the label data is $\{T_{12}, T_{13}, T_{14}, T_{15}, T_{16}, T_{17}\}$, and the sample matrix is obtained recursively. The longer the prediction time, the more the label is increase.

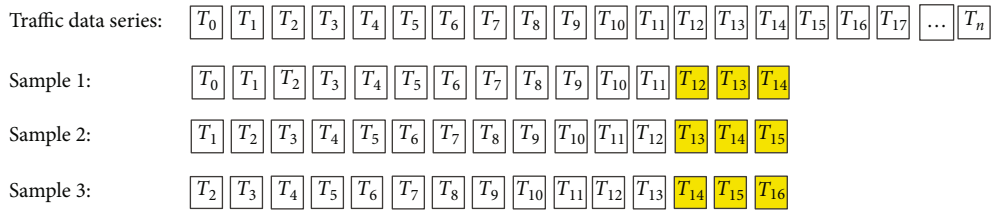


Figure 2: Sample construction.

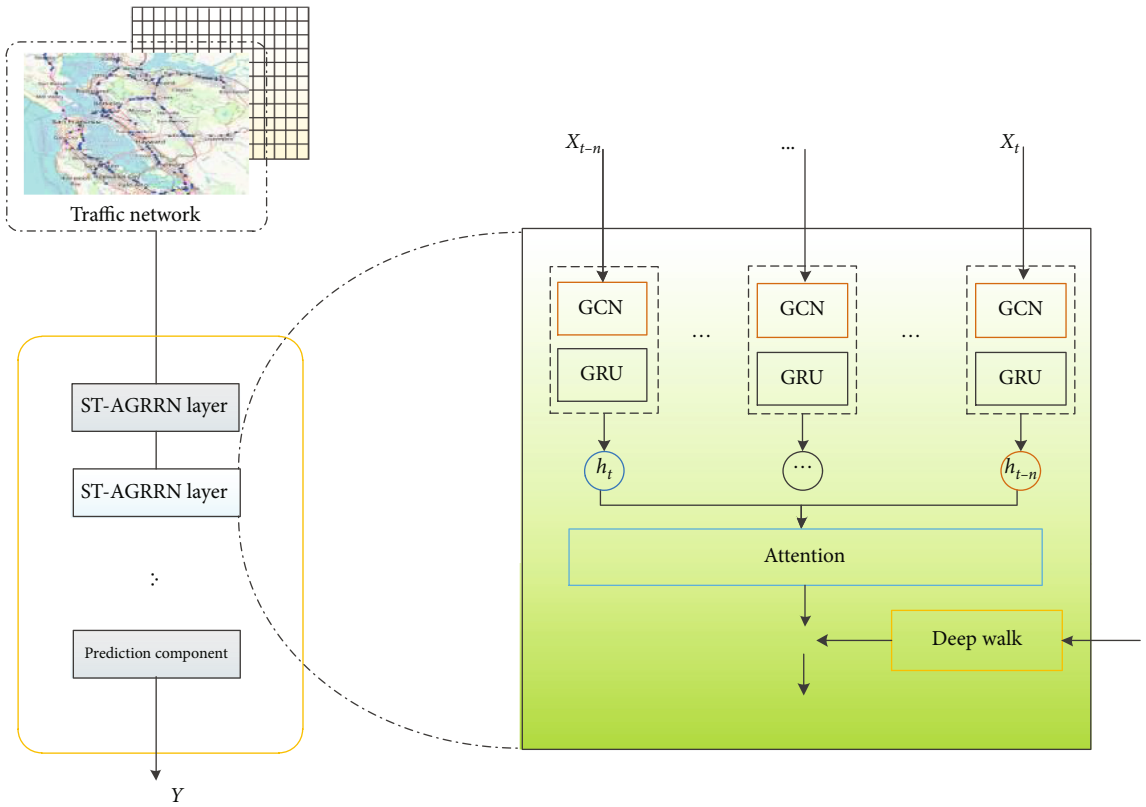


Figure 3: The architecture of the ST-AGRNN.

3.2. Traffic State Prediction Based on ST-AGRNN. The structure of the ST-AGRNN model is shown in Figure 3. In order to fully capture the localized spatial dependencies, we propose a new spatial feature extraction method by combining DeepWalk with a GCN, where DeepWalk obtains position-aware information and the GCN obtains structure-aware graph embedding information. The localized temporal dependencies are captured using the gated recurrent unit network, and the road network global spatio-temporal dependencies are captured using the attention mechanism. The specific details of each part of the model are presented in the next subsections.

3.2.1. Localized Spatial Dependency. Consider the urban road network as an undirected graph $G = (V, E)$, where V is the set of vertices in the graph and E is the set of edges. Denote the adjacency matrix of the graph by W. $D = \text{diag}(d_1, \cdots, d_n)$ denotes the degree matrix of the graph, where $d_i = \sum_{j=1}^{N} W_{ij}$ denotes the number of adjacencies of each vertex. Moreover, the Laplace matrix of the graph is expressed as $L = I_N - D^{-1/2}AD^{-1/2} = U\Lambda U^T$ (where U is an orthogonal matrix composed of eigenvectors), and the Fourier transform and inverse transform of the graph can be expressed as $\hat{x} = U^T x$ and $x = U\hat{x}$, respectively. A two-layer graph convolutional neural network can be represented as follows:

$$Z = f(X, A) = soft\max\left(\hat{A}\,\text{Re}\,LU\left(\hat{A}XW^{(0)}\right)W^{(1)}\right), \quad (1)$$

where X denotes the feature of the node, while A denotes the adjacency matrix of the graph. Calculated in the preprocessing step $\hat{A} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$, where $\tilde{A} = A + I_N$ denotes the adjacency matrix with self-connections, $\widetilde{D_{ii}} = \sum_j A_{ij}$, $W^{(0)}$ is the weight of the input layer to the hidden layer, while $W^{(1)}$ is the weight of the hidden layer to the output layer.

The GCN aggregates information about neighboring nodes via convolution, which is a structure-based graph

Input: The training epoch N; the historical traffic state x_t; the traffic graph $G = (V, E)$; the window size of historical traffic state p; the predicted length of traffic state q;
Output: Learned ST-AGRNN model
1: Initialization parameter θ;
2: Data processing;
3: For $\forall i \in N$ do
4: Select real historical data $x_{t-p}, \cdots, x_t$;
5: Select real future data $x_{t+1}, \cdots, x_{t+q}$;
6: Input real historical data $x_{t-p}, \cdots, x_t$ and the traffic graph $G = (V, E)$ into GCN and GRU to get h_i;
7: Input h_i into attention to get c_i;
8: Use DeepWalk on G and get the embedding result $\tilde{s}$;
9: Concatenate c_i and $\tilde{s}$, $o_t^i = c_i \oplus \tilde{s}$;
10: Optimize θ by minimizing the loss function;
11: End for

Algorithm 1: Training of ST-AGRNN.

Table 1: Traffic speed prediction performance under different benchmark methods in the PeMSD4 and PeMSD8 datasets (bold is the best; underline is the second best.).

Model	PeMSD4 (MAE/RMSE/MAPE(%))		
	15 min	30 min	60 min
HA	2.54/4.96/5.56	2.54/4.96/5.56	2.54/4.96/5.56
ARIMA(2003)	2.51/5.72/5.32	2.75/6.34/5.69	3.21/7.36/6.56
DCRNN(2018)	1.35/2.94/2.68	1.77/4.06/3.71	2.26/5.28/5.10
STGCN(2018)	1.47/3.01/2.92	1.93/4.21/3.98	2.55/5.65/5.39
ASTGCN(2019)	2.12/3.96/4.16	2.42/4.59/4.80	2.73/5.21/5.46
GWN(2019)	1.30/2.68/2.67	1.70/3.82/3.73	2.03/4.65/4.60
LSGCN(2020)	1.45/2.93/2.90	1.82/3.92/3.84	2.22/4.83/4.85
USTGCN(2021)	1.40/2.69/2.81/	1.64/3.19/3.23	2.03/4.25/4.32
ST-AGRNN	**1.19/2.36/2.17**	**1.45/2.98/2.69**	**1.76/3.63/3.24**
Model	PeMSD8 (MAE/RMSE/MAPE(%))		
	15 min	30 min	60 min
HA	1.98/4.11/3.94	1.98/4.11/3.94	1.98/4.11/3.94
ARIMA(2003)	1.90/4.87/5.11	2.12/5.24/5.21	2.79/6.22/5.62
DCRNN(2018)	1.17/2.59/2.32	1.49/3.56/3.21	1.87/4.50/4.28
STGCN(2018)	1.19/2.62/2.34	1.59/3.61/3.24	2.25/4.68/4.54
ASTGCN(2019)	1.49/3.18/3.16	1.67/3.69/3.59	1.89/4.13/4.22
LSGCN(2020)	1.16/2.45/2.24	1.46/3.28/3.02	1.81/4.11/3.89
USTGCN(2021)	1.14/2.15/2.07	1.25/**2.58**/2.35	1.70/**3.27**/3.22
ST-AGRNN	**1.015/2.07/1.82**	**1.24**/2.63/**2.21**	**1.53**/3.33/**2.71**

embedding algorithm. The obtained embedding representation cannot retain the position relationship between nodes, which is a very important relationship between nodes in the traffic network. Deepwalk's objective function forces nodes that are close in the shortest path to be close in the embedding space representation [53]. In order to fully exploit the spatial features of the road network, we introduce the DeepWalk algorithm to learn the position embedding representation between nodes.

The graph embedding algorithm based on the random walk is also close in the embedding space for nodes that are close in the shortest path. This allows the resulting embedding space to also preserve the relative positional relationships. These relations are an important complement to the structure-based embedding space, and are necessary for spatial features in traffic.

The random walk with v_t as the vertex is represented as $\{W_{v_t}^1, W_{v_t}^2, \cdots, W_{v_t}^k\}$, where $W_{v_t}^k$ denotes the k th node in the path with v_t as the root. For all of the nodes in the graph, each node has another similar path. We then obtain a sequence matrix W. The corresponding graph embedding representation containing the location information is then obtained by the

Table 2: Traffic flow prediction performance under different benchmark methods in the PeMSD4 and PeMSD8 datasets (bold is the best; underline is the second best.).

Model	PeMSD4		
	MAE	RMSE	MAPE (%)
HA	38.03	59.24	27.88
ARIMA(2003)	33.73	48.80	24.18
STGCN(2018)	21.16	34.89	13.83
DCRNN(2018)	21.22	33.44	14.17
ASTGCN(r)(2019)	22.93	35.22	16.56
GWN(2019)	24.89	39.66	17.29
LSGCN(2020)	21.53	33.86	13.18
STSGCN(2020)	21.19	33.65	13.90
STFGNN(2021)	20.48	32.51	16.77
Z-GCNETs(2021)	19.50	31.61	12.78
STG-NCDE(2022)	19.21	31.09	**12.76**
ST-AGRNN(ours)	**18.97**	**30.003**	12.81
Model	PeMSD8		
	MAE	RMSE	MAPE (%)
HA	34.86	59.24	27.88
ARIMA(2003)	31.09	44.32	22.73
STGCN(2018)	17.50	27.09	11.29
DCRNN(2018)	16.82	26.36	10.92
ASTGCN(r) (2019)	18.25	28.06	11.64
GWN(2019)	18.28	30.05	12.15
LSGCN(2020)	17.73	26.76	11.20
STSGCN(2020)	17.13	26.80	10.96
STFGNN(2021)	16.94	26.25	10.60
Z-GCNETs(2021)	15.75	25.11	10.01
STG-NCDE(2022)	15.45	24.81	9.92
ST-AGRNN(ours)	**14.95**	**23.15**	**9.21**

update procedure—the skip-gram algorithm. The embedding representation is denoted as s, and then the final result is obtained by the fully connected layer.

$$\tilde{s} = f(W \cdot s + b), \tag{2}$$

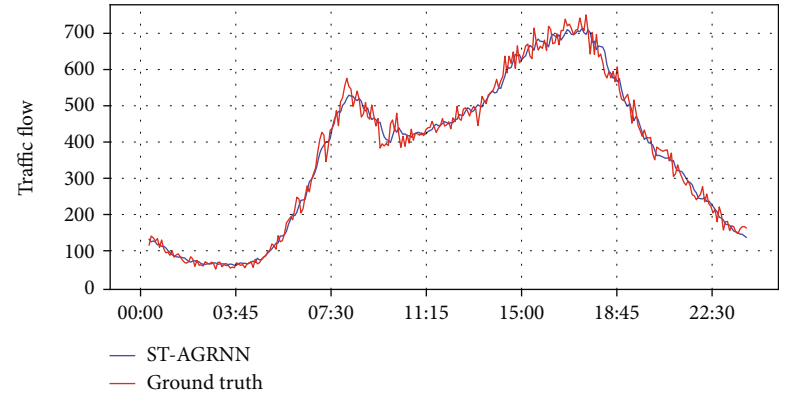
where $\tilde{s}$ denotes the graph embedding representation, while W and b are the learnable weights and biases, respectively.

3.2.2. Localized Temporal Feature. Temporal dependence is another major problem in traffic prediction. Recurrent neural network (RNN) models are very effective for time-series data processing, but they suffer from gradient disappearance and gradient explosion. GRUs and LSTM are variants of RNN that can effectively overcome these problems.

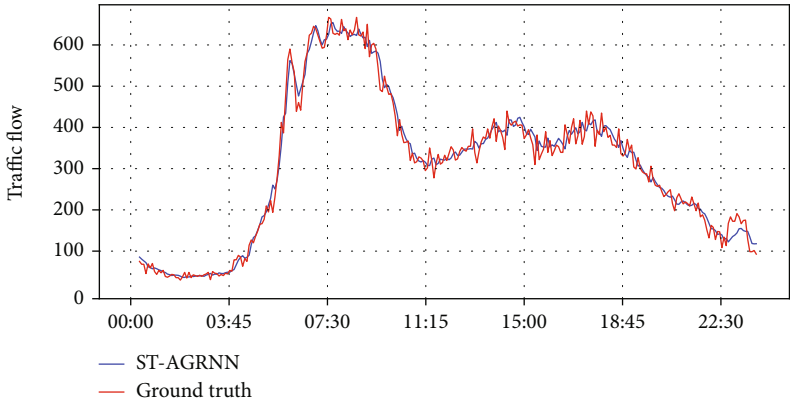
GRU is used to handle temporal dependence. s_t is the output of GCN at time t, x_t is the traffic state at the present moment, and r_t is the reset gate that determines whether the previous moment information is retained or not—if it is 1, then the message is carried to the next moment; if it is 0, then the message is ignored. h_{t-1} is the hidden state at the previous moment. z_t is the update gate, which is a value between 0 and 1 that determines how much information is remembered from the previous moment—if it is 1, then more information is remembered; if it is 0, then more is forgotten. $\widetilde{h}_t$ is the current memory content, and h_t is the output of the current moment.

$$\begin{aligned} s_t &= \mathrm{GCN}(x_t), \\ r_t &= \sigma(W_r \cdot [h_{t-1}, s_t \cdot x_t] + b_r), \\ z_t &= \sigma(W_z \cdot [h_{t-1}, s_t \cdot x_t] + b_z), \\ \widetilde{h}_t &= \tanh\left(W_h \cdot [r_t \odot h_{t-1}, s_t \cdot x_t] + b_h\right), \\ h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \widetilde{h}_t. \end{aligned} \tag{3}$$

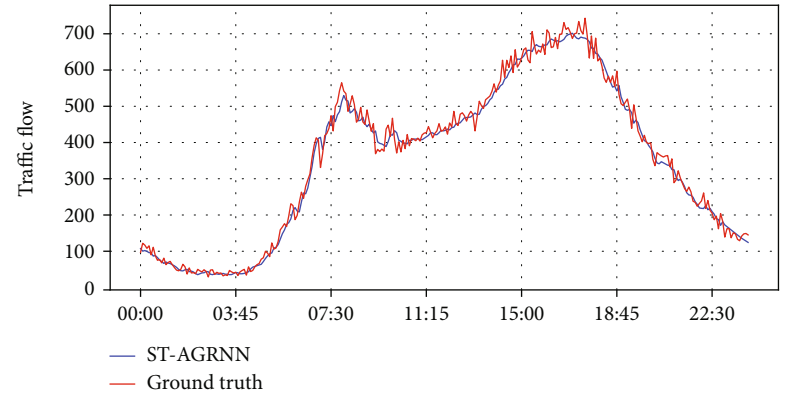
3.2.3. Global Spatio-Temporal Correlations. Critical intersections in cities often have a large impact on regional traffic, and congestion at critical intersections is likely to evolve into congestion in the associated areas. In order to strengthen the modeling ability of traffic networks, this paper obtains global spatio-temporal correlations through the attention mechanism. All of the hidden states of the GRU network are used as the input of the attention network, and then the weights of each hidden state of the GRU are calculated to obtain



(a) Node 111 in PeMSD4 for 15 min

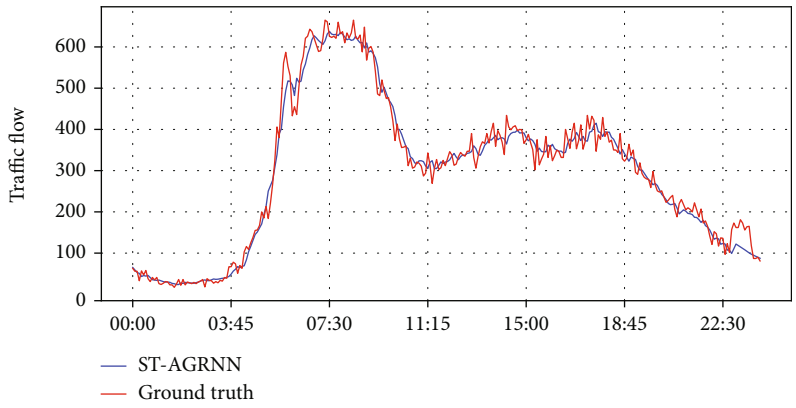


(b) Node 261 in PeMSD4 for 15 min



(c) Node 111 in PeMSD4 for 60 min

Figure 4: Continued.



(d) Node 261 in PeMSD4 for 60 min

FIGURE 4: Traffic flow forecast visualization in PeMSD4.

the traffic information changes in the road network at each moment. The attention network is calculated as follows:

$$e_i = W^{(1)}\left(\tanh\left(W^{(0)}h_i + b^{(0)}\right)\right) + b^{(1)},$$
$$a_i = \frac{\exp(e_i)}{\sum_{k=1}^{n}\exp(e_k)}, \tag{4}$$
$$c_i = \sum_{i=1}^{n} a_i * h_i,$$

where e_i is the attention coefficient, h_i is the GRU hidden state, $W^{(0)}$ and $W^{(1)}$ are the trainable weight parameters, $b^{(0)}$ and $b^{(1)}$ are the trainable bias values, a_i is the normalized attention coefficient, and c_i is the attention weight.

3.3. Prediction Component. We predict future changes in traffic state based on historical traffic states. In the prediction component, we concatenate the attention mechanism and the location-based graph embedding output as follows:

$$o_t^i = c_i \oplus \tilde{s}. \tag{5}$$

The concatenation result is used as the input of the fully connected layer, and the final traffic state is obtained by the sigmoid activation function. It is expressed as y_{t+T}^i, where T is the predicted time step, in the following form:

$$y_{t+T}^i = \text{sigmoid}\left(W_s o_t^i + b_s\right), \tag{6}$$

where W_s and b_s are the learnable weights and biases, respectively.

The training overview of the model is shown in Algorithm 1. We used Adam to optimize the model. We used TensorFlow to implement the proposed model.

4. Experiments

4.1. Experimental Settings. The software and hardware environments for the experiments were configured as follows: PYTHON 3.6.2, NUMPY 1.16.0, TENSORFLOW 1.14.0, and Memory: 64 GB.

For this paper, we used speed and traffic flow to represent traffic states, where 80% of the data were used as the training set and 20% as the test set. In the experiments, the speed was predicted for 15, 30, and 60 minutes, and the flow prediction was predicted from 5 to 60 minutes with 12 time windows.

We use the root mean square error (RMSE), mean absolute error (MAE), and mean absolute percentage errors (MAPE) to evaluate the models.

4.2. Dataset Description. In the experiment, we used two real-world traffic datasets: PeMSD4, and PeMSD8 [43].

PeMSD4 was collected from the Caltrans Performance Measurement System (PeMS) and the traffic data in the San Francisco Bay Area, with 307 sensors on 29 roads. The dataset spanned from January to February 2018.

PeMSD8 refers to the traffic data in San Bernardino from July to August 2016, with 170 detectors on 8 roads.

4.3. Baselines. In this paper, the traffic state includes traffic speed and flow. For the traffic speed, we used the proposed model to predict 15, 30, and 60 minutes. The compared baseline models contain both traditional HA and ARIMA, along with neural network models such as STGCN [7], DCRNN [41], ASTGCN [52], GWN [42], LSGCN [43], and USTGCN [44].

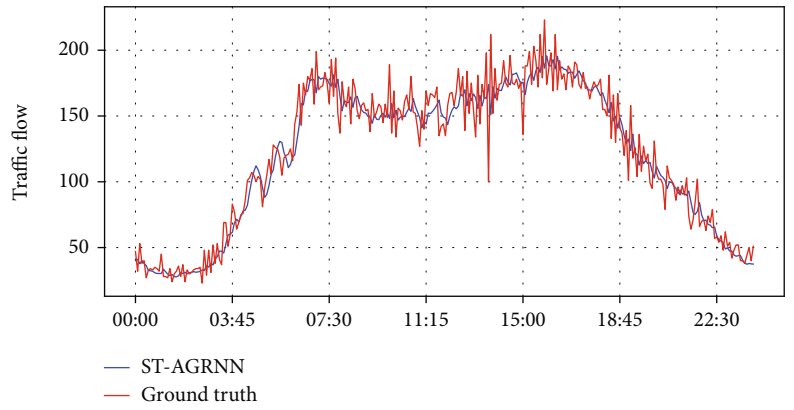
In traffic flow forecasting, all models have a prediction window from 1 to 12, i.e., a prediction time from 5 minutes to 60 minutes, in 5-minute intervals. The baseline models compared included both traditional and neural network models, for a total of 11.

The details of the baseline model are as follows:

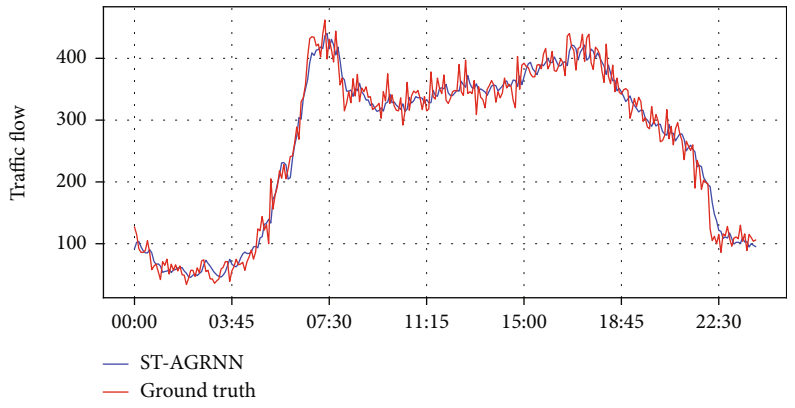
(1) HA: the average traffic information of the previous period is used as the forecast value

(2) ARIMA: autoregressive integrated moving average

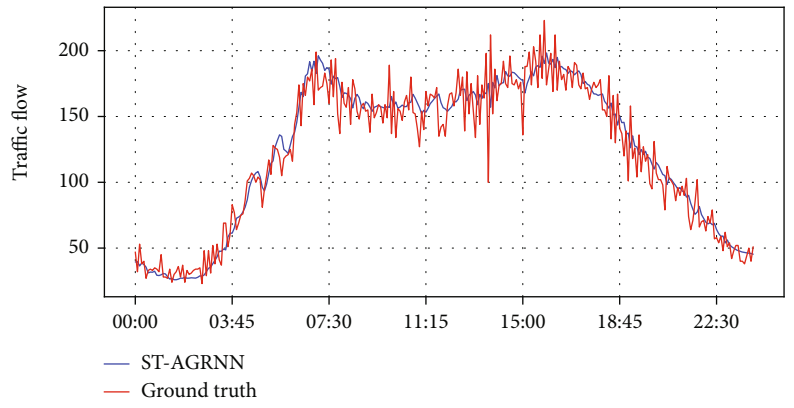
(3) STGCN: spatio-temporal graph convolutional network, which consists of several spatio-temporal convolutional blocks



(a) Node 9 in PeMSD8 for 15 min

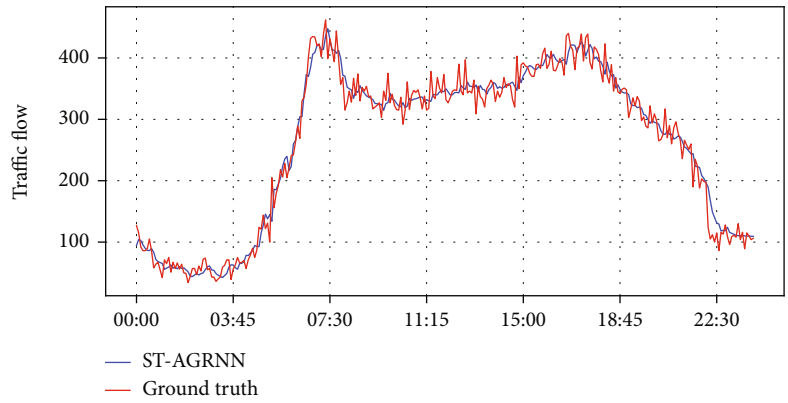


(b) Node 112 in PeMSD8 for 15 min



(c) Node 9 in PeMSD8 for 60 min

Figure 5: Continued.



(d) Node 112 in PeMSD8 for 60 min

Figure 5: Traffic flow forecast visualization in PeMSD8.

(4) DCRNN: diffusion convolutional recurrent neural network, which obtains spatial dependencies through bidirectional random walks and temporal dependencies through an encoder–decoder structure with scheduled sampling

(5) ASTGCN(r): three independent components with the same structure are used to obtain the recent, daily, and weekly dependencies in the traffic data. The spatio-temporal attention mechanism and spatio-temporal convolution are used to obtain the spatio-temporal dependencies within the components. For the sake of experimental fairness, only the recent components are used

(6) GWN: a new adaptive dependency matrix is learned by node embedding to capture the hidden spatial dependencies in the data and obtain temporal dependence via a stacked dilated 1D convolutional component

(7) LSGCN: the model uses spatial gated block and gated linear units (GLU) convolution to capture spatio-temporal features

(8) USTGCN: the model obtains complex spatio-temporal correlations through the proposed unified spatio-temporal convolution strategy

(9) STSGCN [54]: spatio-temporal synchronous graph convolutional network, which uses a spatio-temporal synchronous graph convolutional module to capture the complex localized spatio-temporal correlations and deploys multiple modules to capture the heterogeneities in localized spatio-temporal network series

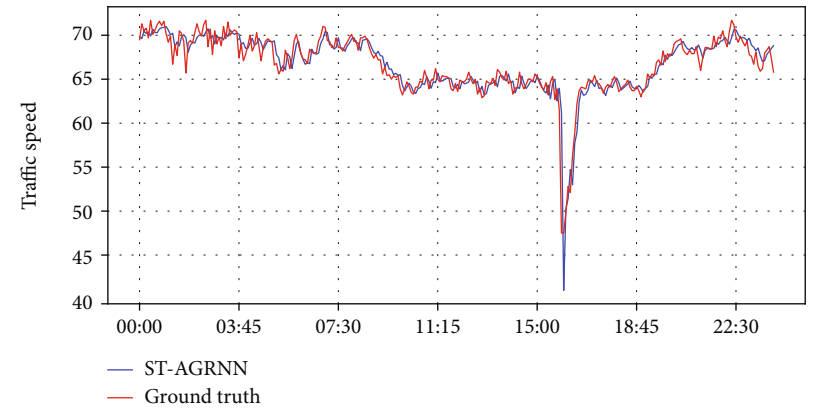
(10) STFGNN [55]: spatio-temporal fusion graph neural network, which uses spatio-temporal fusion graph neural modules and a gated CNN module to capture the spatio-temporal correlations

(11) Z-GCNETs [56]: Z-GCNETs introduce new GCNs with a time-aware zigzag topological layer

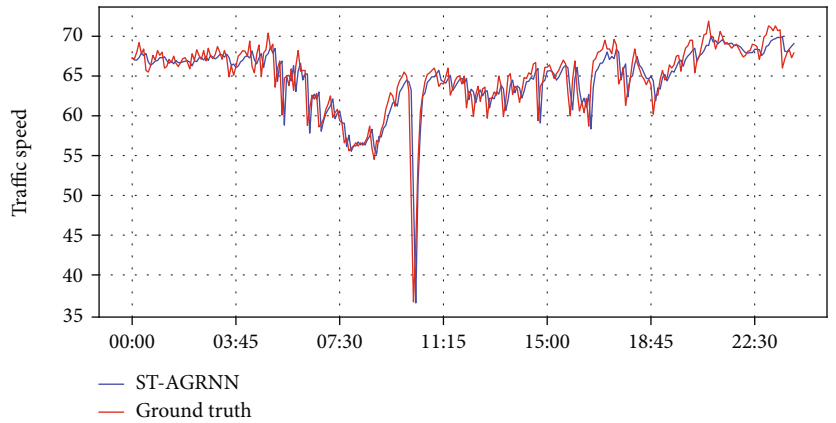
(12) STG-NCDE [57]: spatio-temporal graph neural controlled differential equation, which extends the concept and designs two NCDEs to capture the spatio-temporal correlations

4.4. Experimental Results. The traffic state prediction results for all baseline models and our model are shown in Tables 1 and 2. In Table 1, we can see that our proposed model performs better overall on the datasets PeMSD4 and PeMSD8 compared to the other baseline models for 15-, 30-, and 60-minute traffic speed predictions. Taking the 15-minute speed forecast as an example, on the PeMSD4 dataset, our model is better than HA, ARIMA, DCRNN, STGCN, ASTGCN, GWN, LSGCN, and USTGCN with 53.14, 52.58, 11.85, 19.04, 43.86, 8.46, 17.93, and 15% lower MAE, with 52.41, 58.74,19.72, 21.59, 40.40, 11.94, 19.45, and 12.26% lower RMSE, and with 60.97, 59.21, 19.02, 25.68, 47.83, 18.72, 25.17, and 22.77% lower MAPE, respectively. On the PeMSD8 dataset, our model is better than HA, ARIMA, DCRNN, STGCN, ASTGCN, LSGCN, and USTGCN with 48.73, 46.57, 13.24, 14.7, 31.87, 12.5, and 10.96% lower MAE, with 49.63, 57.49, 20.07, 20.99, 34.9, 15.51, and 3.72% lower RMSE, and with 53.8, 64.38, 21.55, 22.22, 42.4, 18.75, and 12.07% lower MAPE, respectively. From the results, it is clear that ST-AGRNN performs well in both short- and long-term predictions. In particular, on the PeMSD4 dataset, the ST-AGRNN model is optimal on all three-evaluation metrics. Except for the RMSE metric, which is the second best on the PeMSD8 dataset, the other metrics are also optimal for long- and short-term prediction.

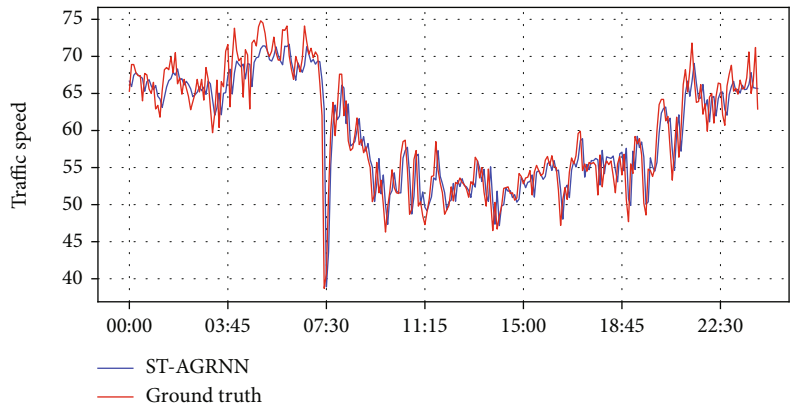
HA and ARIMA are the worst performers because they do not capture spatio-temporal correlations effectively. Since STGCN has cumulative errors, it does not perform as well as DCRNN. DCRNN can effectively obtain complex spatial correlations through diffusion convolution. ASTGCN considers the periodicity of prediction, so it is better than STGCN for long-term prediction.



(a) Node 111 in PeMSD4 for 15 min

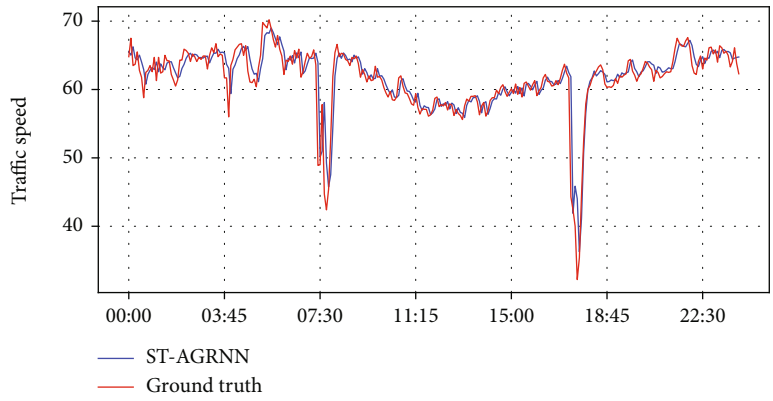


(b) Node 261 in PeMSD4 for 15 min



(c) Node 9 in PeMSD8 for 15 min

Figure 6: Continued.



(d) Node 112 in PeMSD8 for 15 min

Figure 6: Speed forecast visualization in PeMSD4 and PeMSD8.

The spatial gate block of LSGCN integrates the proposed cosAtt and GCN, and in combination with a GLU can effectively extract complex spatio-temporal correlations. Meanwhile, the USTGCN model considers the important historical and present-day patterns in traffic data, in addition to the unified spatio-temporal convolution strategy. Therefore, its prediction performance is the second best.

Table 2 shows the results of traffic flow forecasting performed from 5 minutes all the way to 60 minutes, with a prediction window from 1 to 12, and all of the results are averaged. Compared with all of the baseline models, our proposed model performs the best in traffic flow prediction. From table 2, on the PeMSD4 dataset, our model is better than HA, ARIMA, STGCN, DCRNN, ASTGCN(r), GWN, LSGCN, STSGCN, STFGNN, Z-GCNETs, and STG-NCDE with 50.11, 43.75, 10.34, 10.60, 17.26, 23.78, 11.89, 10.47, 7.37, 2.71, and 1.24% lower MAE, with 49.35, 38.51, 14, 10.27, 14.81, 24.34, 11.39, 10.83, 7.71, 5.08, and 3.49% lower RMSE, and with 54.05, 47.02, 7.37, 9.59, 22.64, 25.91, 2.8, 7.84, 23.61, -0.23, and -0.39% lower MAPE, respectively. On the PeMSD8 dataset, our model is better with 57.11, 51.91, 14.57, 11.11, 18.08, 18.21, 15.67, 12.72, 11.74, 5.07, and 3.23% lower MAE, with 60.92, 47.76, 14.54, 12.17, 17.49, 22.96, 13.49, 13.61, 11.8, 7.8, and 6.69% lower RMSE, and with 66.96, 59.48, 18.42, 15.65, 20.87, 24.19, 17.76, 15.96, 13.11, 7.99, and 7.15% lower MAPE, respectively.

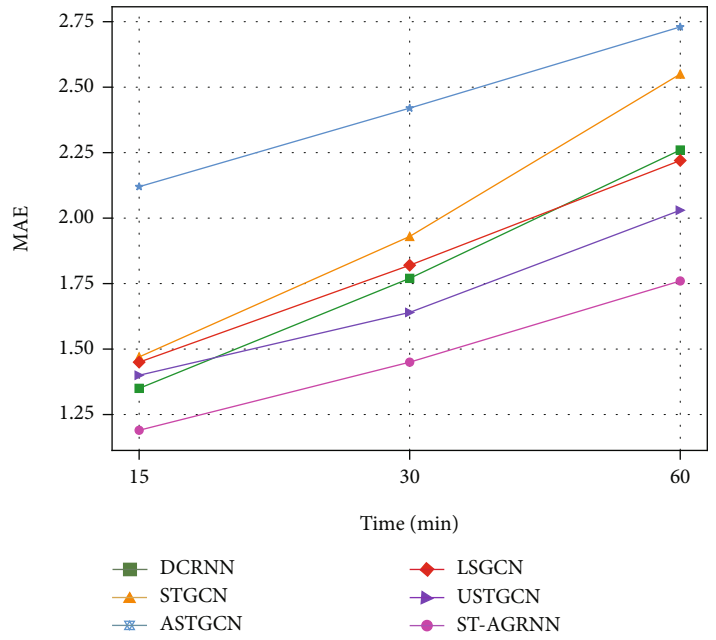
The STSGCN model considers both localized spatio-temporal correlations and the heterogeneities in spatial-temporal data. Therefore, its performance is better than STGCN, DCRNN, ASTGCN(R), GWN, and LSGCN. The SFTGNN obtains hidden spatio-temporal correlations by fusing spatial and temporal graph operations and integrating the gate convolution module at the same time. Z-GCNETs proposed new GCNs with a time-aware Zigzag topological layer to obtain spatio-temporal correlation. The STG-NCDE model uses two neural controlled differential equations (NCDEs) to obtain the temporal and spatial correlations. Since The STSGCN model only extracted localized spatio-temporal correlations, its performance was inferior to that of SFTGNN, Z-GCNets, and STG-NCDE. The ST-AGRNN model obtains both localized and global spatio-temporal correlation and combines location-based graph embedding representation to obtain localized spatial correlation. So, the overall performance on both datasets is better than all baseline models.

4.5. Case Study. We selected two nodes with heavy traffic from the two datasets to show the ground-truth and predicted curves: nodes 111 and 261 in PeMDS4 and nodes 9 and 112 in PeMSD8, as shown in Figures 4 and 5, respectively. From the figures, it can be seen that the model fits this trend well in places with huge traffic flows between 7 : 00 and 9 : 00 a.m. and between 3 : 00 and 6 : 40 p.m. Figure 6 shows the change in the nodes' 15-minute speed. From the figure, the traffic speed also drops sharply at the peak time of corresponding traffic flow.

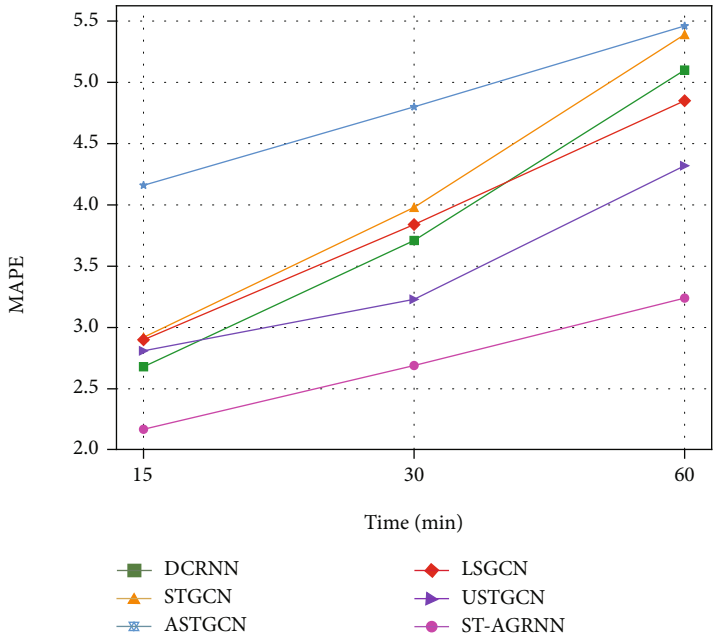
4.6. Error for each Length of Forecasting. Figure 7 shows the trend of the prediction error of the model in terms of prediction speed on two datasets. From the figure, it can be seen that although the error increases for all of the models as the prediction length increases, the error of our model is smaller than baselines and the increasing trend of our model is the flattest. This proves that our model is more stable than the baseline models.

4.7. Ablation Experiments. In the traffic network, the road sections at different locations play different roles in traffic. Road sections in central areas have a greater impact on the surrounding traffic, while remote road sections play a small role in influencing traffic. These are the global spatio-temporal correlations. To verify the importance of global spatio-temporal correlations, we conduct ablation experiments on speed prediction.

From the comparison of the traffic speed prediction results in Table 3, it can be seen that the prediction error of the ST-AGRNN model with the attention mechanism is smaller overall than the error of ST-DWGRU [58] without the attention mechanism. As an example of the 60-minute prediction results, the MAE of ST-AGRNN on the PeMSD4 dataset is 7.3% smaller than that of ST-DWGRU, the RMSE is 9.4% smaller, and the MAPE is 8.2% smaller. The MAE of ST-AGRNN on the PeMSD8 dataset is 2.5% smaller than that of ST-DWGRU, the RMSE is 4.5% smaller, and the

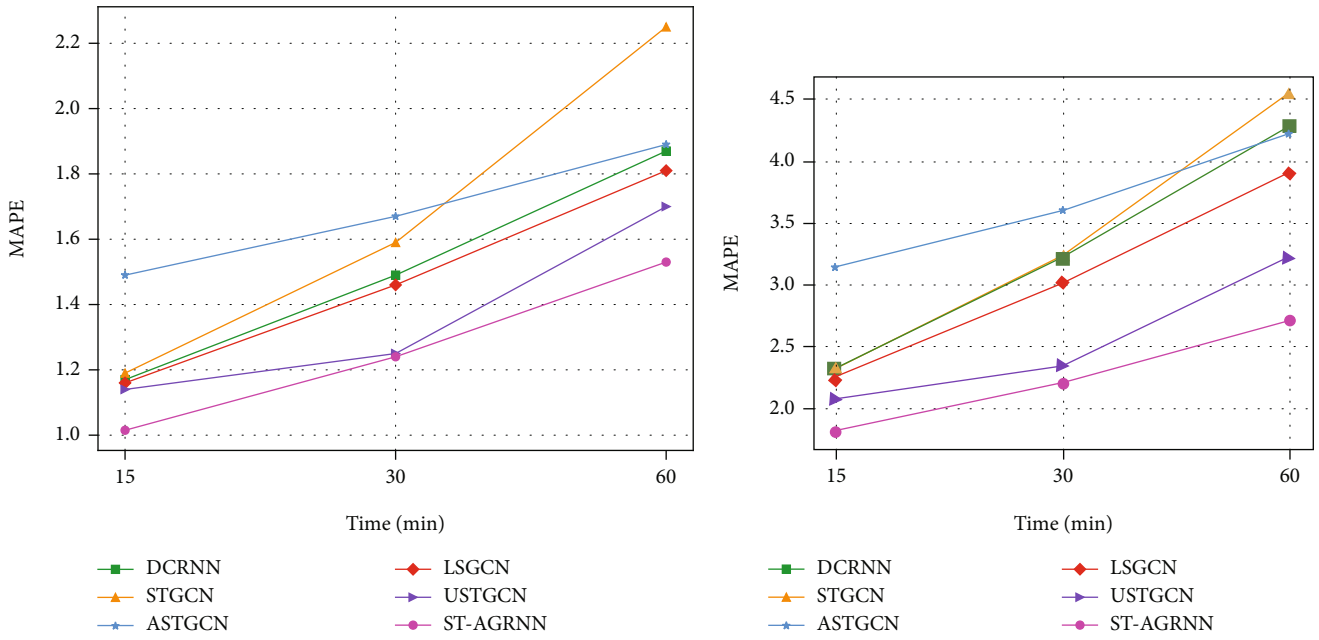


(a) MAE for speed forecast on PeMSD4



(b) MAPE for speed forecast on PeMSD4

Figure 7: Continued.



(c) MAE for speed forecast on PeMSD8

(d) MAPE for speed forecast on PeMSD8

Figure 7: Prediction error trend.

Table 3: Comparison of traffic speed prediction results of the ST-AGRRG and ST-DWGRU models (bold is the best).

T	Metric	PeMSD4		PeMSD8	
		ST-AGRNN	ST-DWGRU	ST-AGRNN	ST-DWGRU
15 min	MAE	**1.19**	1.20	1.015	**1.005**
	RMSE	**2.36**	2.40	**2.07**	2.08
	MAPE	**2.17**	2.21	1.82	**1.81**
30 min	MAE	**1.45**	1.48	**1.24**	1.25
	RMSE	**2.98**	3.12	**2.63**	2.70
	MAPE	**2.69**	2.75	**2.21**	2.24
60 min	MAE	**1.76**	1.90	**1.53**	1.57
	RMSE	**3.63**	4.01	**3.33**	3.49
	MAPE	**3.24**	3.53	**2.71**	2.78

MAPE is 2.5% smaller. From the results, it is clear that the ST-AGRNN model is more effective in obtaining complex spatio-temporal information.

5. Conclusions

A new traffic state prediction model is proposed, in which localized spatial correlation is obtained by a GCN and Deep-Walk, localized temporal correlation is obtained by a GRU, and the global spatio-temporal correlations is obtained by the attention mechanism. Finally, the proposed model ST-AGRNN was tested with two publicly available datasets, namely, PeMSD4 and PeMSD8. In terms of traffic speed prediction, MAE improved by 15-53.14% and 10.96-48.73%, RMSE improved by 12.26-52.41% and 3.72-49.63%, and MAPE improved by 22.77-60.97% and 12.07-53.8% on the PEMSD4 and PEMSD8 datasets, respectively, compared to the baseline models. Meanwhile, the ST-AGRNN model also showed different degrees of improvement in traffic flow prediction compared with the baseline models. From the results, it is clear that ST-AGRNN outperforms all of the baseline models, and is more stable.

Data Availability

Previously reported traffic data that were used to support the study are available. These prior studies (and datasets) are cited at relevant places within the text as references [43].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was financially supported by the National Natural Science Foundation of China (61977001).

References

[1] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2014.

[2] R. Yasdi, "Prediction of road traffic using a neural network approach," *Neural Computing & Applications*, vol. 8, no. 2, pp. 135–142, 1999.

[3] D. Xu, P. Peng, C. Wei, D. He, and Q. Xuan, "Road traffic network state prediction based on a generative adversarial network," *IET Intelligent Transport Systems*, vol. 14, no. 10, pp. 1286–1294, 2020.

[4] H. Xue and F. D. Salim, "TERMCast: Temporal Relation Modeling for Effective Urban Flow Forecasting," in *Advances in Knowledge Discovery and Data Mining. PAKDD 2021*, vol. 12712 of Lecture Notes in Computer Science, Springer, Cham.

[5] L. Cai, K. Janowicz, G. Mai, B. Yan, and R. Zhu, "Traffic transformer: capturing the continuity and periodicity of time series for traffic forecasting," *Transactions in GIS*, vol. 24, no. 3, pp. 736–755, 2020.

[6] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[7] Y. Bing, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional neural network: a deep learning framework for traffic forecasting," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3634–3640, Stockholm, Sweden, 2018.

[8] M.-T. Luong, H. Pham, and C. D. Manning, "Effective Approaches to Attention-Based Neural Machine Translation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421, Lisbon, Portugal, September 2015.

[9] B. Chen, W. Deng, and J. Hu, "Mixed highorder attention network for person re-identification," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 371–381, Seoul, Korea (South), 2019.

[10] P. He, W. Huang, T. He, Q. Zhu, Y. Qiao, and X. Li, "Single Shot Text Detector with Regional Attention," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3066–3074, Venice, Italy, October 2017.

[11] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pp. 167–174, Scottsdale, AZ, USA, December 2015.

[12] J. Bai, J. Zhu, Y. Song et al., "A3T-GCN: attention temporal graph convolutional network for traffic forecasting," *ISPRS International Journal of Geo-Information*, vol. 10, no. 7, p. 485, 2021.

[13] I. Padhi, Y. Schiff, I. Melnyk et al., "Tabular Transformers for Modeling Multivariate Time Series," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3565–3569, Toronto, ON, Canada, June 2021.

[14] S. Wu, X. Xiao, Q. Ding, P. Zhao, Y. Wei, and J. Huang, "Adversarial sparse transformer for time series forecasting," in *Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS'20)*, pp. 17105–17115, Vancouver BC Canada, December 2020.

[15] M. S. Ahmed and A. R. Cook, "Analysis of freeway traffic time-series data by using box-Jenkins techniques," *Transportation Research Record*, vol. 722, pp. 1–9, 1979.

[16] M. M. Hamed, H. R. Al-Masaeid, and Z. M. B. Said, "Short-Term prediction of traffic volume in urban arterials," *Journal of Transportation Engineering*, vol. 121, no. 3, pp. 249–254, 1995.

[17] Q. Y. Ding, X. F. Wang, X. Y. Zhang, and Z. Q. Sun, "Forecasting traffic volume with space-time ARIMA model," *Advanced Materials Research*, vol. 156-157, pp. 979–983, 2010.

[18] M. Van Der Voort, M. Dougherty, and S. Watson, "Combining kohonen maps with arima time series models to forecast traffic flow," *Transportation Research Part C: Emerging Technologies*, vol. 4, no. 5, pp. 307–318, 1996.

[19] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: theoretical basis and empirical results," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[20] J. Guo, W. Huang, and B. M. Williams, "Adaptive Kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 50–64, 2014.

[21] C. P. I. J. V. Hinsbergen, T. Schreiter, F. S. Zuurbier, J. W. C. V. Lint, and H. J. V. Zuylen, "Localized extended kalman filter for scalable real-time traffic state estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 385–394, 2012.

[22] Y. Qi and S. Ishak, "A hidden markov model for short term prediction of traffic conditions on freeways," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 95–111, 2014.

[23] K. Y. Chan, T. S. Dillon, J. Singh, and E. Chang, "Neural-Network-Based Models for Short-Term Traffic Flow Forecasting Using a Hybrid Exponential Smoothing and Levenberg–Marquardt Algorithm," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 644–654, 2012.

[24] J. Wang, W. Deng, and Y. Guo, "New Bayesian combination method for short-term traffic flow forecasting," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 79–94, 2014.

[25] P. Cai, Y. Wang, G. Lu, P. Chen, C. Ding, and J. Sun, "A spatiotemporal correlative _k_ -nearest neighbor model for short-term traffic multistep forecasting," *Transportation Research Part C: Emerging Technologies*, vol. 62, pp. 21–34, 2016.

[26] X. Dongwei, Y. Wang, P. Peng, S. Beilun, Z. Deng, and H. Guo, "Real-time road traffic state prediction based on kernel-KNN," *Transportmetrica A: Transport Science*, vol. 16, no. 1, pp. 104–118, 2020.

[27] Y. Cong, J. Wang, and X. Li, "Traffic flow forecasting by a least squares support vector machine with a fruit fly optimization algorithm," *Procedia Engineering*, vol. 137, pp. 59–68, 2016.

[28] C. Xu, W. Wang, and P. Liu, "A genetic programming model for real-time crash prediction on freeways," *IEEE Transactions*

on Intelligent Transportation Systems, vol. 14, no. 2, pp. 574–586, 2013.

[29] H. Crosby, S. A. Jarvis, and P. Davis, "Spatially-Intensive Decision Tree Prediction of Traffic Flow across the Entire UK Road Network," in *Proceedings of the 2016 IEEE/ACM 20th international symposium on distributed simulation and real time applications*, pp. 116–119, London, UK, September 2016.

[30] A. Nejadettehad, H. Mahini, and B. Bahrak, "Short-term demand forecasting for online car-hailing services using recurrent neural networks," *Applied Artificial Intelligence*, vol. 34, no. 9, pp. 674–689, 2020.

[31] J. W. Van Lint, S. P. Hoogendoorn, and H. J. van Zuylen, "Freeway travel time prediction with state-space neural networks: modeling state-space dynamics with recurrent neural networks," *Transportation Research Record*, vol. 1811, no. 1, pp. 30–39, 2002.

[32] Y. Tian and L. Pan, "Predicting Short-Term Traffic Flow by Long Short-Term Memory Recurrent Neural Network," in *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*, pp. 153–158, Chengdu, China, December 2015.

[33] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU Neural Network Methods for Traffic Flow Prediction," in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 324–328, Wuhan, China, November 2016.

[34] J. Chen, W. Yuan, J. Cao, and H. Lv, "Traffic-flow prediction via granular computing and stacked autoencoder," *Granular Computing*, vol. 5, no. 4, pp. 449–459, 2019.

[35] X. Yuan, B. Huang, Y. Wang, C. Yang, and W. Gui, "Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted SAE," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3235–3243, 2018.

[36] S. Guo, Y. Lin, S. Li, Z. Chen, and H. Wan, "Deep Spatial–Temporal 3D Convolutional Neural Networks for Traffic Data Forecasting," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, pp. 3913–3926, 2019.

[37] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatialtemporal correlation in a hybrid deep learning framework," 2016, https://arxiv.org/abs/1612.01022.

[38] D. Jo, B. Yu, H. Jeon, and K. Sohn, "Image-to-image learning to predict traffic speeds by considering area-wide spatiotemporal dependencies," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1188–1197, 2019.

[39] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *5th International Conference on Learning Representations (ICLR) 2017*, Toulon, France, April 2017.

[40] B. He, Z. Xu, Y. Xu, J. Hu, and Z. Ma, "Integrating Semantic Zoning Information with the Prediction of Road Link Speed Based on Taxi GPS Data," *Complexity*, vol. 2020, Article ID 6939328, 14 pages, 2020.

[41] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion Convolutional Recurrent Neural Network:Data-Driven Traffic Forecasting," in *Proceedings of the ICLR, Vancouver Convention Center*, Vancouver, BC, Canada, April 2018.

[42] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph Wavenet for Deep Spatial-Temporal Graph Modeling," in *Proceedings of the IJCAI*, pp. 1907–1913, Macao, China, August 2019.

[43] R. Huang, C. Huang, Y. Liu, G. Dai, and W. Kong, "Lsgcn: Long shortterm traffic prediction with graph convolutional networks," in *Proceedings of the International Joint Conferences on Artificial Intelligence Organization*, pp. 2355–2361, Yokohama, Japan, 2020.

[44] A. Roy, K. K. Roy, A. A. Ali, M. A. Amin, and A. M. Rahman, "Unified Spatio-Temporal Modeling for Traffic Forecasting Using Graph Neural Network," in *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, Shenzhen, China, July 2021.

[45] D. Bahdanau, K. H. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *ICLR 2015*, San Diego, United States, May 2015.

[46] K. Xu, J. Ba, R. Kiros et al., "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention," in *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37, pp. 2048–2057, Lille France, 2015.

[47] S. Li, X. Jin, Y. Xuan et al., "Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 5243–5253, Vancouver, BC, Canada, December 2019.

[48] D. Daiya and C. Lin, "Stock Movement Prediction and Portfolio Management via Multimodal Learning with Transformer," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2021*, pp. 3305–3309, Toronto, ON, Canada, June 2021.

[49] H. Zhou, S. Zhang, J. Peng et al., "Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35no. 12, pp. 11106–11115, Virtual Conference, February 2021.

[50] C. Park, C. Lee, H. Bahng, Y. Tae, S. Jin, and K. Kim, Eds.S. Ko and J. Choo, "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 (CIKM '20),*, p. 1215, New York, NY, USA, October 2020.

[51] X. Wang, Y. Ma, Y. Wang, W. Jin, X. Wang, J. Tang, C. Jia, and Y. Jian, Eds., "Traffic Flow Prediction Via Spatial Temporal Graph Neural Network," in *Proceedings of the Web Conference 2020 (WWW '20)*, pp. 1082–1092, New York, NY, USA, April 2020.

[52] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention Based Spatialtemporal Graph Convolutional Networks for Traffic Flow Forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 922–929, Honolulu, Hawaii, USA, 2019.

[53] J. You, R. Ying, and J. Leskovec, "Position-aware graph neural networks," *ICML'19*, vol. 97, pp. 7134–7143, 2019.

[54] 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 914–921, New York Hilton Midtown, New York, New York, USA, 2020.

[55] M. Li and Z. Zhu, "Spatial-Temporal fusion graph neural networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35no. 5, pp. 4189–4196, Virtual Conference, 2021.

[56] Y. Chen, I. Segovia, and Y. R. Gel, "Z-GCNETs: Time Zigzags at Graph Convolutional Networks for Time Series Forecasting," in *Proceedings of the 38th International Conference on Machine Learning*, pp. 1684–1694, Virtual Conference, 2021.

[57] J. Choi, H. Choi, J. Hwang, and N. Park, "Graph neural controlled differential equations for traffic forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36no. 6, pp. 6367–6374, Virtual Conference, 2022.

[58] J. Yang, J. Li, W. Lu, L. Gao, and F. Mao, "Spatio-temporal DeepWalk Gated Recurrent Neural Network: A Deep Learning Framework for Traffic Learning and Forecasting," *Journal of Advanced Transportation*, vol. 2022, Article ID 4260244, 11 pages, 2022.

Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 2723101, 9 pages
https://doi.org/10.1155/2022/2723101

Research Article

Traffic Flow Prediction Based on Multi-Spatiotemporal Attention Gated Graph Convolution Network

Yun Ge, Jian F. Zhai, and Pei C. Su

Department of Computer Teaching and Research, University of Chinese Academy of Social Sciences, Beijing 102488, China

Correspondence should be addressed to Yun Ge; gaiyun@ucass.edu.cn

Received 25 May 2022; Revised 14 July 2022; Accepted 21 July 2022; Published 9 September 2022

Academic Editor: Yong Zhang

Accurate prediction of traffic flow plays an important role in ensuring public traffic safety and solving traffic congestion. Because graph convolutional neural network (GCN) can perform effective feature calculation for unstructured data, doing research based on GCN model has become the main way for traffic flow prediction research. However, most of the existing research methods solving this problem are based on combining the graph convolutional neural network and recurrent neural network for traffic prediction. Such research routines have high computational cost and few attentions on impaction of different time and nodes. In order to improve the accuracy of traffic flow prediction, a gated attention graph convolution model based on multiple spatiotemporal channels was proposed in this paper. This model takes multiple time period data as input and extracts the features of each channel by superimposing multiple gated temporal and spatial attention modules. The final feature vector is obtained by means of weighted linear superposition. Experimental results on two sets of data show that the proposed method has good performance in precision and interpretability.

1. Introduction

With the development of urbanization process, people's demand for transportation are increasing day by day. Whether to build an effective transportation system has become an important factor in restricting development of city. Accurate prediction about traffic condition plays a very important role in people's daily travel planning, urban traffic planning, and traffic management and strategy. In order to improve the efficiency of transportation and reduce the time cost for transportation activities in daily work and life, this paper proposed a traffic speed prediction model based on multi-spatiotemporal gated graph convolutional network with attention mechanism.

Based on analyzing the urban road's traffic flow situation, the velocity of vehicles on the road can be predicted. Traffic speed prediction can not only provide managers with scientific decision-making information but also provide appropriate route guidance for urban travelers, which is an important guarantee for the unimpeded flow of urban traffic. Currently, the main traffic speed prediction model can be divided into three categories: the statistical-based methods, the machine learning-based methods, and deep learning-based methods. The statistical-based methods are constructed based on the theory of statistical forecasting and mainly contain the historical average analysis prediction method, regression difference moving average method [1] (ARIMA), Kalman filtering method [2, 3], the grey prediction model method, etc. These models usually have strict requirements on input data and these corresponding algorithm structures are relatively fixed. However, the prediction result of traffic flow can be easily affected by some random interference factors, such as traffic accidents, weather, and special events, which can make the prediction accuracy relatively low. The second type is the method based on machine learning way, which can not only model the nonlinear feature of traffic flow but also continuously adjust the model parameters by means of adaptive learning methods to obtain more accurate prediction results. Therefore, the methods based on machine learning gradually replace the statistical theory-based methods and become the next research focus in traffic flow prediction field.

Algorithms used for prediction mainly include support vector machine [4], K-nearest neighbor [5], Bayesian network [6], and other methods. The third category is the way based on deep learning model, which is also the most common used method at present. Deep learning methods can be used to learn the features about input data without mankind intervention. Such models have strong requirements on nonlinear mapping characteristic feature and less strict requirements on data than those model-driven methods, so they will be more suitable to model the uncertainty status of traffic flow and improve the prediction precision. Because of these advantages, some researchers have applied deep learning methods into the field of traffic prediction and achieved remarkable progress.

Shao et al. [7] applied the Long Short-Term Memory Network (LSTM) model into traffic flow prediction and improved the accuracy of flow prediction by calculation of the spatial characteristics. Liu et al. [8] used the Gated Recurrent Unit (GRU) model to predict urban traffic flow. Since the internal neural cell number of the GRU model is less than that of LSTM, the prediction performance is still good. Traffic flow data not only has dynamic correlation in time but also has strong dynamic correlation in space. In order to extract the temporal and spatial features effectively, Shi et al. [9] proposed Conv-LSTM model, which comprehensively uses CNN and LSTM to capture the spatiotemporal feature. Liu et al. [10] applied it to short-term traffic prediction. Yao et al. [11] put forward the spatiotemporal dynamic network (STDN) model and used CNN and LSTM to capture the spatiotemporal feature. Zhang et al. [12] proposed the spatiotemporal residual network (ST-ResNet) model which uses different residual units to model the information of time proximity, periodicity, and tendency.

Zhao et al. [13] proposed a temporal graph convolutional network (T-GCN) model based on combining GCN model with GRU model. The GCN model was used to learn complex topological structure for capturing spatial feature, and GRU model was used to learn temporal feature of traffic flow changing data. Yu et al. [14] proposed a spatiotemporal graph convolutional network (STGCN) model, which uses one-dimensional CNN model to capture the time dynamic feature and the GCN model was used to obtain the spatial feature of local traffic data. In order to capture the dependence between temporal and spatial feature, Li et al. [15] improved the gated GRU unit and proposed diffused convolution gated loop unit (DCGRU). Combined with encoder and decoder, the DCRNN model for Seq2Seq was proposed. In view of the traffic flow data being time-dependent, Guo et al. [16] used three different components to extract feature from historical data. Song et al. [17] used three different continuous time slices to construct local spatiotemporal models and used sliding windows to segment time periods into three parts. By stacking multiple graph convolution layers, a spatiotemporal synchronous graph convolution network (STSGCN) was established to extract long-term spatiotemporal feature. Although the T-GCN model uses two-layer graph convolution network to aggregate the spatial information about two level neighbors, it still ignores deeply excavating the spatial correlation between higher-order neighbor nodes. Therefore, K-order Chebyshev graph convolution which can cover k-order neighbor nodes was used to complete the spatial convolution operation and extract the spatial feature of higher-order neighbor nodes. In addition, T-GCN model uses a single time series to perform prediction work without mining time dependence between different slices. The spatiotemporal information can also be used in other fields. Wang et al. [18] use spatiotemporal correlation information to reconstruct traffic data. Wang et al. [19] perform passenger flow prediction via dynamic hypergraph convolution networks. Yu et al. [20] proposed a low-rank dynamic mode decomposition model for short traffic flow prediction.

Although these methods have been able to predict traffic flow very precisely, there are still some areas that can be improved. The existing methods can be improved from the following two aspects: improving the scope of neighborhood scale and considering the influence of data with different time periods on future traffic. The traffic flow status in any node on the traffic network can be affected not only by the first-level neighborhood nodes, but also by the second-level neighborhood nodes. The change rule of traffic flow is periodic. The traffic flow on the road is generally large during working hours, and small during other times. Traffic information with different time periods has different influence on the status change of traffic flow in the future. It is of great help to improve the prediction of traffic flow to comprehensively consider the changing rules of traffic flow in different time periods.

Therefore, this paper extracts three different time series datasets which are monthly data, daily data, and weekly data to fully capture temporal characteristics. In general, this paper proposes a multichannel gated spatiotemporal graph convolution with attentional mechanism, which puts three different time series datasets into the model and gets the feature by stacking multiple gated spatiotemporal blocks. The forecasting work was finished by combining all the three different feature vectors with the help of weighted linear combination operation. The main contributions of this paper can be summarized as follows:

(i) We developed a multichannel gated spatiotemporal graph convolution network to learn the dynamic feature of traffic flow data. Specifically, a multichannel feature extraction and fusion framework was proposed. The temporal feature of the traffic data was fully exploited.

(ii) A novel spatiotemporal calculation module was designed by adding attention mechanism. It helps the model to pay more attention to import the feature in each channel.

(iii) Extensive experiments are carried out on read traffic data, which can verify the effectiveness of the model proposed in this paper. The performance of this

prediction model has a certain progress compared to existing methods.

The rest of this paper is organized as follows. Section 2 describes the related work on traffic flow forecasting and the development of graph neural networks. Section 3 introduces the detailed architecture of proposed forecasting network with gated graph neural network and attention. Section 4 presents the experiment setting and the experiment results. Finally, Section 5 concludes the work and presents the findings of this research.

2. Related Work

In this section, we will briefly introduce corresponding theories and definitions referring to the proposed model.

2.1. Graph Neural Network. Convolutional neural network is a feed-forward neural network based on convolutional operation, which can efficiently compute feature information from structured data such as image, speech, and text. However, there are a lot of unstructured data in daily life, such as social network data, human skeleton data, traffic flow data, and other data without regular structure. The traditional CNN models cannot effectively model such unstructured data. In order to effectively capture the local spatial feature of these data, a graph convolution network model for unstructured data was proposed. Graph convolutional network is a kind of neural network structure which is popular in recent years. It is a kind of neural network which extends the convolution operation to graph structure data. Compared with traditional convolutional network models which can only be used in structured data computation, graph convolutional networks are special in capturing unstructured data. The road network structure in reality is typical unstructured data. Thus, the local feature of traffic data can be extracted effectively based on using graph neural network.

The existing graph convolution operation-based methods mainly can be divided into two types: the way based on spatial domain and the way based on frequency domain. The spatial domain-based operation can be defined by aggregating the feature information about adjacent nodes in the graph. The frequency domain-based operation uses Fourier transform to realize the convolution calculation in frequency domain.

According to graph theory, the properties of graph structure can be obtained by calculating Laplacian eigenvalue and eigenvector about adjacency matrix, and the spectrum convolution result on graph can be obtained by calculating the convolution of signal $x \in R^N$ and graph convolution kernel g_θ. The purpose of graph convolution is to predict the state of the node at the next moment according to current status of the node in a graph, which can be defined as

$$H^{(l+1)} = f\left(H^{(l)}, A\right), \tag{1}$$

where $H^{(l)}$ denotes all the note status at time l, A denotes the adjacent matrix, and $f(\cdot)$ denotes mapping function. Different mapping function represents different GCN models. Usually, the node status in the next moment can be obtained by calculating the linear combination of its adjacent notes through multiplying the adjacent matrix with the current status matrix, and the final expression can be defined as follows:

$$H^{(l+1)} = \sigma\left(AH^{(l)}W\right). \tag{2}$$

The weight matrix was used to perform linear mapping operation and the function $\sigma(\cdot)$ was used to calculate the nonlinear mapping operation. The function of the adjacency matrix and state matrix multiplication was used to calculate the addition of adjacency nodes in a matrix manner. However, the information of node itself has not been taken into account. The direct way to solve this problem is adding an identity matrix to the adjacency matrix so as to add the self-loop information of each node into the adjacency matrix. In addition, with the accumulation of the GCN operations, the dimension difference of status information between nodes in the graph will become large. In order to maintain the stability of the operation, the matrix information needs to be normalized before each calculation. Graph convolution operators usually adopt graph Laplacian matrix as the substitution of adjacency matrix, and graph convolution calculation function can be defined as follows:

$$H^{(l+1)} = \sigma\left(D^{-(1/2)}\widetilde{A}D^{-(1/2)}H^{(l)}W^{(l)}\right), \tag{3}$$

where $\widetilde{A} = A + I_N$ represents a new adjacency matrix with self-loop information, $\widetilde{D} = \sum_i \widetilde{A}_{ij}$, $H^l \in R^{N\times F}$ denotes the nodes information in l-th layer, $H^0 = X$, X denotes the initial status of graph nodes, and $W^l \in R^{F\times F}$ denotes the weight value in the l-th layer. Each calculation of graph convolution is the extraction of first-order neighborhood information. Multiorder neighborhood information can be realized by superimposing several convolutional layers.

2.2. Spatiotemporal Attention Mechanism. Graph convolutional neural network can capture the local spatial correlation between adjacent nodes in graph, but different adjacent points have different impact on the current node. The key idea of spatial attention mechanism is to pay adaptive attention to the characteristics of the most relevant nodes according to the input data. In time slice, the information of road network is changing dynamically all the time. Therefore, using spatial attention mechanism and temporal attention mechanism to adaptively capture the node information with higher correlation in each dimension will be of great help to improve the prediction accuracy.

In this paper, soft attention [21] mechanism is used to calculate attention weight. It can extract features from the input sequence and adaptively calculate the importance of each node from the road network information at different time. Firstly, the information of all nodes at time t was aggregated into a vector. The aggregated information

includes the spatial characteristics and node information of the road network at time t can be expressed as follows:

$$q_t = \text{relu}\left(\sum_{i=1}^{N} W h_{ti}\right), \tag{4}$$

where W denotes the trainable parameters and h_{ti} denotes hidden state value of the i-th node in time t. The attention values about all nodes can be formulated as follows:

$$\alpha_t = \text{Sigmoid}\left(U_s \tanh\left(W_h h_t + W_q q_t + b_s\right) + b_u\right), \tag{5}$$

where $\alpha_t = (\alpha_{t1}, \alpha_{t2}, \ldots, \alpha_{tN})$ and α_{ti} denotes the attention value of the i-th nodes at t time. U_s, W_h, and W_q denote trainable parameters; b_s and b_u denote bias vector. This attention mechanism firstly spliced the aggregated information of all nodes at time t with the information of all nodes at the same time and then obtained the attention weight of each node relative to all nodes through the full connection layer. In order to calculate the nonlinear mapping information of nodes at different time, this paper uses the structure of two fully connected layers to calculate attention value. The second hidden state h_{ti} of the i-th node at t time can be calculated by $(1 + \alpha_{ti}) \cdot h_{ti}$ and the weighted graph state will be input into next layer.

2.3. Gated Convolution Network. Gated linear unit was proposed by Dauphin et al., which is a convolutional neural network model with gated mechanism. This model was mainly used to replace the recurrent neural network in natural language processing model. Compared with the gated unit in RNN model, this unit has the advantages of lower complexity, faster gradient propagation efficiency, and being less prone to gradient disappearance or gradient explosion. In addition, the gated linear unit can also process the input data in parallel, which can improve the accuracy of the model as well as the computational efficiency. Let X denote layer input, h_l denote output of this layer which also represents the hidden states of this layer, W and V denote two different convolution cores, and b and c denote two bias parameters; the gated convolution model can be expressed as follows:

$$h_l(X) = (X * W + b) \otimes \sigma(X * V + c). \tag{6}$$

The output of the model was realized through dot product calculation between linear mapping result vector and nonlinear mapping vector. The linear mapping vector can be obtained by multiplying the input vector X with parameter vector W. The nonlinear mapping vector can be calculated by multiplying the input vector X with parameter vector V at first. Then, the nonlinear mapping function can be obtained by using nonlinear function $\sigma(\cdot)$. Because the output of function $\sigma(\cdot)$ can only be 1 or 0, the function of multiplying these two vectors is to perform gated selection operation for each node.

3. Methodology

In this section, we will describe the framework of the proposed method. The traffic flow information was extracted in three channels separately. In each channel, there are two spatiotemporal blocks to fetch space-temporal feature. Each ST block is composed of a spatial block and a temporal block which are used to fetch the spatial feature and temporal feature separately. The input of the three channels corresponds to the traffic flow data containing three impassable periods, respectively. The model structure is shown in Figure 1.

3.1. Problem Definition. The goal of traffic prediction is to predict the traffic information in a certain time based on the historical traffic information on the road. This paper takes traffic speed forecasting as the main objective of the study. This prediction work is performed based on traffic flow data on the road which was collected by traffic sensors distributed throughout the network. Typically, traffic flow data refers to the number of vehicles that pass through a sensor during a specified period of time. The topology structure composed of all sensors in the road network was defined as $G = (V, E, A)$. The vector $V = \{v_1, v_2, \ldots, v_N\}$ denotes vertex set. Assume that only one sensor was placed on each road and the road in the road network can be represented by the sensor. Let N denote the number of the codes and E denote the set of edges in the network. The adjacent matrix $A = R^{N \times N}$ was used to denote the connection between nodes. The feature matrix $X_t \in R^{N \times P}$ denotes the flow status in time t, and P denotes the length of feature vector. The traffic flow prediction problem can be defined as follows: given the traffic flow's status at the time t and other historical data, the $t+1$ time traffic flow data can be calculated in the form of the following equation:

$$\left[X_{t+1}, X_{t+2}, \ldots, X_{t+p}\right] = f\left(G; \left(X_{t-n}, X_{t-n+1}, \ldots, X_t\right)\right), \tag{7}$$

where t is the length of historical time series and n is the length of time series that need to be predicted.

3.2. Graph Convolution on the Traffic Data. Since the structure of the road network is an irregular structure, the traffic flow data generated by vehicles on the road network is also irregular, and it is very suitable to use GCN model to calculate the feature of traffic flow. Because the standard graph convolution computation is too huge, the Chebyshev inequality was often used to get the approximate solutions, and the approximate equation can be formulated as follows:

$$\theta * \varphi x = \theta(L)x \approx \sum_{k=0}^{K-1} \theta_k T_k(\tilde{L})x, \tag{8}$$

where θ is the graph convolution kernel, $T_k(\tilde{L}) \in R^{N \times N}$ is k-order Chebyshev inequality, $L = 2(L/\lambda_{\max}) - I_N \in R^{N \times N}$,

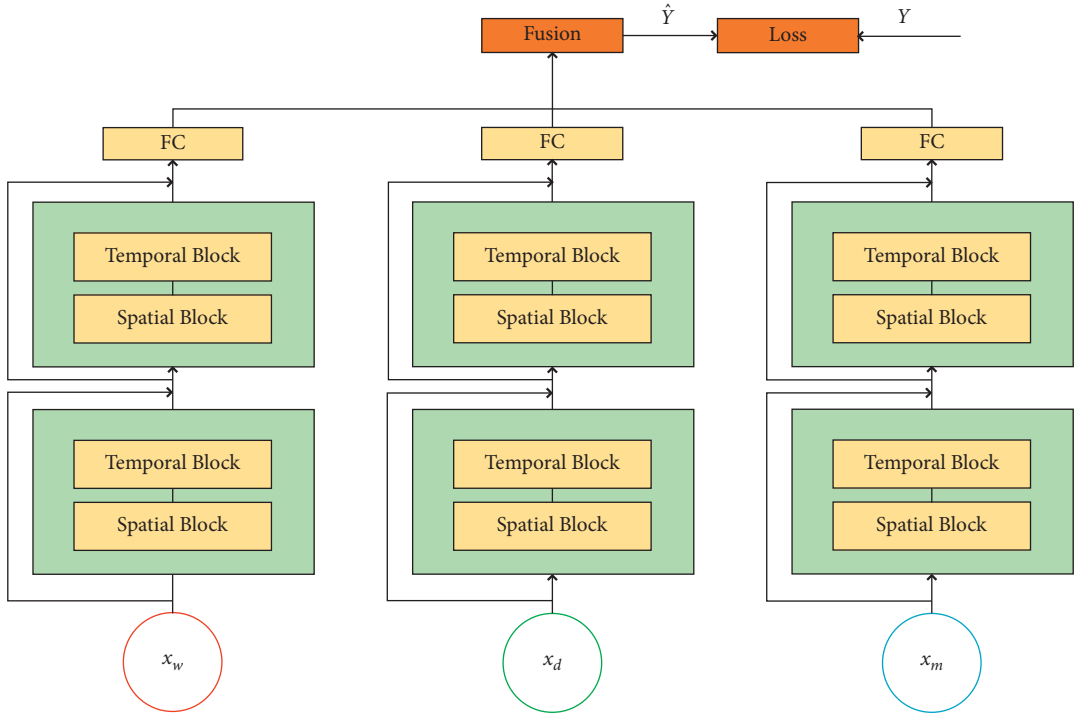


Figure 1: The framework of the proposed model.

$\lambda_{\max}$ is the largest eigenvalue of the Laplace matrix, k is the size of convolution kernel, and the k-th Chebyshev inequality's recursive definition is $T_k(\tilde{L}) = 2xT_{k-1}(x) - T_{k-2}(x)$, while $k = 0$ and $T_0(x) = 1$.

In order to effectively learn local spatial dynamic feature, the spatial attention matrix $W \in R^{N\times N}$ was multiplied with the k-order Chebyshev inequality $T_k(\tilde{L})$ based on dot product. The concrete equation can be formulated as follows:

$$\theta * \varphi x = \theta(L)x \approx \sum_{k=0}^{K-1} \theta_k \left(T_k(\tilde{L}) \odot W\right)x. \tag{9}$$

In this paper, k-order Chebyshev inequality was applied to extract feature of road network information. The k-order convolution operator of Chebyshev graph convolution can cover the features of k-order neighborhood nodes.

3.3. Multiperiod Flow Data Series. In order to capture the temporal dynamic characteristics of traffic flow, this paper uses three different spatiotemporal components to extract the characteristics of historical traffic data. This paper constructs three different traffic flow data sequences with three different periods: week, day, and hour.

3.3.1. The Weekly Periodic Series. The weekly periodic series data X_w was composed of traffic data sampled in weeks. They have the same weekly properties and time intervals as the forecast period. In terms of the variation trend and peak value of traffic conditions, the traffic flow on weekdays is similar to that on weekdays, but not on nonweekdays. Therefore, training with weekly periodic data can help us capture differences between weekdays and nonweekdays data.

3.3.2. The Daily Periodic Series. The daily periodic series X_d was composed of the traffic data sampled in days. Due to the regularity of people's activity track, the traffic flow shows periodic fluctuation. For example, the morning and evening rush hours on weekdays may have similar traffic volumes. Therefore, daily correlation data were added to extract temporal and spatial dynamic correlation.

3.3.3. The Minutely Periodic Series. The minutely periodic series X_m was composed of the traffic data sampled in minutes.

The sequence that has the greatest impact for the future traffic is the traffic situation in the adjacent period. If the current traffic flow on the adjacent road is large, the possibility of congestion at the next moment of this section will be large.

All these three data series have the same structure and can be calculated in the same way. There are two spatiotemporal blocks in the model and a fully connected layer in the end. The spatiotemporal block was composed of spatial block and temporal block. Each block has an attention module. In order to avoid the decrease of training accuracy,

we introduce residual learning module between spatiotemporal blocks. In the end of forecasting model, the outputs of the three channels will be merged by a parameter matrix to form the final feature vector.

3.4. Gated Convolution for Feature Extracting. Graph convolution model can be used to extract spatial information of traffic data effectively. However, traffic flow data is a typical time flow data. Effectively extracting the characteristics of traffic flow information on the time axis is of great help to improve the accuracy of prediction. In this paper, gated convolution model is used to extract temporal and spatial features of traffic information. Compared with RNN model, the gated convolution model has a simpler structure and smaller computational time. In order to capture the characteristics of traffic data on the time axis, we apply gated convolution operation on each time axis to capture the dynamic characteristics of traffic flow data.

3.5. Multichannel Data Merge. Each spatiotemporal convolution module consists of a graph convolution module for spatial information and a gated convolution model for time domain. The gated convolution module captures the features of the time axis along the time axis. The outputs of different channels have different weight in prediction. In this paper, we combine them based on linear combination operation. The fusion equation is shown as follows:

$$\widehat{Y} = W_w \odot \widehat{Y}_w + W_d \odot \widehat{Y}_d + W_m \odot \widehat{Y}_m, \tag{10}$$

where $\odot$ denotes the element-wise Hadamard product, $\widehat{Y}_w$ denotes the output of the channel weekly data, $\widehat{Y}_d$ denotes the output of the channel daily data, and $\widehat{Y}_m$ denotes the output of the channel minutely data. W_w, W_d, and W_m are weighted parameters corresponding to different channel data. In this paper, we take 0.4, 0.2, and 0.4 as the default weight parameter values, because traffic flow status in former time has more impact on the traffic data in the next time.

3.6. Loss Function. The goal of model training is to minimize the error between the actual traffic speed and the predicted value on the road. In this paper, Y_t and $\widehat{Y}_t$ were used to represent the actual traffic speed and predicted speed, respectively. The loss function of MSTAGCN was shown in the following equation:

$$\text{loss} = \left\| Y_t - \widehat{Y}_t \right\| + \lambda L_{\text{reg}}. \tag{11}$$

In this formulation, the first term was used to measure the error between the actual speed and the predicted value. The second term represents L_{reg}, and the regularization term, which helps to avoid the overfitting problem, is a hyperparameter.

4. Results and Discussion

4.1. Datasets. The experiment datasets used in this paper are PeMS04 and PeMS08 which belong to Caltrans performance evaluation system (PeMS, https://www.pems.dot.ca.gov). The geographic information and time information are contained in the data. The PEMS04 is the traffic flow data collected from San Francisco Bay, which includes 3,848 sensors on 29 roads. We pick out the experiment data from 307 sensors. The time range of the dataset is from January 1 to February 28 in 2018 which covers 59 days. The PEMS08 was the traffic flow data collected from SAN Bernardino, which includes 1,979 sensors on 8 roads. We pick out the data from 170 sensors as experiment data.

4.2. Data Preprocessing. The data in these two datasets are sampled in every five minutes. Each sensor contains 288 data records per day, and each record contains three features. They are the traffic flow, average vehicle speed, and occupancy rate responding to sensors during that time period. The spatiotemporal data were divided into training set, validation set, and test set in the ratio of 6 : 2 : 2. At the same time, range normalization was carried out for each feature to keep the data value between [0,1]. The specific calculation formula is as follows:

$$x^* = \frac{x - \min}{\max - \min}. \tag{12}$$

By using the distance between different sensors, the adjacency matrix of the graph was established using the threshold Gauss kernel. The calculation process of the threshold Gaussian kernel can be formulated as follows:

$$W_{ij} = \begin{cases} e^{\left(\text{dist}\left(v_i, v_j\right)^2 / \sigma^2\right)}, & \text{dist}\left(v_i, v_j\right) < s, \\ 0, & \text{dist}\left(v_i, v_j\right) \geq s, \end{cases} \tag{13}$$

where W_{ij} represents the weight of the edge between sensor v_i and sensor v_j, $\text{dist}(v_i, v_j)$ represents the distance between sensor v_i and sensor v_j, σ^2 is the variance of the distance, and s is the threshold. As there are almost no sensors over 1000 meters in the dataset, the threshold s is 1000.

4.3. Evaluation Metrics Subheadings. To evaluate the performance of the proposed model, we choose three metrics to evaluate the difference between real traffic value Y_t and estimated value $\widehat{Y}_t$, which was shown in the following equations.

(1) Root Mean Square Error (RMSE) is calculated as follows:

$$\text{RMSE} = \sqrt{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(y_j^i - \widehat{y}_j^i\right)^2}. \quad (14)$$

(2) Mean Absolute Error (MAE) is calculated as follows:

$$\text{MAE} = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left|y_j^i - \widehat{y}_j^i\right|. \quad (15)$$

(3) Mean Absolute Percentage Error (MAPE) is calculated as follows:

$$\text{MAPE} = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left|\frac{y_j^i - \widehat{y}_j^i}{y_j^i}\right|, \quad (16)$$

where y_j^i denotes the real traffic flow data value in the i-th time, $\widehat{y}_j^i$ denotes the predict value, M denotes the number of samples, and N denotes the number of roads. Specifically, the rule of metrics measuring prediction error is as follows: the smaller the error, the higher the accuracy of the prediction.

4.4. Experiment Settings. To verify the validity of the model, the MSTAGCN model proposed in this paper was compared with the classical GRU model and the recently proposed DCRNN, T-GCN, ASTGCN, and STSGCN models. Table 1 shows the hyperparameter settings of each model and the word layers means the number of hidden layers. The word units represents the number of computing units in each hidden layer and all models in the experiment are composed of the same number of units. k denotes the order of graph convolution, and T_w, T_d, and T_m represent the length of weekly, daily, and minutely sequence.

4.5. Experiment Result. The experimental results are shown in Table 2. In The PEMS08 dataset, MSTAGCN model is always superior to other benchmark models in terms of accuracy. In EMS04 dataset, MST-GCN has the smallest prediction error compared with other forecasting methods and has slightly larger errors in MAE and MAPE result. In the RMSE evaluation results, the proposed method has larger error than STSGCN methods. Due to the simplest model structure, the GRU model has the worst performance in both datasets. The lower prediction results of the former spatial analysis-based model demonstrate that those methods have not effectively model the nonlinear information of the traffic data. In general, the deep learning-based methods have better performance than those non-deep learning models and the convolution operation plays an importance role in improving the accuracy of prediction. The convolution operation can effectively capture the local feature in both the spatial information and the temporal information. Simultaneously using spatial and temporal information is the other effective prediction improving routine. As we can see, the last four methods have better performance than other methods. Besides, the MSTAGCN performs better than other methods, indicating that the multichannel mechanisms applied in the proposed model are effective in capturing the changing routine characteristic. Our MSTAGCN achieves better performance than the previous models proving the feature about traffic changing is nonlinear and single input information cannot provide sufficient information for feature learning.

Figures 2 and 3 exhibit the prediction performance on these two datasets. The GRU model only considers the temporal characteristics and does not take advantage of the spatial information of road network. The accuracy of GRU is not as good as that of temporal correlation method. GRU only considers the temporal correlation and does not use the spatial correlation of road network, so the accuracy of GRU is not as good as that of the method using temporal and spatial correlation. The DCRNN and T-GCN model spatial and temporal feature information separately, but they only use a single time window to extract long-term dependence without considering impaction caused by the periodicity of different time windows. ASTGCN and STSGCN both use different spatiotemporal components to extract corresponding correlation from time windows, but they ignored the correlation between different time period channels. So, the prediction precision will be relatively reduced. In this paper, the MSTAGCN method considers impaction from different periodic data on traffic forecasting work and uses multichannel structure to fuse the spatiotemporal components, so as to capture the long-term spatiotemporal dependence between different periodic traffic data. Therefore, the prediction accuracy of the proposed model is better than that of the existing models, and the prediction effect is better.

Table 1: Hyperparameter settings for different models.

Models	Layers	Units	k	T_w	T_d	T_m
GRU	3	500	—	—	—	5
DCRNN	2	64	3	—	—	5
T-GCN	3	64	2	—	—	5
ASTGCN	2	64	3	24	12	5
STSGCN	4	64	3	—	—	12
MSTAGCN	3	64	3	2	6	5

Table 2: Performance comparison of different models of traffic flow prediction.

Model	PEMS04			PEMS08		
	MAE	RMSE	MAPE (%)	MAE	RMSE	MAPE (%)
GRU	24.34	43.47	16.59	19.01	35.12	13.23
DCRNN	24.06	34.7	16.00	19.36	31.94	11.18
T-GCN	23.71	34.74	16.37	22.98	32.57	11.88
ASTGCN	22.36	32.6	15.18	18.21	27.99	13.22
STSGCN	22.52	34.62	14.92	17.79	26.33	11.80
MSTAGCN	**22.11**	**32.96**	**14.15**	**15.85**	**23.62**	**11.44**

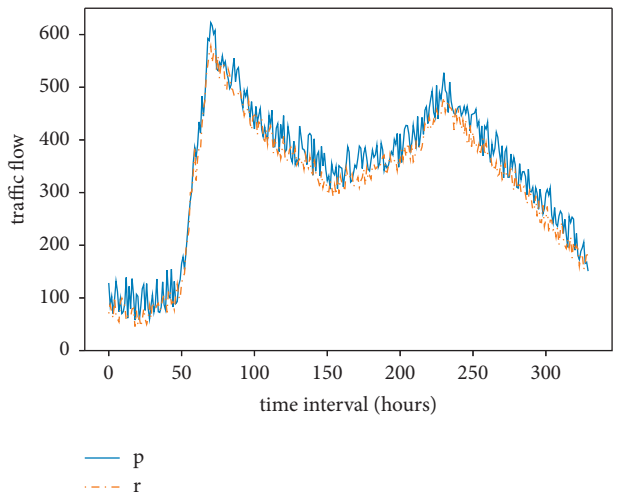


Figure 2: Prediction result on PEMS04.

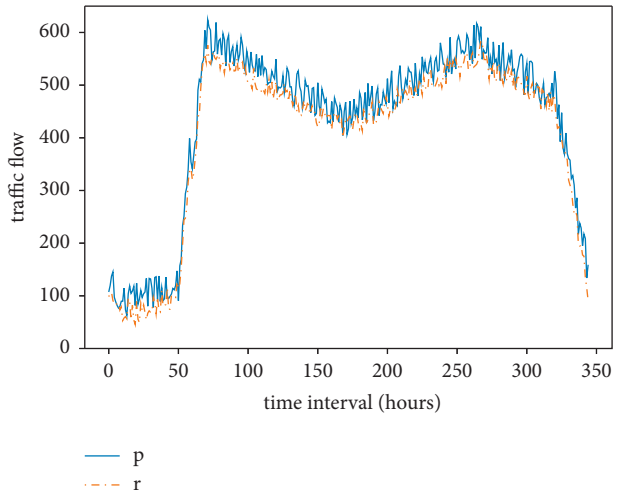


Figure 3: Prediction result on PES08.

5. Conclusions

In this paper, we proposed a multi-spatiotemporal attention gated graph convolution network (MSTAGCN) to capture the spatiotemporal feature about traffic flow data. Firstly, in order to deeply explore the temporal and spatial correlation of nodes, the Chebyshev convolution and gated loop unit were combined to obtain a larger receptive field. Secondly, three periodicity datasets with different time window were picked up to provide comprehensive traffic information. Finally, the MSTAGCN model was constructed by fusing multiple spatiotemporal components with encoder-decoder network structure. The experimental results about highway datasets PEMS04 and PEMS08 in Caltrans performance evaluation system show that the performance of the new model is significantly better than other models, and it can be applied to the actual road network to improve traffic prediction precision efficiency. In the next step, datasets about urban road networks will be collected to explore the adaptability of the model under complex urban road networks.

Data Availability

All data and program files included in this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors gratefully acknowledge the support of the National Natural Science Foundation of China (Grant no. NSFC 61602486).

References

[1] C. Han, S. Su, and C. Wang, "Real-time adaptive prediction of short-term traffic flow based on ARIMA model," *Journal of System Simulation*, vol. 16, no. 7, pp. 1530–1532, 2004.

[2] S. V. Kumar, "Traffic flow prediction using kalman filtering technique," *Procedia Engineering*, vol. 187, pp. 582–587, 2017.

[3] Q. Shen, Y. Wang, and Y. Huang, "Grey verhulst-markov model with improved initial value and its application," *Statistics & Decisions*, vol. 36, no. 7, pp. 30–33, 2020.

[4] X. Feng, X. Ling, H. Zheng, Z. Chen, and Y. Xu, "Adaptive multi-kernel SVM with spatial–temporal correlation for short-term traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 6, pp. 2001–2013, 2018.

[5] C. X. Tao and M. Xie, "Short-term traffic flow prediction method based on nonparametric regression of k-nearest neighbors," *Systems Engineering Theory and Practice*, vol. 30, pp. 376–384, 2010.

[6] J. Wang, W. Deng, and J. Zhao, "Short-term traffic flow forecast based on bayesian network multi-method combination," *Transportation Systems Engineering and Information*, vol. 11, no. 4, pp. 147–153, 2011.

[7] H. X. Shao and B. H. Soong, "Traffic flow prediction with long short-term memory networks (lstms)," in *Proceedings of the 2016 IEEE Region 10 Conference*, 2016, Article ID 29862989.

[8] M. Liu, J. Wu, and Y. Wang, "Traffic flow prediction based on deep learning," *Journal of System Simulation*, vol. 30, no. 11, pp. 4100–4105, 2018.

[9] X. J. Shi, Z. R. Chen, H. Wang, Y. Dit-Yan, W. Wai-kin, and W. Wang-chun, "Convolutional Lstm Network: A Machine Learning Approach for Precipitation Now Casting [EB/OL]," 2015, https://arxiv.org/abs/1506.04214.

[10] Y. Liu, H. Zheng, X. Feng, and Z. Chen, "Short-term traffic flow prediction with Conv-LSTM," in *Proceedings of the 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*, pp. 1–6, Nanjing, China, 2017.

[11] H. X. Yao, X. F. Tang, H. Wei, G. Zheng, and Z Li, "Revisiting spatial-temporal similarity: a deep learning framework for traffic prediction," *AAAI*, vol. 33pp. 5668–5675, Honolulu, 2019.

[12] J. Zhang, Y. Zheng, and D. Qi, "Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows prediction," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pp. 1655–1661, AAAI, San Francisco, USA, 2016.

[13] L. Zhao, Y. J. Song, C. Zhang et al., "T-GCN: a temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, 2020.

[14] B. Yu, H. T. Yin, and Z. X. Zhu, "Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, June 2018.

[15] Y. Li, R. Yu, S. Cyrus, and L. Yan, "Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting," in *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, BC, Canada, 2018.

[16] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 922–929, Palo Alto, CA, 2019.

[17] 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 914–921, Palo Alto, CA, 2020.

[18] Y. Wang, Y. Zhang, X. Piao, H. Liu, and K. Zhang, "Traffic data reconstruction via adaptive spatial-temporal correlations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 4, pp. 1531–1543, 2019.

[19] J. Wang, Y. Zhang, Y. Wei, Y. Hu, X. Piao, and B. Yin, "Metro passenger flow prediction via dynamic hypergraph convolution networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 12, pp. 7891–7903, 2021.

[20] Y. Yu, Y. Zhang, S. Qian, S. Wang, Y. Hu, and B. Yin, "A low rank dynamic mode decomposition model for short-term traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, 2021.

[21] A. Vaswani, N. Shazeer, N. Parmar, N. G. Aidan, K. Lukasz, and P. Illia, "Attention is all you need," 2017, https://arxiv.org/abs/1706.03762?context=cs.

Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 8676805, 9 pages
https://doi.org/10.1155/2022/8676805

Research Article

Data Modeling of Impact of Green-Oriented Transportation Planning and Management Measures on the Economic Development of Small- and Medium-Sized Cities

Yuan Lu,[1] Jinyan Shao,[2] and Yifeng Yao [1]

[1]*School of Architecture and Design, Beijing Jiaotong University, Beijing 100044, China*
[2]*Beijing Urban Construction Design and Development Group Co. Limited, Beijing 100044, China*

Correspondence should be addressed to Yifeng Yao; yfyao@bjtu.edu.cn

Received 16 May 2022; Accepted 17 June 2022; Published 11 July 2022

Academic Editor: Yong Zhang

With the rapid growth of urbanization and motorization in China in recent years, the demand for transportation in people's work and daily lives has increased. In this context, a number of issues such as urban traffic congestion, energy consumption, and environmental pollution have become increasingly severe. As a result, the tremendous socioeconomic, resource, and environmental pressures have been placed on the development of urban transportation. Sustainable economic and social development requires green development as a precondition. The economical and efficient use of resources and the protection and improvement of the ecological environment are conducive to the formation of a new pattern of modernization for the harmonious development of man and nature. Transportation planning is an essential technical field for promoting the development of green and ecological cities, and it is one of the primary responsibilities of urban planning. The application of green ecological planning technology and the scientific and reasonable development of traffic planning and management measures can aid in reducing energy consumption and, thus, achieving the goal of environmental protection. In this field, green transportation is a mature green ecological planning technology. Green transportation development is not only a key solution to urban transportation problems, but also an essential means of achieving sustainable urban development, so it has become a hot topic in the field of transportation. As the ideal city of the postindustrial era, ecocity can serve as a model for the sustainable development of China's small and medium-sized cities. After all, industrial development is an unsustainable path, so human society must embrace green development. The core of green development lies in shifting from an exclusive reliance on industrialization to the urban transformation into an ecological civilization. Given the current contradictions between economic growth and resource and environmental degradation, promoting green and environmentally conscious transportation planning with resource conservation in mind is a crucial means of resolving these contradictions. Government incentives and restrictions are essential for the development of green transportation. Therefore, it is crucial to study the impact of environmentally conscious transportation planning and management measures on the economic growth of small and medium-sized cities. This will provide relevant departments and stakeholders with guidance and a reference for formulating policies that will contribute to the harmonious development of China's green transportation and economy.

1. Introduction

Transportation, as a vital lifeline for social development, is a crucial material production sector that ensures the smooth division of labor in the region and promotes national economic development [1]. To be specific, transportation can organically connect the production, distribution, exchange, and consumption of goods. Transportation plays a crucial role in promoting the rational flow of production factors, such as talents, [2] capital, information [3], and resources [4], and in fostering social and economic growth. Small cars have become more and more popular and have taken over as the general public's mode of transportation, particularly in the twenty-first century. However, from the perspective of

protecting the ecological environment, the transportation industry is destructive in terms of both energy consumption and environmental pollution [5]. As a result, as economic development accelerates urbanization and the urban population continues to grow, so does the demand for urban residents to travel. In this context, the increasing number of private cars has led to an increase in transportation energy consumption and greenhouse gas emissions, resulting in urban traffic congestion, air pollution, and urban heat island effect. Therefore, these phenomena pose a number of threats to sustainable urban development [6]. The transportation industry, as an important part of economic development, must take the initiative to assume the social responsibility of achieving a coordinated development between economic growth, environmental protection, and social harmony. Under this development concept, green transportation has emerged [7]. Therefore, green transportation has become a new driving force for economic development, in line with the current social trend.

In response to climate change, many countries around the world have begun to promote green and low-carbon transportation systems [8]. The goal of carbon neutrality also places significant demands on the sustainable development of the transportation sector [9]. As a result of rapid urbanization, China's motor vehicle ownership is increasing, and traffic congestion and air pollution are becoming increasingly prominent. In recent years, the average annual growth rate of carbon emissions in China's transportation sector is more than 5%, making it the fastest-growing sector in terms of greenhouse gas emissions, with total emissions accounting for about 15% of the country's total carbon emissions [10]. As a result, improving the transportation environment and developing green transportation are closely related to the goal of achieving carbon neutrality in China. In recent years, with the rapid development of China's economy and the acceleration of urbanization, the motorization of urban transportation has also expanded rapidly, and the number of urban motor vehicles has increased rapidly [11]. As shown in Figure 1, in 2013, there were only 32.56 million private cars in China, but this number reached 273.46 million in 2021 [12]. Therefore, private car ownership in China continues to grow, which indicates that more and more residents are inclined to use cars [13]. Nevertheless, this trend can not only cause negative impacts on the rapid development of public transportation, but also worsen the urban traffic structure [14]. The increase of urban motor vehicles has brought about a series of severe issues, such as traffic congestion, environmental pollution, accelerated energy consumption of resources, and noise disturbance, which seriously affect the process of sustainable urban development [15].

The rapid expansion of motor vehicles has brought tremendous pressure on urban traffic in China [16]. As a result, many small and medium-sized cities are experiencing serious problems such as traffic congestion. During the daily morning and evening rush hours, there are especially long queues of vehicles [17]. This significantly reduces the efficiency of urban roads and, as a result, increases the likelihood of traffic accidents, which negatively impacts the ability of urban residents to enjoy quality travel services [18]. To meet the growing demand for transportation, cities are expanding their transportation infrastructure. This process takes up nonrenewable resources such as land and construction materials, which is not conducive to sustainable transportation development [19, 20]. At the same time, motor vehicles consume a lot of fossil energy, which is also a major source of air pollution [21]. As a result of the unprecedented socioeconomic and environmental pressure on resources, the mode of urban transportation development now faces greater challenges [22]. The development of urban green transportation is not only an important solution to urban transportation problems, but also an inevitable choice for sustainable urban development and has evolved into a topic of national significance.

With the rapid advancement of economic development and urbanization, the urban traffic travel rate and travel distance increase, resulting in a conflict between economy and traffic demand and between traffic and environment. On the one hand, through systematic analysis of the mechanism of urban green transportation development, it is helpful to grasp the law of urban green transportation development [23]. This will aid in identifying the most influential factors and in proposing effective measures to promote the development of green transportation in cities. On the other hand, by establishing a set of applicable urban green transportation development evaluation index system and studying the level of urban green transportation development based on empirical data, we can quantitatively grasp the development trend of green transportation through evaluation [24]. Thus, the shortcomings and deficiencies can be identified, and relevant departments can be provided with reference bases for taking targeted measures. The evaluation scope of green transportation includes two aspects. The first one is to evaluate the development status of transportation in cities that have been engaged in green transportation construction for a long time [25]. According to the evaluation results, the development level can be determined and corresponding improvement measures and policies can be proposed. What is more, for cities that have only carried out transportation planning without green transportation construction or have been under construction for a short period of time, the planning options are evaluated. Then, according to the evaluation results, we can select and adjust the plan.

In addition, research on the relationship between transportation and economic development has been a topic of intense interest in a variety of disciplines, such as transportation economics, regional economics, and development economics. In the existing literature, most of the studies have been conducted on external urban public transportation such as railroads, roads, and terminals [26]. However, the role of urban public transportation, as the main internal public transportation task of cities, in the process of economic development has received scant attention, especially in terms of empirical analysis [27]. As a driving force of economic transformation and social progress, transportation should adapt to the shifting production, lifestyle, and travel consumption patterns of people. At the same time, giving priority to the development of urban

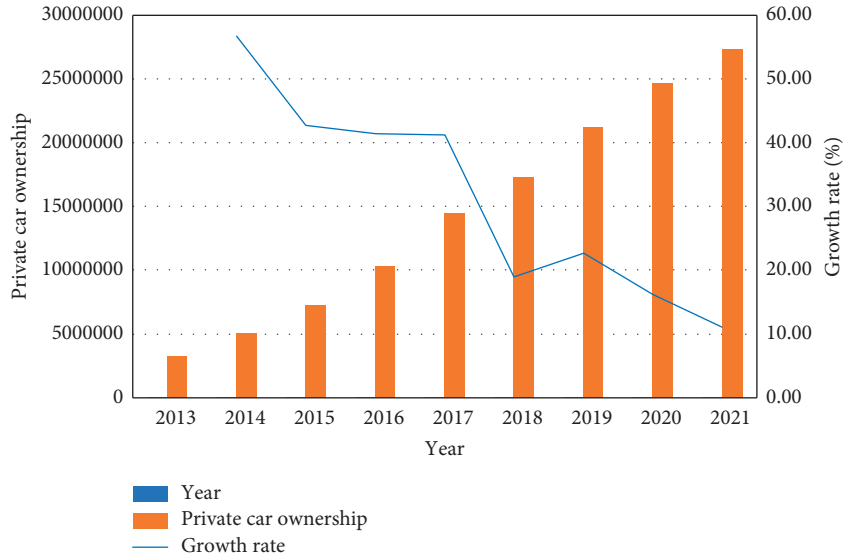


Figure 1: Private car ownership from 2013 to 2021 in China.

public transportation is the way to promote the thorough integration of transportation and economic and social development [28]. As a result, it is of great practical significance to conduct an empirical study on the impact of urban public transportation development on the green economy in conjunction with the current new economic development model of green economy [29].

Currently, major cities devote a significant amount of manpower and resources to the construction of fundamental transportation infrastructure. To be specific, the total area of urban roads and their network capacity are expanding, but the supply-and-demand disparity for roads remains severe [30]. As a result, many small and medium-sized cities in China are experiencing heavy traffic congestion, which drastically reduces the quality of the transportation system's service. In recent winters, major northern cities across the country have been plagued by haze, and air quality has significantly deteriorated everywhere. As a result, many cities have had to implement restrictions on construction site closures and car traffic [31]. However, most of these emergency measures are not effective at treating the symptoms. One of the major sources of urban pollution is traffic pollution. The rapidly increasing number of motor vehicles is exacerbating this situation and is becoming a major source of urban environmental pollution. The fast-paced socioeconomic development requires a green urban transportation system that is in harmony with it [32]. The root cause of urban environmental degradation is closely related to urban transportation planning.

Green transportation is a new type of urban transportation system proposed to improve urban transportation efficiency, promote social equity and stability, save construction and maintenance costs, reduce traffic congestion, and reduce environmental pollution. Due to the different levels of green transportation development in different cities and the uncoordinated development of green transportation subsystems, it is difficult to make the most effective use of green transportation system because of the poor connection between green transportation modes. Also, urban traffic problems have seriously restricted the healthy development of cities. Therefore, how to maintain rapid economic growth while considering ecological stability, to achieve sustainable urban development and sustainable use of transportation resources, is a problem that must be faced and solved in the process of socialist modernization in China. The trend of sustainable development in many fields, such as economy, environment, culture, and society, has led to the birth of green transportation. Green transportation is a new type of urban transportation system that improves the efficiency of urban transportation, reduces traffic congestion, reduces environmental pollution, and promotes social equity and rational use of resources. Many cities, at home and abroad, have conducted extensive research into the planning and construction of green transportation. Therefore, this study takes green transportation as the guide and conducts data modeling on the impact of urban transportation planning and management measures on the economic development of small and medium-sized cities.

2. Green Transportation

2.1. Concept of Green Transportation. Green transportation is an integrated urban transportation system with the goal of safety, convenience, high efficiency, low pollution, and low energy consumption, which is built with advanced technology and is compatible with human living environment and economic growth. In addition, green transportation is an urban transportation mode based on advanced scientific methods and technologies, considering efficiency and fairness and aiming to establish an urban transportation system that gives priority to public transportation. Therefore, it can effectively promote the harmonious development of urban transportation and ecological environment. At the same time, green transportation is a harmonious transportation

system that aims at reducing traffic congestion, reducing environmental pollution, and promoting the rational use of resources to meet the requirements of sustainable development of urban environment, economy, and society. According to the concept of green transportation system and the different degrees of environmental impact brought by various modes of transportation, the green transportation system can be ranked according to the priority of green transportation, as shown in Figure 2.

There is no uniformity in the research on the concept of green transportation so far. Although different scholars have different understandings of the concept of green transportation, they all share the same concept of sustainable development of urban transportation. From the macroperspective, green transportation should meet the sustainable development needs of urban transportation to the greatest extent possible under various unfavorable external conditions. From the microperspective, green transportation should not only meet individual travel needs, but also minimize transportation energy consumption, maximize resource efficiency, and reduce environmental pollution.

In fact, green transportation is composed of two basic elements, "green" and "transportation." Among them, "transportation" can be understood as the mode of transportation, while "green" indicates the way of development and the requirement of quality. The concept of green transportation corresponds to the theoretical basis of economics such as ecological economics, energy economics, and environmental economics. As a result, in the context of deepening conflicts between economic and social development and resources and environment, green transportation can, to a certain extent, promote a shift in the economic development approach to a sustainable development approach. At the same time, more specific research fields of economics have been created, including green economy, circular economy, low-carbon economy, and ecological economy. What is more, the concepts of green transportation, green cycle, low-carbon transportation, and ecotransportation have also been derived. In summary, the relationship between these green concepts is illustrated in Figure 3.

In the field of economics, green economy is an environmentally friendly and healthy economic approach. Therefore, the essence of green economy is a method of economic development characterized by the preservation of the of human living environment, the reasonable protection of resources, and the promotion of human health. In other words, the core of green economy is the harmonious development of ecology and economy. Low carbon economy is an economic development model that reduces high energy consumption and emphasizes low energy consumption, low pollution, and low emissions. Therefore, the essence of a low-carbon economy is the efficient use of energy and the use of clean energy in the continuous pursuit of economic development. At its core is the innovation of energy-saving and emission reduction technologies. Ecological economy follows the principle of "circular economy" and emphasizes the full potential of natural resources reuse within the carrying capacity of the ecosystem, so as to achieve economic development and environmental protection.

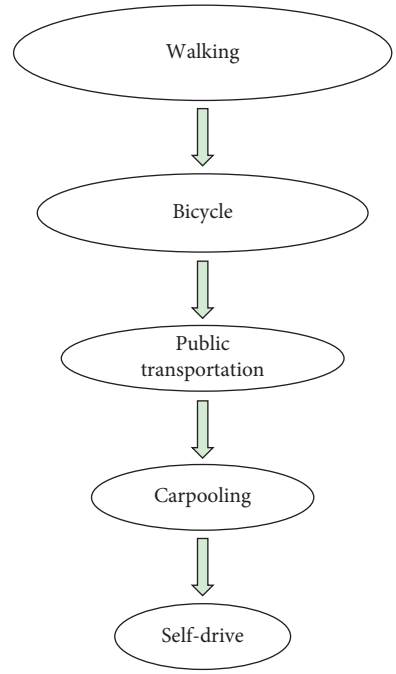


Figure 2: Green transportation system.

2.2. Characteristics of Green Transportation. As a new development concept and the current mainstream development direction of the transportation industry, green transportation is quite different from traditional transportation. First of all, the scope of traditional transportation is much larger than the travel mode of green transportation. In addition, green transportation has a development focus, while traditional transportation does not. For example, green transportation emphasizes and promotes low energy and low pollution travel modes such as walking, bicycling, urban public transportation, and new energy vehicles. However, the development model of traditional transportation is highly arbitrary, as the choice of transportation mode is largely determined by the preferences of individual citizen. In summary, the differences between green transportation and traditional transportation are shown in Table 1.

Green transportation aims to reduce pollution and protect the environment, but it does not restrict the freedom of individuals to travel. On the contrary, the p is not only to achieve the harmonious development of transportation, environment, and economy, but also to make people travel better. First, in the process of developing green transportation, the management and improvement of public transportation can better meet people's demand for quality travel. What is more, in the process of developing green transportation, the government's promotion of green transportation can make citizens aware of the environmental benefits of green transportation. As a result, citizens can willingly choose green transportation. This approach, which ultimately makes citizens willing to choose green transportation, reflects a people-centered approach. Finally, the widespread promotion of green transportation has greatly increased not only the awareness of our citizens to travel green, but also the quality of our citizens. Therefore, the

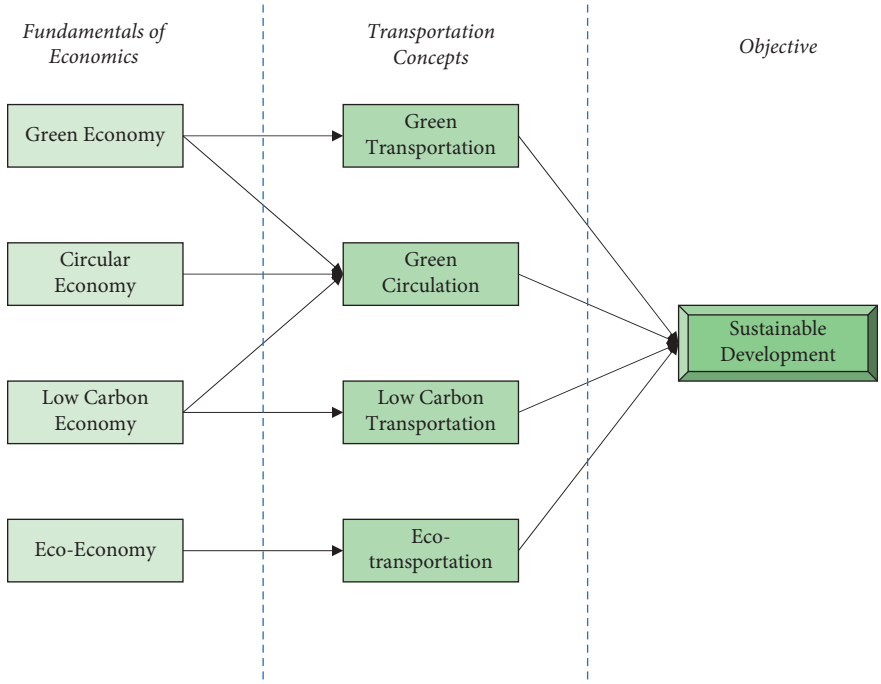


Figure 3: Relationship among concepts related to green economy.

Table 1: Differences between green transportation and traditional transportation.

Development mode	Travel mode	Consumed energy	Amount of consumed energy	Amount of pollution
Traditional transportation	Citizens' choice of travel mode is arbitrary	Oil-based	Much	Much
Green transportation	One of the starting points for citizens' choice of travel mode is energy saving and environmental protection	Clean energy-based	Less	Less

emergence of green transportation is not only a people-oriented approach, but also a win-win situation for the economy, the environment, and the quality of citizens.

3. Economic Effect of the Green Transportation Model

Neoclassical economic theory focuses on physical capital and considers that factors of production move instantaneously between different geographical locations and do not have spillover effects on neighbouring regions. Neo-economic geography, on the other hand, considers the spatial dependence of regional activities and the spatial spillover of capital stocks. For example, product investment or market expansion can lead to costly changes in the income or expenditure of new capital. Changes in a factor of production or an observed attribute in a spatial region can have positive or negative effects on multiple factors in neighboring regions, thus driving the accumulation of capital markets.

3.1. Assumption of the Model. In the economic effect of green transportation model proposed in this research, the following assumptions should be followed:

(1) It is assumed that the impact of domestic force majeure factors on the economic effect of transportation is negligible at the time of the data.

(2) It is assumed that only total sulfur dioxide emissions and total dust emissions are considered as the source of green environmental indicators in the economic effect of green transportation.

(3) It is assumed that roads and railroads occupy an absolute position in the transport infrastructure in the social economy and that other transport infrastructure inputs are not measured.

(4) It is assumed that the spatial section unit is considered as a central point when calculating the spatial section unit distance.

3.2. Autocorrelation Test of Economic Effect of Transportation. The estimation of coefficients becomes complicated when applying the traffic economic effects model regression to analyze the error or lagged terms. This highlights the need for spatial autocorrelation of the model. The description of spatial autocorrelation is reflected in the spatial structure. Therefore, it is not limited to the geographic sense only, where the global correlation statistic only provides a basic

premise and overall description for the spatial autocorrelation of the study, and its correctness is based on the premise of spatial homogeneity.

In determining the spatial correlation of variables among regions, the spatial residual correlation test selected for this study is shown below:

$$\rho = \frac{n}{\sum_{i=1}\sum_{j=1} w_{ij}} \frac{\sum_{i=1}\sum_{j=1} W\left(V_i - \overline{V}\right)\left(V_j - \overline{V}\right)}{\sum_{i=1}\left(V_i - \overline{V}\right)^2}, \quad (1)$$

where ρ refers to the spatial residual correlation, n indicates the number of selected samples, w_{ij} represents the weight matrix, and V_i and V_j refer to the spatial correlation variables.

3.3. Statistical Test of Economic Effect of Transportation. The existing exponential test is a correlation test based on spatial residuals and therefore has poor significance. Therefore, the introduction of Lagrange multiplier statistic with spatial autoregressive effect without spatial residual correlation can be better tested spatially, and its model can be defined as

$$\begin{aligned} H_0&: Y = \alpha \times V + \varepsilon, \\ H_1&: Y = \alpha \times W \times Y + \alpha \times V + \varepsilon. \end{aligned} \quad (2)$$

By constructing the following two LM statistics, this model can be tested and selected. That is, when there is no residual correlation, the model can be tested for the existence of spatial substantive correlation:

$$\begin{aligned} E &= \frac{\left(e'We/m^2\right)^2}{K} \sim \chi^2(1), \\ L &= \frac{\left[e'Wy/\left(ee'/n\right)\right]^2}{Z} \sim \chi^2(1), \end{aligned} \quad (3)$$

where E refers to the LM error, L refers to the LM lag, and

$$\begin{aligned} m^2 &= ee'/n, \\ K &= \left(W^2 + W'W\right)', \\ Z &= \left(W^2 + W'W\right)' + ee'/n. \end{aligned} \quad (4)$$

The LM statistic can only be used to initially judge the model selection by significance. The specific process of spatial autocorrelation test is shown in Figure 4 as a way to determine the final model type.

3.4. Selection of the Model. The spatial social activity network of decision variables is interconnected, and there are mainly the following static spatial models to study the economic effects of transportation (Figure 5). The first one is the spatial autoregressive process of the error term into the traditional effects model of the spatial error model (SEM). The second one is the spatial lag model (SLM), in which the spatial lag term of the explanatory variable is added to the traditional effect model. The third one is the spatial Durbin model (SDM), which means that the spatial lag term of the explanatory variable is added to the SLM to circumvent the endogeneity problem.

3.4.1. Spatial Error Model. The spatial error model implies that the spatial effect between regions is realized through the error term; that is, the economic effect of transportation between regions is stochastic, and the model can be expressed as

$$\begin{aligned} y_{ij} &= \beta \times x_{ij} + \gamma \times c_n + \varepsilon, \\ \varepsilon &= \alpha \times W_\varepsilon + \mu, \\ \mu &\sim N\left(0, \sigma^2 I\right), \end{aligned} \quad (5)$$

where y_{ij} is the explained variable, x_{ij} refers to the explanatory variable, γ indicates the coefficient of the constant term c_n, ε is the error term, and W_ε represents the spatial lag term.

3.4.2. Spatial Lag Model. In the spatial lag model, the explanatory variables in the neighboring regions affect the regions in the system through spatial radiation spillover. Since the SLM model includes the lagged term of the explanatory variables when analyzing the economic effects of transportation, it can be named as a spatial autoregressive model, and its model can be expressed as

$$\begin{aligned} y_{ij} &= \rho \times W \times y_{ij} + \beta \times x_{ij} + \gamma \times c_n + \varepsilon, \\ \varepsilon &\sim N\left(0, \sigma^2 I\right), \end{aligned} \quad (6)$$

where ρ refers to the autoregressive coefficient and $W \times y_{ij}$ denotes the spatial interaction of the proximity region on the explanatory variables of the observed region.

3.4.3. Spatial Durbin Model. When the endogenous interaction effect and the autocorrelated perturbation term cannot reasonably explain the spatial action, a more generalized spatial Durbin model is introduced, incorporating both a spatial error term and a spatial lag model, whose model can be expressed as

$$\begin{aligned} y_{ij} &= \rho \times W \times y_{ij} + \beta \times x_{ij} + \eta \times W \times x_{ij} + \gamma \times c_n + \varepsilon, \\ \varepsilon &\sim N\left(0, \sigma^2 I\right). \end{aligned} \quad (7)$$

In the spatial lagged and spatial Durbin models, the explanatory and explained variables appear spatially correlated. In view of the above theories, this paper applies MLE for effect analysis to the measurement of spatial panel data.

3.5. Case Study. Given the impact of feedback effects on the transport infrastructure stock indicators, especially their first-order lagged term regression coefficients, Table 2 looks at the short-term effects and long-term effects, respectively. Specifically, this study deeply explores the decomposition of green transportation economic effects of provincial transportation infrastructure stock indicators in China through the effect decomposition of dynamic spatial Durbin model.

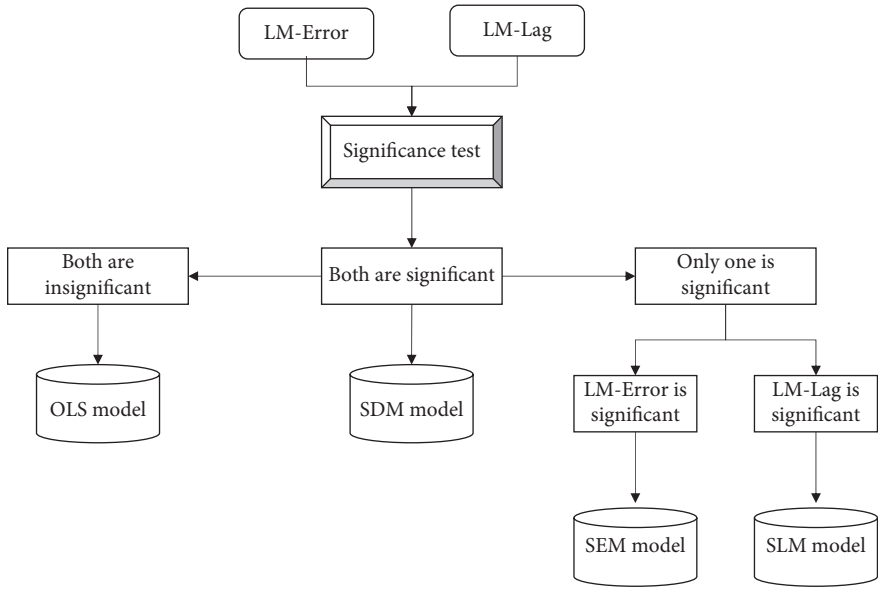


Figure 4: Process of selecting model.

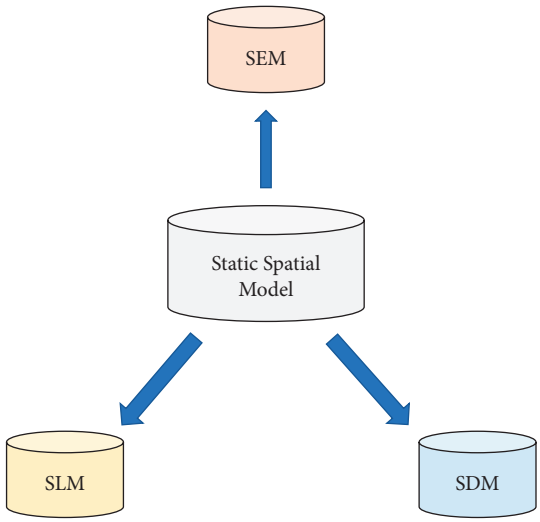


Figure 5: Various static spatial models.

Table 2: Result of spatial Durbin model.

	Road	Highway	Railway	GDP	Population
W_1	0.05	46.72	98.71	0.02	0.07
W_2	−0.48	123.21	431.82	0.38	0.14
W_3	0.12	47.78	1.23	0.16	0.06
W_4	0.03	66.89	12.56	0.67	0.01

The effect decomposition values of the stock indicators verify the robustness of the coefficient estimation results. The significance level of each weight matrix under fixed effects is higher than that of random effects, and the transport infrastructure stock indicators are significant under both the adjacency matrix (W_1) and the economic matrix (W_4), which verifies the validity of the coefficient estimation and effect decomposition measures. By comparing the short-term and long-term effects of the stock indicators under each weight matrix, it is clear that the short-term effects of most of the explanatory variables are more significant at the 1% significance level, and the economic matrix is more significant. Therefore, this indicates that, under the global development situation of green transportation and the policy guidance of China's strong transportation country, the long-term effect of transportation infrastructure investment in the green economic growth of each province is poor. In the process of physical capital accumulation, the indirect effect is obvious under each weight matrix, and the contribution of transportation infrastructure stock indicators gradually increases in prominence under the economic weight. This verifies that the green transportation economic effect of transportation infrastructure inputs is inherent as a direct carrier of production factors circulation in the capital market and regional network structure.

4. Conclusion

The results of this study show that the green transportation economic effects of various types of transportation infrastructure inputs are uneven. Specifically, the trend of transportation infrastructure inputs contributing to green total factor productivity growth in the region is slowing down. In the process of investment, the negative effect of pollution emission of various transportation infrastructure modes on green transportation economy is gradually increasing. Therefore, the green transportation economy effect of transportation infrastructure investment should be considered as a priority in the early stage of transportation planning, increasing, at the same time, the density of highway and railroad. The positive externality of the green transportation economic effect should be fully incorporated into transportation infrastructure investment. In addition, this paper defines green transportation as an urban

transportation system that uses new energy and energy-efficient means of transportation for the purpose of achieving sustainable social development. As a result, the basic requirement of green transportation is to achieve the long-term development of urban transportation. The essence of green transportation is to establish a sustainable transportation system that can meet the transportation needs of residents while saving energy and protecting the ecological environment. In other words, green transportation can meet the largest social transportation needs with the least energy consumption and environmental impact and achieve the coordinated development of transportation, environment, and economy.

The study of urban green transportation development is a systematic project. Due to time constraints and personal knowledge accumulation, this paper only focuses on the economic impact of urban green transportation. Therefore, the research is not deep enough, and there are still some problems that need further research. The matching degree of the weight matrix selection in this paper needs to be verified in depth. If possible, all types of spatial weight matrices should be experimented in the future to select the optimal spatial weight matrix.

Data Availability

The labeled data set used to support the findings of this study is available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by National Natural Science Foundation of China (no. 51908028).

References

[1] R. H. M. Pereira, T. Schwanen, and D. Banister, "Distributive justice and equity in transportation," *Transport Reviews*, vol. 37, no. 2, pp. 170–191, 2017.

[2] M. Grazia Speranza, "Trends in transportation and logistics," *European Journal of Operational Research*, vol. 264, no. 3, pp. 830–836, 2018.

[3] G. Marsden and L. Reardon, "Questions of governance: rethinking the study of transportation policy," *Transportation Research Part A: Policy and Practice*, vol. 101, pp. 238–251, 2017.

[4] C. Cleophas, C. Cottrill, J. F. Ehmke, and K. Tierney, "Collaborative urban transportation: recent advances in theory and practice," *European Journal of Operational Research*, vol. 273, no. 3, pp. 801–816, 2019.

[5] J. Guerrero-Ibáñez, S. Zeadally, and J. Contreras-Castillo, "Sensor technologies for intelligent transportation systems," *Sensors*, vol. 18, no. 4, p. 1212, 2018.

[6] Y. V. Fan, S. Perry, J. J. Klemeš, and C. T. Lee, "A review on air emissions assessment: Transportation," *Journal of Cleaner Production*, vol. 194, pp. 673–684, 2018.

[7] L. Zhu, F. R. Yu, Y. Wang, B. Ning, and T. Tang, "Big data analytics in intelligent transportation systems: a survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 383–398, 2019.

[8] C. Guan, S. Srinivasan, and C. P. Nielsen, "Does neighborhood form influence low-carbon transportation in China?" *Transportation Research Part D: Transport and Environment*, vol. 67, pp. 406–420, 2019.

[9] Y. Zhou, W. Fang, M. Li, and W. Liu, "Exploring the impacts of a low-carbon policy instrument: a case of carbon tax on transportation in China," *Resources, Conservation and Recycling*, vol. 139, pp. 307–314, 2018.

[10] O. Lah, "Decarbonizing the transportation sector: policy options, synergies, and institutions to deliver on a low-carbon stabilization pathway," *Wiley Interdisciplinary Reviews: Energy & Environment*, vol. 6, no. 6, p. e257, 2017.

[11] H. Xing, C. Stuart, S. Spence, and H. Chen, "Alternative fuel options for low carbon maritime transportation: pathways to 2050," *Journal of Cleaner Production*, vol. 297, p. 126651, 2021.

[12] Y. Gan, Z. Lu, H. Cai, M. Wang, X. He, and S. Przesmitzki, "Future private car stock in China: current growth pattern and effects of car sales restriction," *Mitigation and Adaptation Strategies for Global Change*, vol. 25, no. 3, pp. 289–306, 2020.

[13] Y. Jiang, P. Gu, Y. Chen, D. He, and Q. Mao, "Influence of land use and street characteristics on car ownership and use: evidence from Jinan, China," *Transportation Research Part D: Transport and Environment*, vol. 52, pp. 518–534, 2017.

[14] P. Zhao and Y. Bai, "The gap between and determinants of growth in car ownership in urban and rural areas of China: a longitudinal data case study," *Journal of Transport Geography*, vol. 79, p. 102487, 2019.

[15] Y. Ao, D. Yang, C. Chen, and Y. Wang, "Exploring the effects of the rural built environment on household car ownership after controlling for preference and attitude: evidence from Sichuan, China," *Journal of Transport Geography*, vol. 74, pp. 24–36, 2019.

[16] Y. Hui, Y. Wang, Q. Sun, and L. Tang, "The impact of car-sharing on the willingness to postpone a car purchase: a case study in Hangzhou, China," *Journal of Advanced Transportation*, vol. 2019, p. 9348496, 2019.

[17] Y. Zhang, C. Li, Q. E. Liu, and W. Wu, "The socioeconomic characteristics, urban built environment and household car ownership in a rapidly growing city: evidence from Zhongshan, China," *Journal of Asian Architecture and Building Engineering*, vol. 17, no. 1, pp. 133–140, 2018.

[18] S. Wang, C. Shi, C. Fang, and K. Feng, "Examining the spatial variations of determinants of energy-related CO2 emissions in China at the city level using Geographically Weighted Regression Model," *Applied Energy*, vol. 235, pp. 95–105, 2019.

[19] B. Cheng, C. Fan, H. Fu, J. Huang, H. Chen, and X. Luo, "Measuring and computing cognitive statuses of construction workers based on electroencephalogram: a critical review," *IEEE Transactions on Computational Social Systems*, pp. 1–16, 2022.

[20] B. Cheng, J. Huang, J. Li, S. Chen, and H. Chen, "Improving contractors' participation of resource utilization in construction and demolition waste through government incentives and punishments," *Environmental Management*, vol. 225, pp. 1–15, 2022.

[21] B. Cheng, K. Lu, J. Li, H. Chen, X. Luo, and M. Shafique, "Comprehensive assessment of embodied environmental impacts of buildings using normalized environmental impact factors," *Journal of Cleaner Production*, vol. 334, p. 130083, 2022.

[22] X. Huang, X. J. Cao, J. Yin, and X. Cao, "Effects of metro transit on the ownership of mobility instruments in Xi'an, China," *Transportation Research Part D: Transport and Environment*, vol. 52, pp. 495–505, 2017.

[23] F. Xia, A. Rahim, X. Kong, M. Wang, Y. Cai, and J. Wang, "Modeling and analysis of large-scale urban mobility for green transportation," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1469–1481, 2018.

[24] M. Lu, R. Xie, P. Chen, Y. Zou, and J. Tang, "Green transportation and logistics performance: an improved composite index," *Sustainability*, vol. 11, no. 10, p. 2976, 2019.

[25] N. Chen and C. H. Wang, "Does green transportation promote accessibility for equity in medium-size US cites?" *Transportation Research Part D: Transport and Environment*, vol. 84, p. 102365, 2020.

[26] T. Mayer and C. Trevien, "The impact of urban public transportation evidence from the Paris region," *Journal of Urban Economics*, vol. 102, pp. 1–21, 2017.

[27] A. Galkin, Y. Davidich, Y. Kush, N. Davidich, and I. Tkachenko, "Improving of urban public transportation quality via operator schedule optimization," *Journal of Urban and Environmental Engineering*, vol. 13, no. 1, pp. 23–33, 2019.

[28] U. Kuvvetli and A. R. Firuzan, "Applying Six Sigma in urban public transportation to reduce traffic accidents involving municipality buses," *Total Quality Management and Business Excellence*, vol. 30, no. 1-2, pp. 82–107, 2019.

[29] F. Ortenzi, M. Pasquali, P. P. Prosini, A. Lidozzi, and M. Di Benedetto, "Design and validation of ultra-fast charging infrastructures based on supercapacitors for urban public transportation applications," *Energies*, vol. 12, no. 12, p. 2348, 2019.

[30] A. Psaltoglou and E. Calle, "Enhanced connectivity index–A new measure for identifying critical points in urban public transportation networks," *International Journal of Critical Infrastructure Protection*, vol. 21, pp. 22–32, 2018.

[31] R. Yuan, F. Guo, Y. Qian et al., "A system dynamic model for simulating the potential of prefabrication on construction waste reduction," *Environmental Science and Pollution Research*, vol. 29, no. 9, pp. 12589–12600, 2022.

[32] D. Poudel, "Management of cooperatives focusing on Asta-Ja and globalization," *Articles*, vol. 21, no. 1, pp. 77–84, 2019.

Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 4672617, 8 pages
https://doi.org/10.1155/2022/4672617

Research Article

Rail Transit Prediction Based on Multi-View Graph Attention Networks

Li Wang, Xin Wang, and Jiao Wang

School of Computer Science, The Open University of China, Beijing 100039, China

Correspondence should be addressed to Li Wang; wlpolo@ouchn.edu.cn

Received 23 February 2022; Revised 16 May 2022; Accepted 24 May 2022; Published 6 July 2022

Academic Editor: Yong Zhang

Traffic prediction is the cornerstone of intelligent transportation system. In recent years, graph neural network has become the mainstream traffic prediction method due to its excellent processing ability of unstructured data. However, the network relationship in the real world is more complex. Multiple nodes and various associations such as different types of stations and lines in rail transit always exist at the same time. In an end-to-end model, the training accuracy will suffer if the same weights are assigned to multiple views. Thus, this paper proposes a framework with multi-view and multi-layer attention, which aims to solve the problem of node prediction involving multiple relationships. Specifically, the proposed model maps multiple relationships into multiple views. A graph convolutional neural network of multiple views with multi-layer attention learns the optimal regression of nodes. Furthermore, the model uses an autoencoder module to alleviate the over-smoothing problem during the training phase. With the historical dataset of Beijing rail transit, the experiment proves that the prediction accuracy of the model is generally better than the baseline traffic prediction algorithms.

1. Introduction

As the core function of the intelligent transportation system, traffic forecasting has practical significance for the actual needs of intelligent command and dispatch, traffic planning and layout, and public travel convenience. The prediction of passenger flow in and out of rail transit stations is one of the research hotspots in the field of smart transportation. An accurate passenger flow prediction method will be beneficial to the transportation system for reasonable route scheduling, road network design, crowd evacuation adjustment, and other specific applications. Most of the previous studies have focused on methods based on mathematical modeling as well as machine learning. However, in terms of rail transit, due to the unique topological structure of rail transit and the travel patterns of passengers, it is difficult to obtain efficient and accurate prediction results with the simple application of traditional methods, and related research is relatively limited.

In recent years, graph convolutional neural networks have achieved excellent performance in the field of traffic prediction by virtue of their excellent processing capabilities for non-Euclidean data. In fact, networks are ubiquitous in the real world, such as transportation networks, social networks, and recommendation networks. By modeling the network as a graph, subsequent prediction tasks can be performed. The graph-based non-Euclidean topology not only describes the connection relationship between stations, but also constrains the flow path of data. Therefore, the nongraph method can only make predictions for each station and average the prediction results, and cannot make full use of the topology of rail transit.

However, node relationships in the real world are more complex and contain many types of interrelated relationships. A view could represent a certain relationship. However, the node relationship information will be lost to an extent if only a single view is used for representation [1]. Multiple views can more accurately model different types of relationships, thereby ensuring that the model retains more comprehensive node information, which in turn enables more accurate node-level predictions. In rail transit, structurally, different types of lines and stations can be

assigned to different view features. On the other hand, from the perspective of traffic flow characteristics, the pattern of passenger travel in different time spans can be viewed as different spatial-temporal features [2]. However, when the model contains multiple node relationships at the same time, how to ensure that the model integrates different node relationships with optimal weights to achieve more accurate prediction becomes a key issue.

Since the same node has a different importance in different views, the relationships between nodes in different views should be given different weights. Conversely, the same weights will negatively affect the final prediction and weaken the meaning of the information provided by multiple views. Therefore, we design a multi-layer attention mechanism to achieve weight optimization for different views. In addition, during the training of the graph neural network, the problem of over-smoothing significantly affects the training effect as the number of network layers deepens. That is, the hidden layer representation of each node converges to the same value during the training process of the graph neural network, which eventually leads to poor training results.

In response to the above problems, we propose a traffic prediction model based on multi-view graph attention network (MV-GAT), and its main contributions can be summarized as follows:

(1) An end-to-end rail passenger flow prediction model is proposed. The proposed model achieves fine-grained multi-view modeling for rail transit characteristics at the input and node-level prediction at the output.

(2) Through the multi-layer attention module, the proposed model can assign different weights to different nodes and relationships within multiple views, thereby learning the optimal regression of nodes.

(3) In addition, the self-encoder module transfers the latent information captured by each layer of the self-encoder to the corresponding graph convolution layer, ensuring the validity of the structural information of each layer in the network, and further improving the effect of node prediction.

The model is evaluated through experiments on the Beijing rail transit historical dataset, and the superiority of the model is verified by comparison with existing models. Furthermore, multi-view and multi-layer attention have good interpretability, as shown in ablation experiments.

2. Related Work

The research content of this paper mainly involves graph convolutional neural network and graph attention mechanism.

2.1. Graph Convolutional Networks. Graph convolutional networks (GCNs) are currently used in many domains such as traffic prediction [3], recommender systems [4, 5], and traffic situation analysis [6]. On graphs, its tasks include graph classification [7], node classification [8], link prediction [9, 10], and graph pooling [11]. GCNs have different kernels that learn node embeddings to be applied to downstream tasks. For example, DeepWalk [12] and node2vec [13] are both random walk-based methods. The model SDNE [14] uses autoencoders to maintain the proximity of first- and second-order networks, using highly nonlinear functions to obtain embeddings. Existing traffic flow forecasting techniques include traditional mathematical modeling methods, such as ARIMA [15], as well as deep learning methods. Among them, deep learning methods are subdivided into nongraph-based methods, such as LSTM [16], and nongraph-based methods, such as GCN models. Traditional mathematical modeling methods as well as nongraph-based methods do not consider the topology of the graph and can only make individual predictions for individual sites. Deep learning methods based on graphs can achieve node-level prediction, but currently the mainstream methods are mainly single view [17].

Single-view graph neural networks contain only one relationship between nodes [18]. Although single view has many advantages, such as easy to understand and easy to design neural network models, it is difficult to accurately capture the complex relationships between nodes, which play a crucial role in the effectiveness of information transfer and problem solving [19]. It has been pointed out that graph data possess similarity information between different nodes, which in turn has been proposed to preserve similarity information in the hidden layer of graph convolutional neural networks [20]. However, these methods rarely exploit the multi-view prediction in end-to-end network models.

2.2. Graph Attention Mechanism. The attention mechanism was first proposed for natural language processing and has now been widely used for many sequence-related tasks. The advantage of the attention mechanism is that it can amplify the impact of important parts of a sequence, and the introduction of the attention mechanism also facilitates the use of graph neural networks. Because graph convolutional networks rely on the eigenvalues of the Laplacian matrix, it is difficult to extract convolutional operations from the overall static graph structure. In an attention network, the output at a given moment depends on the attention it allocates across multiple inputs, i.e., the learning weight assigned to each part of the input, with larger weights implying the output of the pair at that particular moment.

As the attention mechanism in the seq2seq model [21], each output is affected by the different weights assigned to the different inputs. The concept of hard attention [22] is designed as a stochastic process that uses Monte Carlo sampling methods to estimate the gradient of the module, thus enabling back-propagation of the gradient. In addition, attention mechanisms include global attention and local attention [23], as well as multi-headed attention [24]. Multi-headed attention is used to extract features more comprehensively by mapping node representations into multiple node representations through linear mapping and

combining the computational results. Inspired by the above work, the possibility of using a multi-layer attention mechanism to fuse multi-view information to reveal the deep relationships between nodes becomes one of our considerations.

3. Methodology

The necessary preliminaries are firstly illustrated, followed by introducing of the overall architecture of the proposed model, and then the details of each component are elaborated.

3.1. Preliminaries. This section will introduce some concepts and symbols used in this paper. For a regular graph G with vertex set V, the edge set E and weight W can be denoted as $G = (V; E = (e_i)_{i\in I}; W)$. For an undirected graph, the incidence matrix $H \in \mathbb{R}^{N\times I}$ can be defined as

$$h(v,e) = \begin{cases} 1, & \text{if } v \in e, \\ 0, & \text{if } v \notin e. \end{cases} \tag{1}$$

For the vertices in graph, the degree is defined as the sum of all weights connected to the vertices; for the edges in graph, the degree is defined as the total number of vertices connected by the edge:

$$\begin{cases} d(v) = \sum_{e\in E} w(e_i)h(v,e_i), & i \in I, \\ \delta(e) = \sum_{v\in V} h(v,e_i), & i \in I. \end{cases} \tag{2}$$

In the process of modeling information in real life, usually only a single view is used to represent the relationships between nodes. A single view contains only one relationship, but due to the complex relationship in real life, it is difficult to capture the comprehensive node relationship with only one view, which will inevitably lead to the omission of information, which will lead to deviations in the subsequent processing of the model. A multi-view contains various relationships between nodes. It can capture structural information more accurately than a single view and better discover implicit relationships between nodes.

Thus, a multi-view graph can be denoted as $G = (V, E^{(1)}, E^{(2)}, \ldots, E^{(m)}, X)$, which $V = \{v_i\}_{i=1}^{n}$ represents the set of nodes in the graph. $e_{i,j}^{(m)} \in E^{(m)}$ indicates the m-th view, node i is connected to node j, and $x_i \in X$ denotes node feature v_i. The node structure in Graph G can be represented by multiple adjacency matrices $\{A^{(m)}\}_{m=1}^{M}$; if $e_{i,j}^{(m)} \in E^{(m)}$, then $a_{i,j}^{(m)} = 1$; otherwise $a_{i,j}^{(m)} = 0$. In our work, the connection between the node and itself is not considered, i.e., $a_{i,i}^{(m)} = 0$.

The purpose of the work is to predict traffic flow with the proposed model. The input of the model is the historical transit flow data $\mathbf{X}^t = (x_1^t, x_2^t, \ldots, x_N^t) \in \mathbb{R}^{N\times C\times T}$, where N indicates the total number of vertices, C is the number of channels of the feature, and T is the time dimension. At the output end of the proposed model, node-level prediction results are supposed to be obtained, which can be denoted as $\mathbf{Y}^{\mathrm{t+m}} = (y_1^{t+m}, y_2^{t+m}, \ldots, y_N^{t+m}) \in \mathbb{R}^N$.

3.2. MV-GAT: The Proposed Model. For complex relationships between entities in the real world, it is difficult to fully grasp the node structure information if only a single view is used to represent the node relationships. In rail transit, considering only the line connections between stations ignores the relationships between stations at the feature level, such as the OD characteristics of passenger trips between stations with different time spans. During the morning and evening peak hours, large passenger trips show relatively fixed patterns, which can also be used as a view for traffic flow prediction. At the same time, it is important to avoid the problem of premature model fitting as the number of layers of the network model increases. When the model uses multiple views as input, how to fuse these views becomes a new problem. The fusion process must ensure that the model can ignore noisy information and that the most relevant information of the nodes is extracted among the multiple views.

To address the above issues, we propose the overall framework of the model, as shown in Figure 1. The basic idea is to use the multi-layer attention module to capture the node information contained in the multi-view to ensure that the best node representation can be learned, and to use the autoencoder module to ensure that the model learns the structural information between the data, which is represented as a multi-view graph.

In the multi-view module, multiple views are used to ensure complete information extraction. Specifically, in this forecasting task, the multiple views include a static view based on the connectivity of tracks and routes, and an OD view of passenger flows for three different time spans: hourly, daily, and weekly.

The autoencoder module learns the accurate data representation and mitigates the over-smoothing problem. The two parts of the input are connected to the autoencoder module and the GCN module, and each layer in the autoencoder module is guaranteed to be connected to the corresponding GCN layer, so that the structural information between nodes learned in the autoencoder can be integrated into the GCN module.

In the multi-layer attention module, multi-layer attention is used to fuse the multi-view information to obtain an optimal representation of the data. The multi-layer attention module ensures that the model learns different weights at different nodes and in different views.

3.3. Multi-View Graph Convolution. For the single-view graph, the input is $G_k = (A_k, X)$. The multi-view graphs generated by the relationship between the nodes are $G_m = (A_m, X_{att}^{(m)})$, where m is the number of views. Each input is fed into an exclusive convolution module. The output of the convolution is Z_k and Z_m. Take Z_k; for example, the output of the l-th layer of the graph convolution can be expressed as

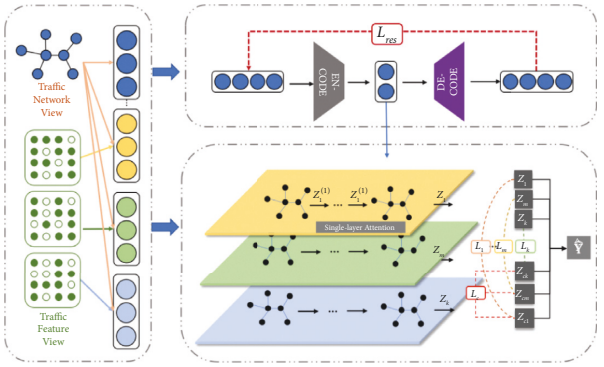


FIGURE 1: The end-to-end framework of proposed model. The MV-GAT model includes multi-view input, multi-layer attention module, and the autoencoder module. The node-level prediction results are obtained as output.

$$Z_m^{(l)} = ReLU\left(\tilde{D}^{-1/2}\widetilde{A_m}\tilde{D}^{-1/2}Z^{(l-1)}W^{(l)}\right), \tag{3}$$

where $W^{(l)}$ is the weight matrix of GCN at the l-th layer, the preliminary $Z_m^{(0)} = X_{att}^{(m)}$, and $X_{att}^{(m)}$ is the node embedding learned by single-view attention network in view m. $Z_m^{(0)} = A_m$. $\widetilde{A_m} = A_m + I$, and $\tilde{D}$ is the diagonal matrix of $\tilde{A}$.

It is difficult for multi-view convolution to learn the commonality between different views only by learning each view individually, so multi-view convolution is supposed to be added to extract common information between different views. The proposed model uses previously constructed input graphs G_k and G_m as inputs to multi-view convolution, the output of multi-view convolution module is Z_c, and the output of the l-th layer of the convolution can be expressed as

$$Z_c^{(l)} = ReLU\left(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}Z^{(l-1)}W^{(l)}\right), \tag{4}$$

where $W^{(l)}$ is the weight matrix of the l-th layer of GCN, the preliminary Z is $Z^{(0)} = X$, $\tilde{A} = A + I$, and $\tilde{D}$ is the diagonal matrix of $\tilde{A}$.

3.4. Autoencoder Module. The proposed method introduces an autoencoder to learn the structural information of the data and pass the learned information to the corresponding GCN layers, and the added autoencoder module also helps to alleviate the over-smoothing problem of the GCN.

Assuming that the autoencoder has L layers, the expression learned in the l-th layer in the autoencoder is $H^{(l)}$:

$$H^{(l)} = ReLU\left(W_e^{(l)}H^{(l-1)} + b_e^{(l)}\right). \tag{5}$$

In the formula, *ReLU* is the activation function of the fully connected layer, and $W_e^{(l)}$ and $b_e^{(l)}$ are the weight matrix and bias of the l-th layer in the autoencoder. In addition, $H^{(0)}$ is the feature matrix X. Then, the input data of decoding part are reconstructed through the fully connected layer.

$$H^{(l)} = ReLU\left(W_d^{(l)}H^{(l-1)} + b_d^{(l)}\right). \tag{6}$$

Here, $W_d^{(l)}$ and $b_d^{(l)}$ are the weight matrix and bias of the l-th layer of decoder. In order to pass the node representation into the GCN module, the node representations are learned from the autoencoder, such as $H^{(1)}, H^{(2)}, \ldots H^{(L)}$. After being passed into the GCN module, the GCN can hold two different kinds of information, the data itself and the data structure. For example, the output of l-th layer learned in the single view can be expressed as $Z_k^{(l)}$.

The representation $H^{(l)}$ learned by the autoencoder can reconstruct the data itself and contains a different valuable information. Combining the two representations leads to a more complete representation.

$$\widetilde{Z}_k^{(l-1)} = (1-\epsilon)Z_k^{(l-1)} + \epsilon H^{(l-1)}. \tag{7}$$

Here, ϵ is the balance coefficient with an initial setting of 0.5. In this way, the autoencoder and GCN can be connected layer by layer. We use ReLU as the activation function to solve the gradient vanishing problem.

3.5. Multi-Layer Attention. Since the model takes multiple views as input, the proposed method designs a multi-layer attention module to effectively integrate the node representations learned in different views to form an optimal combination. First, the proposed method uses a single-view attention layer to learn the influence of different neighbor nodes on the predicted node in the same view. Then, a multi-view attention layer is used to learn the influence of different views on the predicted node. Finally, the two parts are combined to obtain the optimal weighted combination of the nodes to be predicted.

In the single-view attention layer, the influence of different neighbor nodes on the predicted node in each view can be learned. Since each node plays a different role in the process of node embedding, the impact on the final node prediction result is also different. Self-attention is thereby used to learn the weights between each node. For instance, in the view m, calculating the attention index of a pair of nodes (i, j) can be formulated as

$$e_{ij}^{(m)} = att\left(x_i, x_j\right). \tag{8}$$

Here, *att* represents the attention mechanism, and since the multiple views are undirected graphs, the importance of node i to node j is the same as node j to node i. Therefore, $e_{ij}^{(m)}$ is a symmetric matrix.

After calculating the $e_{ij}^{(m)}$ of node j, the weight coefficient is normalized as

$$\begin{aligned}\alpha_{ij}^{(m)} &= \mathrm{softmax}_j\left(e_{ij}^{(m)}\right)\\ &= \frac{\exp\left(\mathrm{LeakyReLU}\left(a_m^T\cdot\left[x_i\|x_j\right]\right)\right)}{\sum_{k\in N}\exp\left(\mathrm{LeakyReLU}\left(a_m^T\cdot\left[x_i\|x_k\right]\right)\right)}.\end{aligned} \tag{9}$$

In the equation, $\|$ represents the connection operation, and a_m^T is the attention vector in the single view. The node embedding of node i in the view can be obtained by the feature aggregation of neighbor nodes with feature

coefficients. Multi-head attention is utilized in order to make the training process more stable. Softmax and ReLU are both activate functions. Specifically, the single-view attention layer repeats K times and connects the learned embedding to a specific view. The learned node embedding and feature matrix are spliced to get $X_{att}^{(m)}$. In the following equation, $z_i^{(m)}$ is the embedding of node i learned in the view m.

$$z_i^{(m)} = \Big\Vert_{k=1}^{K} \text{Sigmoid}\left(\sum_{j \in N} \alpha_{ij}^{(m)} \cdot x_j \right). \tag{10}$$

A single view contains only one type of relationship between nodes, while a multi-view contains relationships between different nodes. To learn more comprehensive node embeddings, it is necessary to integrate multiple node embeddings learned from different views. For different nodes or associations, the weights assigned to different views are different, so it is necessary to design a multi-view attention layer that automatically assigns different weights to different views to solve this problem.

The input of multi-view attention layer is the single-view graph convolution Z_k and $Z_{(m)}$ and the multi-view convolution Z_c, and the attention mechanism $att(Z_k, Z_{(m)}, Z_c)$ learns the weights corresponding to different views $(\alpha_k, \alpha_{(m)}, \alpha_c)$:

$$\left(\alpha_k, \alpha_{(m)}, \alpha_c\right) = att\left(Z_k, Z_{(m)}, Z_c\right). \tag{11}$$

Here, $\alpha_k, \alpha_{(m)}, \alpha_c$ are the attention weights of different views, respectively. For node i, a nonlinear transformation is applied on the node embedding, and then the shared attention vector q is taken to calculate the attention value ω_m^i.

$$\omega_m^i = q^T \cdot \tanh\left(W \cdot \left(z_m^i\right)^T + b\right). \tag{12}$$

The W is weight matrix and b is bias. The attention index of node i in other embedding matrices can be obtained in the same way. Then, the final weight can be calculated by normalizing multiple attention values.

$$\begin{aligned} \alpha_m^i &= \text{softmax}\left(\omega_m^i\right) \\ &= \frac{\exp\left(\omega_m^i\right)}{\exp\left(\omega_m^i\right) + \exp\left(\omega_m^i\right) + \exp\left(\omega_m^i\right)}. \end{aligned} \tag{13}$$

The multiple embeddings are then linearly combined. The larger the α_m^i, the more important the view is.

$$Z = \alpha_k \cdot Z_k + \alpha_{(m)} \cdot Z_{(m)} + \alpha_c \cdot Z_c. \tag{14}$$

The above multi-view attention module solves the problem of assigning different weights to the views, thereby enabling adaptive inter-view importance learning.

3.6. Objective Function. In order to allow the convolution to capture richer information, we increase the difference between Z_k, Z_m, Z_c. Here, we take advantage of the Hilbert–Schmidt independence criterion (HSIC) to measure the independence between the outputs:

$$\text{HSIC}\left(Z_i, Z_j\right) = (n-1)^{-2} tr\left(RK_iRK_j\right). \tag{15}$$

Here, K_iK_j is the Gram matrix, $k_{i,ij} = k_i(z_i^i, z_i^j)$, $k_{j,ij} = k_j(z_j^i, z_j^j)$. And $R = I - 1/nee^T$. I is the identity matrix, and e is the corresponding identity column vector. In the same way, all other views are also calculated by HSIC, denoted as L_s.

The multi-view loss function is supposed to learn as much consistency between different views as possible. After normalizing the matrices $\{Z_{ci}\}_{i=1}^4$ to $\{Z_{\text{cinor}}\}_{i=1}^4$ with L2 normalization, the similarity between nodes $\{S_i\}_{i=1}$ is calculated, and the sum is denoted as L_m.

$$\{S_i\}_{i=1} = Z_{cinor} \cdot Z_{cinor}^T. \tag{16}$$

Since in the autoencoder module, the output of the decoder is the reconstructed original data. The node-level traffic flow prediction results will be output through a complete fully connected layer, and the multi-channel is mapped to a single channel, which can be expressed as

$$L_p = \sum_t \left\| \left(\hat{X}^t + b\right) - X^t \right\|^2. \tag{17}$$

The final loss function is L, where a, b are the parameters.

$$L = L_{\text{p}} + aL_m + bL_s. \tag{18}$$

4. Experiments and Results

The proposed MV-GAT model is evaluated by comparing it with state-of-the-art baselines. The experimental dataset and baselines are first introduced, followed by the parameter setup. Finally, the experimental results and experimental analysis are presented.

4.1. Experiment Setting. We adopt the historical data of Beijing metro as the experimental dataset. MetroBJ [25] is a five-month passenger flow dataset, formally collected in 2015, with a granularity of 5 minutes. The dataset covers the entire subway network with 325 stations and 22 lines, covering the daily traffic data in July, August, September, November, and December. The time horizon is five months, covering weekdays and weekends. This time series contained in this dataset is long enough for us to divide multiple time spans to build multiple feature-level views.

The dataset contains the desensitized swipe ID, the line station and time of entering the subway, and the traffic flow data of the line station and time of leaving the subway. In the actual use of this method, firstly, a node set containing 325 nodes is constructed based on the subway stations in Beijing in this dataset, and a basic view containing 22 edges is constructed with reference to the subway network lines. On this basis, the DBSCAN algorithm is used to cluster the historical passenger flow data under three different time spans of hours, days, and weeks, and construct corresponding multi-views. Compared with the traditional

k-means algorithm, the DBSCAN algorithm does not need to input the number of clusters k and can find clusters of any shape, and at the same time, it can find outliers during clustering. Finally, the traffic flow values of each node in the next 5 minutes, 10 minutes, and 15 minutes are output to calculate the accuracy of the proposed model.

The comparison methods include two categories of nongraph methods and graph-based methods. The compared methods contain autoregressive integrated moving average (ARIMA) model [26], support vector regression (SVR) [27], and long short-term memory (LSTM) [28]. Graph-based deep learning methods contain temporal graph convolutional network (T-GCN) [29], spatio-temporal graph convolutional network (STGCN) [30], and diffusion convolutional recurrent neural network (DCRNN) [31]. The detailed parameter settings are listed as follows.

(1) ARIMA: ARIMA is a common time series forecasting methods. The degree of differencing d, lag order p, and the order of moving average q are determined with the "auto arima" in the "pyramid" library.

(2) SVR: One improvement of SVR is the tolerated deviation ε when calculating the loss. During training, the model with linear kernel has a penalty term C of 0.1 and a deviation ε of 0.1.

(3) LSTM: The compared LSTM model has hidden layers of [31] recurrent units. During the training phase, the batch size is 32, the activation function is sigmoid, and the learning rate is set to 10^{-2}.

(4) T-GCN: The temporal graph convolutional network has hidden units of GRU. The batch size is set to 64 while training, and the learning rate is set to 10^{-3}.

(5) STGCN: The spatial-temporal graph convolutional network has two convolution blocks with channel of [64,16,64]. The convolution kernel size is 3, and the batch size is 64.

(6) DCRNN: The diffusion convolutional recurrent neural network is a data-driven traffic prediction model with autoencoder framework. It has two RNN layers of 64 units. The batch size is set to 64, and the learning rate is set to 10^{-3}.

To quantitatively evaluate the prediction accuracy of the proposed method, the results of the experiments take mean absolute error (MAE) and root mean square error (RMSE) as performance metrics:

$$\begin{aligned} \mathrm{MAE} &= \sum_{i=1}^{T}\sum_{j=1}^{N}\frac{\left|X_{ij}-\widehat{X}_{ij}\right|}{T*N} \\ \mathrm{RMSE} &= \sum_{i=1}^{T}\sum_{j=1}^{N}\frac{\left(X_{ij}-\widehat{X}_{ij}\right)^{2}}{(T*N)^{1/2}}, \end{aligned} \tag{19}$$

where X_{ij} is the ground truth, the $\widehat{X}_{ij}$ is prediction value, T is the time length, and N is the node number. When MAE and RMSE are used as evaluation indicators, the lower the value, the higher the accuracy. All experiments are tested with the platform of CPU of "Intel(R) Xeon(R) Platinum 8268 CPU @ 2.90 GHz" and GPU of "NVIDIA GTX 2080Ti." The number of epochs of training phase is 50, and the batch size is 64. The learning rate is set to 10^{-2} and decreases to 10^{-4} gradually.

4.2. Experiment Results. To fully utilize the different views over multiple time spans, we use the data of a whole month as the experimental data. The experiments use ten-fold cross-validation to get stabler results. Considering the size of dataset per month, the training set, testing set, and valid set are split with 8 : 1 : 1 on the time dimension. The experimental results are shown in Table 1.

Table 1 shows the prediction accuracy when the historical data of July and September are used as the experimental dataset. As can be seen from the results, the accuracy of the ARIMA method is significantly lower than that of the machine learning and deep learning methods. SVR significantly outperforms ARIMA, and at the same time, LSTM is better than SVR by virtue of modeling long- and short-term sequences. T-GCN, an earlier method that combines graph networks with time series dependency, achieves similar accuracy to the relatively mature LSTM.

As a classical framework, STGCN has achieved more accurate prediction results, especially in the medium-term prediction of the next 45 minutes, where obvious advantages can be seen. With a unique architecture, DCRNN also achieves accurate results. Among all methods, our proposed method achieves better accuracy, especially on short-term predictions of 15 minutes and 30 minutes. Compared with machine learning methods and graph-based deep learning methods, there are significant improvements. More experimental results of flow prediction are shown in Table 2.

It can be seen from Table 2 that the prediction accuracy of each method is similar to that presented in Table 1. It shows that the rail transit shows a basically stable operation law in each month. It is worth mentioning that, similar to the previous set of experiments, STGCN achieves a clear advantage in 45-minute prediction results. It reflects the complexity of traffic forecasting from the side. In many cases, it is difficult to solve short-term forecasting, medium-term forecasting, and even long-term forecasting problems simultaneously with one model.

To prominently compare the role of each module of the proposed model, we design a set of ablation contrast experiments, as shown in Table 3. In this set of ablation experiments, we mainly compared the difference between single view and multi-view, and the role of the autoencoder.

The experimental data adopt the passenger flow data of Beijing rail transit in July. We first tested the single-view network model without the autoencoder module. The single view is the graph of the rail transit network. While removing the autoencoder, other parts of the proposed model remain unchanged. It can be seen that the prediction accuracy of this method is unsatisfactory, and it cannot even beat the STGCN model on this dataset. In the case of single view, whether the multi-layer attention mechanism has the effect of negative optimization is a new problem worth investigating.

Table 1: Accuracy results of rail passenger flow prediction experiment.

Methods	July		September	
	MAE	RMSE	MAE	RMSE
ARIMA	18.34/20.12/23.32	29.14/33.37/36.81	19.64/21.31/24.06	30.74/34.30/36.66
SVR	14.73/16.55/18.26	25.24/31.33/32.71	15.89/16.71/17.36	28.30/33.01/34.09
LSTM	10.76/12.27/12.86	21.22/22.33/23.74	11.95/12.56/13.77	23.95/26.43/28.34
T-GCN	10.88/12.46/12.73	20.93/22.72/24.61	11.00/12.77/13.75	23.98/25.98/28.35
STGCN	9.04/10.29/**10.88**	19.38/20.49/22.51	10.99/11.95/13.42	21.03/23.03/**24.06**
DCRNN	8.41/9.73/11.56	19.43/23.76/25.77	8.72/9.24/12.73	20.94/**22.01**/25.82
MV-GAT	**8.35/9.57**/11.60	**19.41/21.84/22.38**	**8.67/9.13/12.45**	**20.85**/22.13/25.67

Table 2: Accuracy results of rail passenger flow prediction experiment.

Methods	November		December	
	MAE	RMSE	MAE	RMSE
ARIMA	14.22/18.81/24.39	30.06/33.52/35.24	18.39/20.52/24.22	30.24/33.81/35.06
SVR	13.92/16.52/16.12	26.89/28.75/28.56	14.12/15.75/16.92	26.56/28.52/28.89
LSTM	11.49/13.79/14.05	21.36/22.33/25.50	11.05/12.33/14.49	21.50/22.79/25.36
T-GCN	10.99/12.76/13.44	21.48/23.68/25.62	10.90/13.78/13.34	21.42/23.74/25.85
STGCN	9.06/10.65/11.32	21.74/22.22/**23.55**	8.20/10.26/**11.25**	20.72/22.28/**23.25**
DCRNN	8.83/9.71/11.79	21.52/23.15/26.04	8.24/9.65/11.87	21.28/23.24/26.95
MV-GAT	**8.80/9.62/11.20**	**20.83/22.18**/23.39	**8.15/9.57**/11.43	**20.64/22.13**/24.72

Table 3: Ablation contrast experiment.

Methods	July	
	MAE	RMSE
Single view w/o autoencoder	9.21/10.31/12.38	19.68/22.49/23.51
Single view w/autoencoder	8.98/10.20/12.21	19.61/22.27/23.19
Multi-view w/o autoencoder	8.44/9.61/11.69	19.50/21.91/22.49
Multi-view w/autoencoder	**8.35/9.57/11.60**	**19.41/21.84/22.38**

By adding the autoencoder module to the single-view model, the prediction accuracy is improved, but the improvement is relatively limited. The autoencoder module can alleviate the gradient vanishing problem during training to a certain extent, especially for graph convolutional deep network models with many layers. Limited by the graph scale of the dataset used in this experiment, the number of layers in the network model is not many. Therefore, in the deeper graph convolution prediction model, it is worth looking forward to whether the autoencoder module can play a larger role.

After the introduction of multi-view, the prediction accuracy of the model is significantly improved compared to single view, with or without an autoencoder module. Among them, the model achieves the best prediction results when the multi-view module and the autoencoder module coexist.

5. Conclusions

This paper proposes a multi-view and multi-layer attention-based GCN model for the problem of rail traffic flow prediction. Considering that it is difficult to fully express the relationship between nodes in the node classification problem using only a single view, this model introduces multi-view and utilizes a multi-layer attention mechanism and an autoencoder module to achieve more accurate temporal prediction. Experimental results on the Beijing dataset show that our model outperforms other nongraph and graph-based benchmark methods. In the future, we will optimize the framework of the proposed method and try to design models for directed graphs. We also want to explore more comprehensively the application of graph-based deep learning in intelligent transportation systems.

Data Availability

The data supporting this proposed model are from previously reported studies and datasets, which have been cited. The processed data are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The research of this article was funded by Building Project for Continuing Education Online Resource Platform and Characteristic Professional High-Level Team of Colleges and

Universities and Beijing Municipal Education Commission, nos. 2019-630 and 2019.12.

References

[1] X. Wang, H. Ji, C. Shi, and B. Wang, "Heterogeneous graph attention network," in *Proceedings of the The world wide web conference*, pp. 2022–2032, San Francisco, CA, USA, May 2019.

[2] Y. Wang, Y. Zhang, X. Piao, H. Liu, and K. Zhang, "Traffic data reconstruction via adaptive spatial-temporal correlations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 4, pp. 1531–1543, 2019.

[3] J. Wang, Y. Zhang, L. Wang, X. Piao, and B. Yin, "Multitask hypergraph convolutional networks: a heterogeneous traffic prediction framework," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[4] R. Ying, R. He, K. Chen, P. Eksombatchai, and W. L. Hamilton, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 974–983, London, U.K, August 2018.

[5] W. Fan, Y. Ma, Q. Li, Y. He, and E. Zhao, "Graph neural networks for social recommendation," in *Proceedings of the The World Wide Web Conference*, pp. 417–426, San Francisco, CA, USA, May 2019.

[6] G. Huo, Y. Zhang, B. Wang, and Y. Hu, "Text-to-Traffic generative adversarial network for traffic situation generation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 3, pp. 2623–2636, 2021.

[7] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *Proceedings of the 32 AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, New Orleans, LA, USA, February 2018.

[8] J. Wu, J. He, and J. Xu, "Net: degree-specific graph neural networks for node and graph classification," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 406–415, Anchorage, Alaska, USA, July 2019.

[9] J. You, R. Ying, and J. Leskovec, "Position-aware graph neural networks," in *Proceedings of the International Conference on Machine Learning. PMLR*, pp. 7134–7143, Long Beach, CA, USA, June 2019.

[10] T. N. Kipf and M. Welling, "Variational graph autoencoders," 2016, https://arxiv.org/abs/1611.07308.

[11] S. Abu-El-Haija, B. Perozzi, A. Kapoor, N. Alipourfard, and K. Lerman, "Mixhop: higher-order graph convolutional architectures via sparsified neighborhood mixing," in *Proceedings of the international conference on machine learning. PMLR*, pp. 21–29, Long Beach, CA, USA, June 2019.

[12] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710, New York, NY, USA, February 2014.

[13] A. Grover and J. Leskovec, "node2vec: scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 855–864, New York, NY, USA, July 2016.

[14] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1225–1234, New York, NY, August 2016.

[15] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: theoretical basis and empirical results," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672, 2003.

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

[17] X. Wang, M. Zhu, D. Bo, P. Cui, and C. Shi, "Am-gcn: adaptive multi-channel graph convolutional networks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1243–1253, New York, NY, USA, August 2020.

[18] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, and P. Cui, "Structural deep clustering network," in *Proceedings of the Web Conference 2020*, pp. 1400–1410, Taipei Taiwan, April 2020.

[19] H. Nt and T. Maehara, "Revisiting graph neural networks: all we have is low-pass filters," 2019, https://arxiv.org/abs/1905.09550.

[20] H. Gao, J. Pei, and H. Huang, "Conditional random field enhanced graph convolutional neural networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 276–284, Anchorage, Alaska, USA, May 2019.

[21] K. Cho, B. Van Merriënboer, C. Gulcehre et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, https://arxiv.org/abs/1406.1078.

[22] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, and R. Salakhutdinov, "Show, attend and tell: neural image caption generation with visual attention," in *Proceedings of the International conference on machine learning. PMLR*, pp. 2048–2057, Lille, France, July 2015.

[23] M. T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015, https://arxiv.org/abs/1508.04025.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, and A. N. Gomez, "Attention is all you need," 2017, https://arxiv.org/abs/1706.03762.

[25] J. Wang, Y. Zhang, Y. Wei, Y. Hu, X. Piao, and B. Yin, "Metro passenger flow prediction via dynamic hypergraph convolution networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 12, pp. 7891–7903, 2021.

[26] S. Lee and D. B. Fambro, "Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1678, no. 1, pp. 179–188, 1999.

[27] R. Chen, C.-Y. Liang, W.-C. Hong, and D.-X. Gu, "Forecasting holiday daily tourist flow based on seasonal support vector regression with adaptive genetic algorithm," *Applied Soft Computing*, vol. 26, pp. 435–443, 2015.

[28] R. Fu, Z. Zhang, and L. Li, "Using lstm and gru neural network methods for traffic flow prediction," in *Proceedings of the 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 324–328, IEEE, Wuhan, China, November 2016.

[29] L. Zhao, Y. Song, C. Zhang, T. Lin, M. Deng, and H. Li, "T-gcn: a temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, 2019.

[30] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting," 2017, https://arxiv.org/abs/1709.04875.

[31] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: data-driven traffic forecasting," 2017, https://arxiv.org/abs/1707.01926.

Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 9463559, 11 pages
https://doi.org/10.1155/2022/9463559

Research Article

A Three-Stage Anomaly Detection Framework for Traffic Videos

Junzhou Chen,[1,2] **Jiancheng Wang**,[1,2] **Jiajun Pu**,[1,2] **and Ronghui Zhang**[1,2]

[1]*School of Intelligent Systems Engineering, Shenzhen Campus of Sun Yat-sen University, No. 66 Gongchang Road, Guangming District, Shenzhen, Guangdong 518107, China*
[2]*Guangdong Provincial Key Laboratory of Fire Science and Intelligent Emergency Technology, Sun Yat-sen University, Guangzhou 510006, China*

Correspondence should be addressed to Ronghui Zhang; zhangrh25@mail.sysu.edu.cn

Received 25 February 2022; Revised 6 April 2022; Accepted 11 June 2022; Published 5 July 2022

Academic Editor: Yong Zhang

As reported by the United Nations in 2021, road accidents cause 1.3 million deaths and 50 million injuries worldwide each year. Detecting traffic anomalies timely and taking immediate emergency response and rescue measures are essential to reduce casualties, economic losses, and traffic congestion. This paper proposed a three-stage method for video-based traffic anomaly detection. In the first stage, the ViVit network is employed as a feature extractor to capture the spatiotemporal features from the input video. In the second stage, the class and patch tokens are fed separately to the segment-level and video-level traffic anomaly detectors. In the third stage, we finished the construction of the entire composite traffic anomaly detection framework by fusing outputs of two traffic anomaly detectors above with different granularity. Experimental evaluation demonstrates that the proposed method outperforms the SOTA method with 2.07% AUC on the TAD testing overall set and 1.43% AUC on the TAD testing anomaly subset. This work provides a new reference for traffic anomaly detection research.

1. Introduction

With rapid economic development, a leapfrog has been achieved in transportation. Contrary to the wishes of [1], the number of civilian vehicles and the road network density are increasing, and the road network structure is becoming more complex. As a result, traffic management schemes are proposed correspondingly; numerous measures such as CCTV cameras and radars are put on the roadside to regulate the driving behavior of drivers [2–4]. Studies on the vehicle are carried out [5–9]. However, numerous traffic accidents with terrible consequences still happen every year [10]. According to the National Bureau of Statistics, in 2020, there were 244,674 traffic accidents in China, resulting in 61,703 deaths, 250,723 injuries, and a direct property loss of about 206 million dollars [11].

The extent of the damage often depends on when traffic controllers discover the incident and the duration of the traffic incident [12]. The lack of timely accident reporting will result in many deaths due to delays in medical assistance, prolonged traffic jams, and even secondary accidents.

Therefore, real-time detection of traffic incidents is an effective way to reduce their impact significantly. With the development of technology and the advancement of research, various detection technologies and data sources are used in automatic traffic accident detection studies. Traditional traffic data provides rich and relatively available data sources [13, 14], such as traffic data, vehicle speed data, and occupancy data. Numerous machine learning models are also applied to detect traffic incidents with traffic data and have achieved good results [15–18]. Some studies employed online data from mobile phones to detect traffic incidents, such as Twitter and Weibo. Specifically, they used web crawler technology to detect incidents through data processing, filtering, reasoning, and other processes [19, 20]. Moreover, Zhang and He [21] integrated the social media data with traffic data and achieved a better effect.

Another effective solution is to use surveillance video data. On the one hand, surveillance cameras are extensively used on modern roads and help traffic managers obtain rich surveillance video data of road areas. On the other hand, with the rapid development of computer vision and artificial

intelligence, many advancements have been achieved in video analysis and understanding research. Video-based surveillance for traffic incident detection became possible whether in the middle of the night or when the traffic flow is low.

For the research on traffic video anomaly detection, the video anomaly detection method can be divided into two categories according to the model type: the traditional machine learning method and the deep learning method. Traditional machine learning methods are mainly based on the Gaussian mixture model [22], histogram feature [23–25], hidden Markov model [26, 27], appearance feature [28, 29], and Bayesian network model [30]. Deep learning methods are mostly based on appearance features and motion features in specific scenes, and the final anomaly detection is performed by reconstruction error [31–36], prediction error [37–40], or hybrid transfer learning classification [41, 42].

However, the two methods mentioned above are often mixed and cannot be accurately distinguished in recent years. Therefore, we follow [43] and broadly classify video anomaly detection methods into three categories according to the detection granularity: video level, slice level, and frame level. This paper proposes a three-stage anomaly detection framework for traffic video. The main contributions can be summarized as follows:

(a) We proposed a novel weakly supervised learning method for traffic video anomaly detection. Specifically, in the first stage, the ViVit network is employed as a feature extractor to capture the spatiotemporal features from the input video. In the second stage, the class and patch tokens are fed separately to the segment-level and video-level traffic anomaly detectors. In the third stage, we finished the construction of the entire composite traffic anomaly detection framework by fusing outputs of two traffic anomaly detectors above with different granularity.

(b) We propose a segment-level traffic anomaly detector based on the global spatiotemporal features (class token), a video-level traffic anomaly detector based on the similarity of patch tokens from different segments, and a composite traffic anomaly detection framework. By entirely using video-level similarity features and all segment-level global spatiotemporal features, the long-tail distribution problem in traffic video anomaly detection tasks can be effectively solved.

(c) The experimental results demonstrate the effectiveness of the proposed method. Specifically, our proposed architecture achieves 91.71% and 63.09% on the overall set and anomaly subset of the TAD testing set, which are 2.07% and 1.43% higher than the SOTA method, respectively.

The rest of the paper is organized as follows. Section 2 discusses studies related to video anomaly detection in terms of three different detection granularities: video level, segment level, and frame level. The details of our three-stage anomaly detection framework are described in Section 3. Section 4 shows the implementation details and quantitative results of the experiments. Section 5 gives the conclusions and the focus of future work.

2. Literature Review

Rapid technological progress in computer vision and machine learning has enabled better video understanding. Many studies on traffic anomaly detection via surveillance video have been carried out in recent decades. Following [43], the techniques that could be applied in traffic video anomaly detection can be divided into three categories: video level [44], segment level [45], and frame level [46]. The details of the various method categories are described as follows.

2.1. Video-Level Methods. Popular single-class classification methods directly detect novelty by measuring the gap between the original and reconstructed inputs, such as Support Vector Machine (SVM) [44, 47] and SVDD [48, 49]. In general, video-level methods treat anomaly detection as a novel detection problem. Liu et al. [50] proposed a single-objective generative adversarial active learning method that directly generates information-rich potential outliers based on a mini-max game between the generator and the discriminator. Ngo et al. [51] used a similar approach based on generative adversarial networks (GANs).

2.2. Segment-Level Methods. Segment-level detection is a method between video level and frame level, which divides the input video into multiple segments instead of frames. In recent years, this research has become increasingly popular, and there is a growing body of related work. Some work built memory modules that learn only normal patterns from normal data and determine the presence of anomalies by computing reconstruction errors [33, 35]. In another interesting work, Georgescu et al. [52] proposed joint learning of multiple tasks by self-supervision to produce differential anomaly information: three self-supervised tasks and an ablation study. Moreover, a two-stage framework is also a popular research approach. Waqas et al. [41] applied pretrained 3D networks to extract spatiotemporal features and trained the classifier with multi-instance learning techniques. Following this work, Zhu and Newsam [45] introduced optical flow; Lin et al. [53] proposed a dual-branch network; Lv et al. [54] replaced the feature extractor with a TSN and proposed an HCE module to capture dynamic changes; Feng et al. [55] applied pseudolabel and self-attentive feature encoders for training; Wu et al. [56] also proposed a dual-branch network but with tubular and temporal branches and so on. This strategy can improve detection accuracy and localize anomalies using a small amount of annotated information.

2.3. Frame-Level Methods. Based on the classical directional optical flow histograms, references [23–25, 29] have developed their own way of extracting frame-level features for

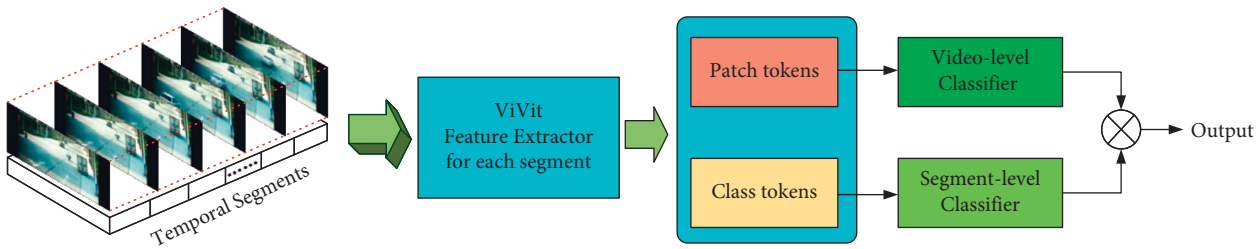


Figure 1: Three-stage video-based traffic anomaly detection algorithm framework.

anomaly detection, but they are scene-dependent. More generative models were used to predict future frames and calculate the reconstruction error between predicted and real frames. On this basis, reference [35] used U-Net and memory module; reference [36] used AE and DPU module. Both of them generate "normal" future frames and determine whether they are anomalous. Moreover, after generative adversarial networks (GANs) proved their ability to generate "normal" future frames, many researchers have focused their interest on detecting traffic anomalies at the frame level. In a similar way to determining anomalies by prediction errors [37–40, 46], the frame-level detection method based on GAN networks compares the current frames constructed by GANs with the ground truth current frames [31, 37, 57–59]. Besides GAN, there are other methods to detect traffic incidents at the frame level. Ryan Medel and Svakis [60] built an end-to-end frame-level anomaly detector using a long and short-term memory (Conv-LSTM) network. Zhou et al. [43] first detected boundary frames as potential incident frames and confirmed by encoding spatiotemporal features whether these frames are incident frames.

The following summary can be made from the above review, video-level methods usually aggregate features for single-class prediction, which can take full advantage of fully supervised tasks but cannot identify anomaly locations. Segment-level methods can be trained by weakly supervised learning mechanisms such as multi-instance learning to perform effective anomaly detection and localization while maintaining a few annotations (video-level annotations). Frame-level methods generally perform single-frame detection by calculating the reconstruction error between predicted and real frames, and although the localization is accurate, their application scenarios are limited and have significant errors. Therefore, in this paper, we combine the advantages of video-level methods and fragment-level methods to complement each other and propose a three-stage composite traffic anomaly detection framework to achieve the anomaly detection and localization of anomaly videos.

3. Method

As a carrier of spatiotemporal information, frames in video contain temporal information that is not available in mutually irrelative images. Therefore, understanding and analyzing videos is more complicated and time-consuming than understanding and analyzing images directly. Many current video anomaly detection methods are generally divided into two steps: the first step is to extract spatiotemporal features from the input video using a pretrained 3D model; the second step is to model the extracted spatiotemporal features and evaluate the anomaly score.

As shown in Figure 1, we propose a three-stage anomaly detection method for traffic videos. Unlike other methods, we use the pretrained ViVit to extract features from video segments and propose a composite framework of video-level and segment-level traffic anomaly detectors. Specifically, we first split the input video into multiple segments and then use the pretrained ViVit to extract spatiotemporal features from those segments. After that, their global spatiotemporal features (class tokens) and local spatiotemporal features (patch tokens) are delivered to the segment-level and video-level traffic anomaly detectors, respectively. Finally, the output results of the above two detectors are compound corrected to complete the final anomaly value evaluation.

In this paper, to avoid ambiguity, class tokens refer to the segment-level global spatiotemporal features extracted by the pretrained ViVit model, and patch tokens refer to the segment-level local spatiotemporal features extracted by the pretrained ViVit model.

3.1. Extract Spatiotemporal Features Based on ViVit. Unlike the 3D convolution-based feature extractor [61–63], the Transformer-based ViVit model can effectively model the long contextual information of the input video by using its attentional architecture. Therefore, here we use ViVit model 2 (Factorized Encoder) [64], which was pretrained [65] on the Kinetics-400 dataset, as the feature extractor.

The above ViVit model 2 adopts the embedding method of ViVit-B, that is, a tubelet embedding for the input video, whose tubelet size is set to $h \times w \times t = 16 \times 16 \times 2$. The Factorized Encoder consists of two independent transformer encoders. The first is a spatial encoder that models the short spatiotemporal relationships of nonoverlapping adjacent $t = 2$ frames and feeds its output (spatial class token) to the next encoder. The second is the temporal encoder, which uses the spatial class token within the above nonoverlapping adjacent $t = 2$ frames to model the video long spatiotemporal relationship. Finally, the global spatiotemporal features (class token) and the local spatiotemporal feature (patch tokens) are obtained.

Before extracting the temporal features, we perform a preprocessing operation on the input video. Specifically, we resize each frame in the video to 224×224 and normalize it. Like Waqas et al. [41], we slice the processed video into

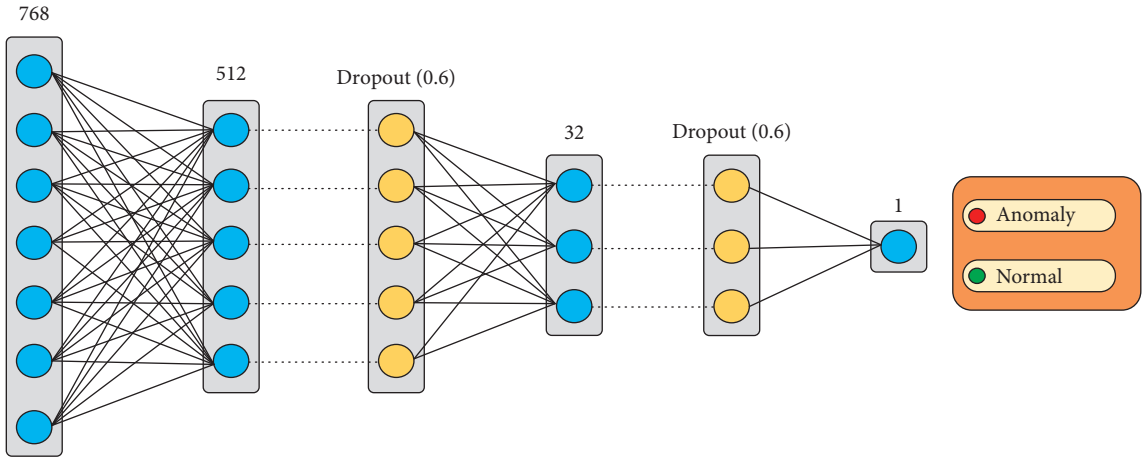


Figure 2: Segment-level classifier.

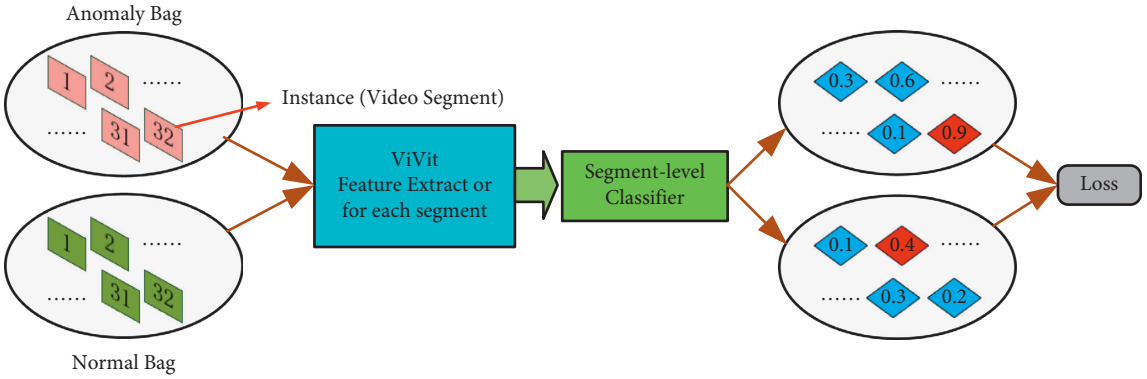


Figure 3: Multi-instance learning-based segment-level classification.

multiple video subunits, which are then distributed into 32 segments, where each video subunit is 16 frames. However, unlike the reference, we perform the averaging operation for each video subunit in the segment directly rather than after feature extraction. Then, each segment is subjected to spatiotemporal feature extraction using the pretrained ViViT model to obtain 1 class token and 8 patch tokens. The class token aggregates all the spatiotemporal features of the whole segment and represents the whole spatiotemporal segment. The patch token aggregates the certain local spatiotemporal features in the segment and represents the local spatiotemporal segment and its local contextual spatiotemporal segment. Finally, the class tokens of all segments are delivered to the segment-level anomaly detector for segment-level detection; the patch tokens of all segments are delivered to the video-level anomaly detector for video-level detection.

3.2. Segment-Level Traffic Anomaly Detector. As shown in Figure 2, we propose a segment-level classifier based on class token (768 dimensions). Our segment-level classifier is made up of five layers, detailed in Figure 2. Its last layer outputs an anomaly score, and the closer the score to 0, the greater the probability that the input segment is normal. Conversely, the closer the score to 1, the greater the probability that the input segment is abnormal.

Here, we use the multi-instance learning mechanism to train our segment-level traffic anomaly detector, a weakly supervised learning method, following [41]. As shown in Figure 3, it is the training framework of our segment-level traffic anomaly detector based on multi-instance learning:

(a) Input both positive bag (anomaly video) and negative bag (normal video) into 32 segments, and then compile those segments as a positive bag $\mathcal{B}_a$ and a negative bag $\mathcal{B}_n$. Each segment in its bag is called the instance, so the positive bag and the negative bag can be described as follow:

$$\begin{aligned} \mathcal{B}_a &= \{a_i, i = 1, \ldots, m\}, \\ \mathcal{B}_n &= \{n_i, i = 1, \ldots, m\}, \end{aligned} \tag{1}$$

where a_i is the instance in the positive pack and n_i is the instance in the negative pack. Our use case has $m = 32$.

(b) Under the basic assumption of multi-instance learning, there are only bag-level labels. Besides, each positive bag contains at least one positive example, while each negative bag contains no positive examples:

$$\begin{aligned} y_i^a &= 1, \quad \exists a_i \in \mathcal{B}_a, \\ y_i^n &= 0, \quad \forall n_i \in \mathcal{B}_n, \end{aligned} \tag{2}$$

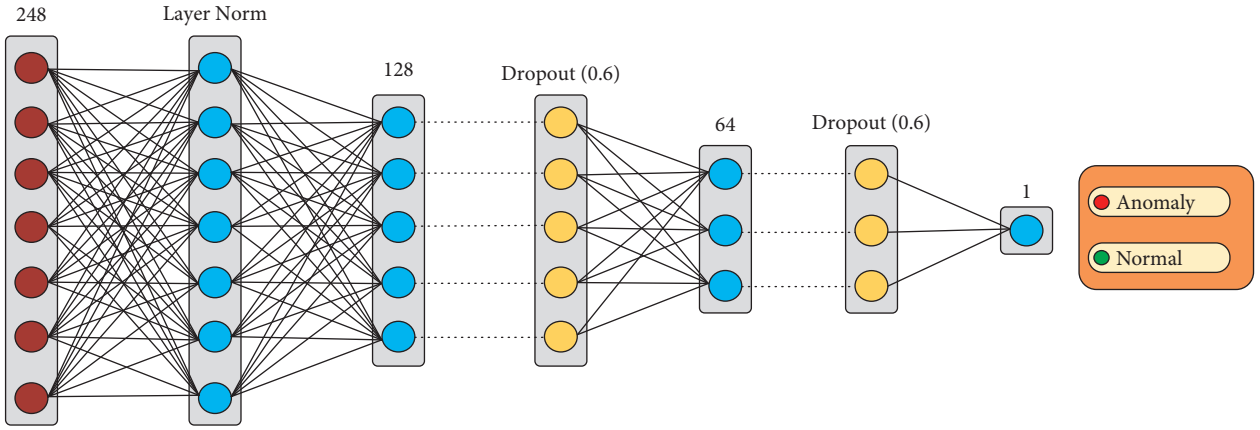


Figure 4: Video-level classifier.

where y_{ai} is the label of instance a_i and y_{ni} is the label of instance n_i. The instance is a positive instance when its label is 1, but a negative instance when its label is 0.

(c) Using pretrained ViVit mentioned in Section 3.1, we can extract the feature from all instances in both positive bag and negative bag to obtain their corresponding class token vector as follow:

$$\begin{aligned} \mathscr{C}_a &= \{c_i^a, i = 1, \ldots, m\}, \\ \mathscr{C}_n &= \{c_i^n, i = 1, \ldots, m\}v, \end{aligned} \tag{3}$$

where $\mathscr{C}_a$ is the class tokens feature set, extracted from the positive bag $\mathscr{B}_a$ with pretrained ViVit model, the same as $\mathscr{C}_n$.

(d) Put extracted feature (class token) of each instance into the segment-level classifier and acquire an anomaly score:

$$\begin{aligned} \mathscr{S}_a &= \{s_i^a = \mathscr{F}_s((c_i^a)i = 1, \ldots, m\}, \\ \mathscr{S}_n &= \{s_i^n = \mathscr{F}_s(c_i^n), i = 1, \ldots, m\}. \end{aligned} \tag{4}$$

Each training sample $\mathscr{X}$ should include one positive bag and one negative bag together, namely, $\mathscr{X} = \{\mathscr{B}_a, \mathscr{B}_n\}$. We use a combination of the following three loss functions to train the segment-level classifier $\mathscr{F}_s$. The first loss function is margin ranking loss. Choose the biggest instance anomaly score in positive and negative packets as their bag-level anomaly score for metric ranking loss calculation, where the metric parameter margin is set to 1.

$$l_{\text{margin}} = \max\left(0, \max_{n_i \in \mathscr{B}_n} \mathscr{F}_s(c_i^n) - \max_{a_i \in \mathscr{B}_a} \mathscr{F}_s(c_i^a) + \text{margin}\right). \tag{5}$$

The second loss function is the temporal smoothness term. Since video is a sequence of continuous frames combined, we split it into segments. In theory, the output anomaly score should be relatively smooth between segments. The temporal smoothness term is designed as

$$l_{\text{smooth}} = \sum_{i}^{(m-1)} \left(\mathscr{F}_s(c_i^a) - \mathscr{F}_s(c_{i+1}^a)\right)^2. \tag{6}$$

The third one is the sparsity term. For anomalies only take a small part of the entire video, the anomaly instance should be sparse in the positive bag:

$$l_{\text{sparsity}} = \sum_{i}^{m} \mathscr{F}_s(c_i^a). \tag{7}$$

Therefore, our final loss function becomes

$$\mathscr{L}_s = l_{\text{margin}} + \eta_1 l_{\text{smooth}} + \eta_2 l_{\text{sparsity}}. \tag{8}$$

Here, the η_1 and η_2 coefficients weight time smooth loss and sparse term loss separately.

3.3. Video-Level Traffic Anomaly Detector. As shown in Figure 4, we presented a novel video-level classifier. The input layer of the classifier is the similarity of patch tokens from adjacent segments of the same video. Our video-level classifier is made up of six layers, detailed in Figure 4. Its last layer outputs an anomaly score, and the closer the score to 0, the greater the probability that the input video is normal. Conversely, the closer the score to 1, the greater the probability that the input video is abnormal.

Here, we choose the cosine similarity to measure the degree of difference between two feature vectors. For example, given $\mathscr{P}_i$ and $\mathscr{P}_j$, let the cosine similarity calculation function be $\mathscr{F}_{\cos}$, and then, the similarity Sim_{ij} between two vectors is calculated as follows:

$$\begin{aligned} \text{Sim}_{ij} &= \mathscr{F}_{\cos}\left(\mathscr{P}_i, \mathscr{P}_j\right), \\ \mathscr{F}_{\cos}\left(\mathscr{P}_i, \mathscr{P}_j\right) &= \frac{\mathscr{P}_i \cdot \mathscr{P}_j}{\|\mathscr{P}_i\| \|\mathscr{P}_j\|} = \frac{\sum_{k=1}^{n} \mathscr{P}_k^i \times \mathscr{P}_k^j}{\sqrt{\sum_{k=1}^{n} \left(\mathscr{P}_k^i\right)^2} \times \sqrt{\sum_{k=1}^{n} \left(\mathscr{P}_k^j\right)^2}}. \end{aligned} \tag{9}$$

A normal video should remain continuous in its timeline, even after it is segmented. The continuity between adjacent segments can be reflected in their similarity. Therefore, a normal video should maintain a relatively high similarity between adjacent segments. In contrast, an abnormal video would be discontinuous in its timeline due to the presence of abnormal clips. So, the similarity between

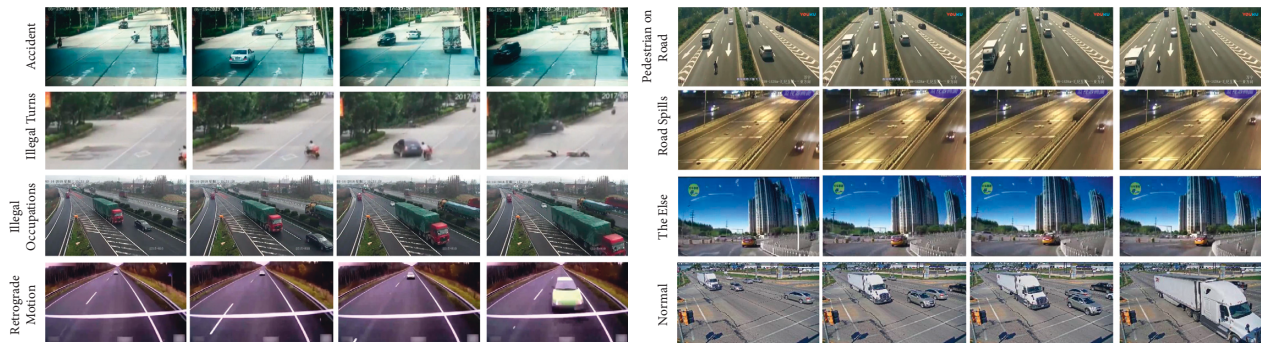


Figure 5: Examples of anomaly frames in the TAD dataset [54].

adjacent segments in anomaly video should dramatically decrease and unstable similarity between adjacent segments in which anomaly occurs.

Based on the above observation and analysis, we proposed a video-level traffic anomaly detector to focus on the feature similarity between segments and output the video-level anomaly score. Specifically, after the feature extraction in Section 3.1, an input video could get 32 groups of patch tokens (8 in each group). Then, we calculate the cosine similarity between each corresponding pair of patch token features in adjacent groups and finally get $8 \times 31 = 248$ patch token cosine similarity. Therefore, an entire video can be represented by a 248-dimensional similarity space feature vector, which is fed into a video-level traffic anomaly discriminator for forwarding derivation to obtain its video-level anomaly score.

In essence, our feature-similarity-based video-level traffic anomaly detector is a binary classification task whose parameters can be optimized with Binary Cross-Entropy Loss. After training on a large set of video-level labeled data, it is capable of performing high-performance anomaly traffic video discrimination.

$$\mathcal{L}_v = \mathcal{Y} * \log(\widehat{\mathcal{Y}}) + (1 - \mathcal{Y}) * \log(1 - \widehat{\mathcal{Y}}). \tag{10}$$

Here, $\mathcal{Y}$ is the label of input video, and $\widehat{\mathcal{Y}}$ is the output of the video-level traffic anomaly detector.

3.4. Composite Traffic Anomaly Detection. As mentioned earlier, video-level traffic anomaly detectors focus on feature similarity between adjacent video segments, while segment-level traffic anomaly detectors pay attention to modeling global spatiotemporal features within video segments. Theoretically, feature similarity between segments has stronger integrity and stability compared to global spatiotemporal features within segments. Therefore, the video-level anomaly traffic detector can provide a more reliable output and assist the segment-level detector in anomaly identification. Inspired by [33, 35], we design the following composite operation (equation (11)). When the anomaly score of the video-level traffic anomaly detector exceeds the threshold value, we normalize the output of the segment-level traffic anomaly detector by a min–max normalization [37]:

Table 1: Statistic of TAD dataset.

Dataset	Videos	Frames	Label level
Training set	400	452,220	Video level
Testing overall set	100	88,052	Frame level
Testing anomaly subset	60	18,900	Frame level

$$\mathcal{S}_C = \begin{cases} \dfrac{\mathcal{S} - \min\limits_{i \in [1,\ldots,m]} \mathcal{S}}{\max\limits_{i \in \{1,\ldots,m\}} \mathcal{S} - \min\limits_{i \in \{1,\ldots,m\}} \mathcal{S}}, & \text{if } \widehat{\mathcal{Y}} > \lambda, \\ \mathcal{S}, & \text{otherwise,} \end{cases} \tag{11}$$

where $\mathcal{S}_C$ is the composite traffic anomaly score, $\mathcal{S}$ is the output of segment-level traffic anomaly detector, $\widehat{\mathcal{Y}}$ is the output of video-level traffic anomaly detector, and λ is the preset threshold.

4. Experiment

4.1. Dataset and Training Details. We conducted the experiments on the TAD dataset built by Lv et al. [54], a total of 500 traffic surveillance videos with 250 normal and anomaly videos, respectively. The average frames in each clip of the TAD dataset are 1075. The anomalies randomly occur in the anomaly clips and take about 80 frames on average. The anomalies, including vehicle accidents, illegal turns, illegal occupations, retrograde motion, pedestrians on the road, and road spills, take place in various scenarios, weather conditions, and daytime periods.

Some examples of anomaly videos in the TAD dataset are shown in Figure 5. While training and testing, we followed [54] to split the TAD dataset into two parts, with a training set of 400 videos and a test set of 100 videos. Other statistics are shown in Table 1.

All experiments were carried out on PyTorch and hardware configuration of NVIDIA GeForce RTX 2070 GPU, 16 G RAM, CPU i7-10700k @3.80 GHz machine. We jointly use margin ranking loss, time smooth loss, and sparse term loss to train our segment-level anomaly traffic detector as mentioned in Section 3.2, where we set margin = 1, $\eta_1 = 8 \times 10^{-5}$, and $\eta_2 = 8 \times 10^{-5}$. It was trained of 1000 epochs with batch size 4. Binary Cross-Entropy Loss was

Table 2: Result of TAD dataset

Class	Method	Overall set AUC (%)	Anomaly subset AUC (%)
Unsupervised	Luo et al. [32]	57.89	55.84
	Liu et al. [37]	69.13	55.38
Weakly supervised	Sultani et al. [41]	81.42	55.97
	Zhu et al. [45]	83.08	56.89
	Lv et al. [54]	89.64	61.66
	Ours	**91.71**	**63.09**

applied to train the video-level traffic anomaly detector, which was 1000 epochs with batch size 8.

Both detectors were SGD Optimizer paired with Cosine Annealing LR; we both set their Optimizer parameters $lr = 0.001$, momentum $= 0.9$, and weight_decay $= 1 \times 10^{-4}$ and kept the best performed model parameters as the optimal model. In our experiment, by comparing different preset thresholds, it is proved that $\lambda = 0.6$ works best.

4.2. Evaluation Metrics. For the evaluation metrics of anomaly detection, we first defined "true positive (TP)," "false positive (FP)," "true negative (TN)," and "false negative (FN)," which represent the difference between the predicted and actual classes.

TP: the predicted class is "anomaly," and so is the actual class.

TN: the predicted class is "normal," and so is the actual class.

FN: the predicted class is "normal," but the actual class is "anomaly."

FP: the predicted class is "anomaly," but the actual class is "normal."

The true positive rate (TPR) is the probability that an actual positive will test positive, and the false positive rate (FPR) is defined as the probability of falsely rejecting the null hypothesis. TPR and FPR are calculated as follows:

$$\begin{aligned} \text{TPR} &= \frac{\text{TP}}{\text{TP} + \text{FN}}, \\ \text{FPR} &= \frac{\text{FP}}{\text{FP} + \text{TN}}. \end{aligned} \tag{12}$$

We choose the area under the frame-level ROC curve (AUC) as the primary evaluation metric for traffic video anomaly detection. The frame-level AUC is insensitive to the imbalance of sample classification and, therefore, suitable as our primary evaluation metric. Meanwhile, as an evaluation metric, the frame-level AUC reflects the detection performance of a method in locating traffic video anomalies. The closer the AUC value is to 1, the better the detection performance is.

The receiver operating characteristic curve (ROC) mentioned above is a graph showing the performance of the classification model at all classification thresholds, and the plotted curve represents the relationship between TPR and FPR.

We also used some other evaluation metrics to evaluate the ablation study of our proposed method. Precision and recall are two important evaluation metrics for detection evaluation. The precision (equation (13)) of a class reflects the proportion of the number of TP among the total number of elements that are predicted and labeled as the positive class. Recall (equation (14)) is defined as the proportion of the number of TP among the total number of the positive classes. Recall and precision are contradictory measures, and the $F1$-score (equation (15)) is defined as a combination of recall and precision.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{13}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{14}$$

$$F1 - \text{score} = 2 * \frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}. \tag{15}$$

4.3. Comparison with SOTA Method. In this paper, we compare the performance of the proposed method with several other SOTA methods, and their quantitative results on TAD are shown in Table 2. Among all the methods, the work by Luo et al. [32] and Liu et al. [37] uses an unsupervised approach and trains with only the normal video training set. Otherwise, Sultani et al. [41], Zhu et al. [45], Lv et al. [54], and our work use weakly supervised learning methods with the video-level labeled training set for training. The above SOTA results on TAD refer to [54].

The comparative results of the performance on TAD are given in Table 2. They represent that the weakly supervised learning methods outperform the unsupervised learning methods. For example, the relatively inefficient weakly supervised learning method [41] reaches 81.42% AUC on the overall set and 55.97% AUC on the anomaly subset, yet still about 12.29% and 0.13% higher than the best unsupervised learning method [37]. Besides, among the current SOTA methods, Lv et al. perform best on both the overall set and anomaly subset, with 89.64% AUC on the overall set and 61.66% AUC on the anomaly subset. However, the proposed method outperforms the optimal SOTA with 2.07% and 1.43% higher AUC on the overall set and anomaly subset, separately. The results show that our work has been the SOTA on the TAD dataset.

The above quantitative analysis proves the following points: (1) Unsupervised learning methods have limited performance in complex scenarios and when data anomalies are not significant. (2) Weakly supervised learning methods

Table 3: Ablation studies on TAD dataset.

Dataset	Methods	Recall (%)	Precision (%)	F1-score	AUC (%)
Overall set	T-SAD	92.16	90.17	0.9088	91.05
	T-CAD	92.00	90.48	0.9109	91.71
Anomaly subset	T-SAD	66.68	62.68	0.6154	62.04
	T-CAD	66.62	63.15	0.6279	63.09

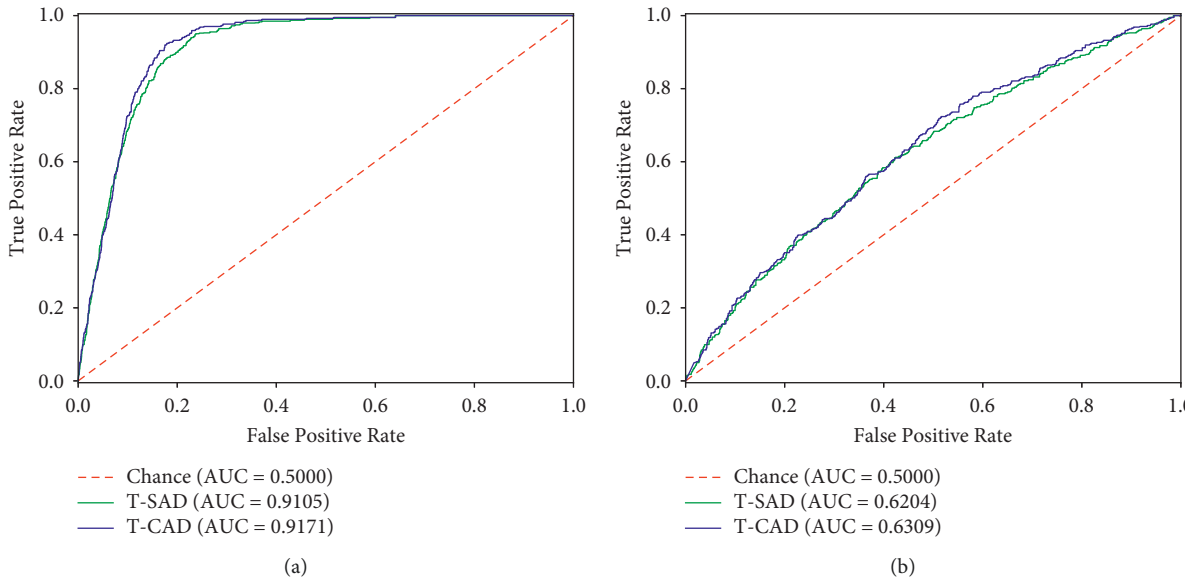


Figure 6: ROC curves on TAD dataset.

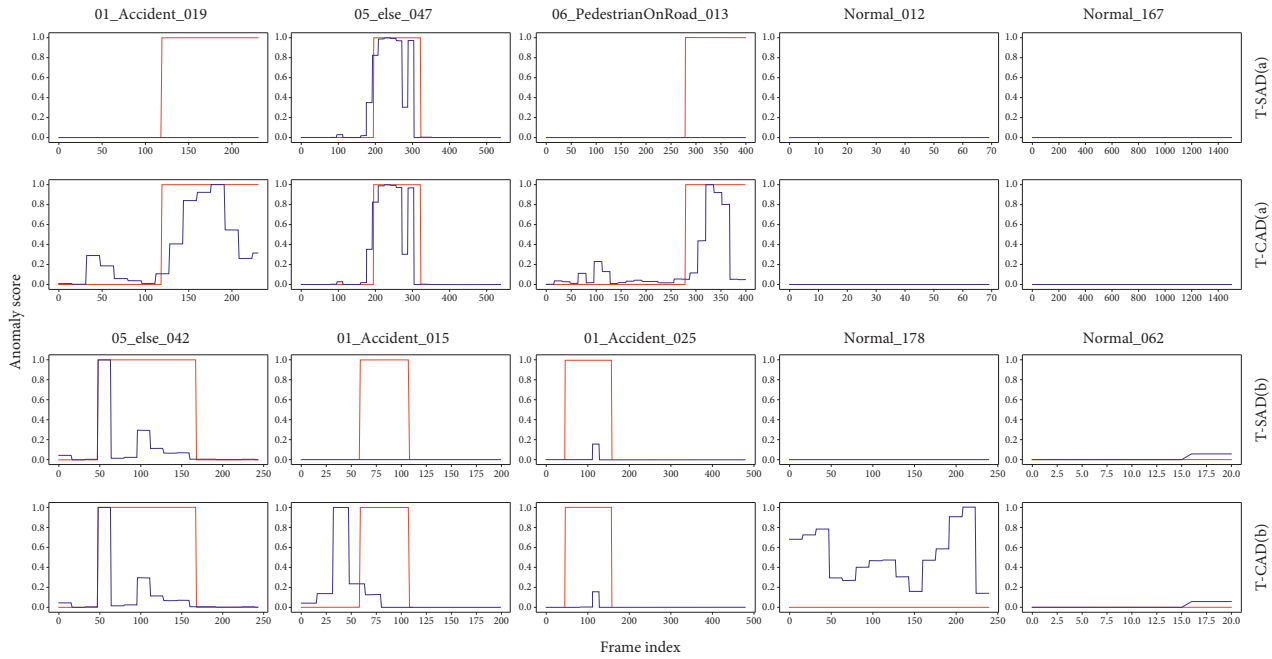


Figure 7: Video detection result.

can significantly improve the learning and representational ability of neuronal networks on training data while maintaining a small number of annotations. (3) The proposed method is more advanced in anomaly detection and localization, where the ViVit-based feature extractor can effectively characterize the pattern features of video data, and the ViVit-based composite traffic anomaly detection method can more accurately capture the anomalous features

in video data, making the method in this paper significantly better than the existing SOTA method [54].

4.4. Ablation Studies. We conducted ablation experiments to analyze the performance advantages of the Transformer-based Segment-level traffic Anomaly Detector (T-SAD) itself and the performance advantages of the Transformer-based Composite traffic Anomaly Detection method (T-CAD). As shown in Table 3, the AUC of T-SAC reached 91.05% and 62.04% on the overall set and anomaly subset, respectively, exceeding the current SOTA method [54] by 1.14% and 0.38%, respectively. In addition, the AUC values of T-CAD were 0.66% and 1.05% higher than those of T-SAD on the overall set and anomaly subset, respectively, demonstrating the better performance of T-CAD compared with T-SAD in anomaly localization.

Figure 6 visualizes the ROC curves of T-SAD and T-CAD on the overall set and anomaly subset and vividly demonstrates the superiority of the proposed method. As seen from Figure 6, the ROC curve of T-CAD clearly wraps around the ROC curve of T-SAD, proving that the T-CAD outperforms the T-SAD in all aspects of the overall set and anomaly subset.

We further visualized the detection results of T-SAD and T-CAD on the overall set separately. In the visualized results in Figure 7, row T-CAD (a) shows some reliable outputs from T-CAD on the test set, where T-SAD (a) is the corresponding outputs of T-SAD. It shows an improvement that T-CAD did compare to T-SAD. Still, in Figure 7, T-CAD (b) is some failure outputs from T-CAD on the overall testing set and its corresponding T-SAD. Enhancing detection ability could cause a higher probability of misdetection to catch abnormal features distributed sparely in anomalous videos. The exaggeration of the failure outputs is in keeping with the trait of T-CAD, widening the gap in T-SAD results. Nonetheless, no matter the overall set or anomaly subset, performance enhancement proved the effectiveness of our T-CAD structure.

5. Conclusion

In this work, we propose a three-stage anomaly detection for traffic videos. First, we utilize a pretrained ViVit model as the feature extractor to capture the spatiotemporal features of the input video. Then, we put the class tokens into the segment-level traffic anomaly detector for segment-level detection, pretrained with a multi-instance learning strategy. We similarly put the patch tokens into the video-level traffic anomaly detector for video-level detection. Finally, we fuse the video-level and segment-level detection outputs as our final output. From the experimental results, our proposed architecture achieves 91.71% AUC and 63.09% AUC on testing overall set and testing anomaly subset, which outperforms the SOTA method with 2.07% and 1.43%, respectively. Overall, the quantitative results demonstrate the effectiveness of using a spatiotemporal feature extractor and our composite traffic anomaly detection framework on the traffic video anomaly detection problem.

The feature extraction, fusion of foreground and background information, and modeling of relationships between foreground objects may be helpful for anomaly feature extraction, which is worth doing in the future. In addition, the spatial location detection of anomalies and the specific classification of anomalies are also worthy topics for research.

Data Availability

The data used to support the findings of this study are available from the first author and the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was partially supported by the Shenzhen Fundamental Research Program (no. JCYJ20200109142217397), the Guangdong Natural Science Foundation (nos. 2021A1515011794 and 2021B1515120032), the National Natural Science Foundation of China (no. 52172350), and the Guangzhou Science and Technology Plan Project (nos. 202007050004 and 202206030005).

References

[1] J. d D. Ortúzar, "Future transportation: sustainability, complexity and individualization of choices," *Communications in Transportation Research*, vol. 1, Article ID 100010, 2021.

[2] A. Franklin, "The future of cctv in road monitoring," *IEE Seminar on CCTV and Road Surveillance*, vol. 10, 1999.

[3] Ki Yong-Kul, J.-W. Choi, Ho-J. Joun, G.-H. Ahn, and K.-C. Cho, "Real-time estimation of travel speed using urban traffic information system and cctv," in *Proceedings of the 2017 International Conference on Systems, Signals and Image Processing (IWSSIP)*, 22-24 May 2017.

[4] F. Baselice, G. Ferraioli, G. Matuozzo, V. Pascazio, and G. Schirinzi, "3d automotive imaging radar for transportation systems monitoring," in *Proceedings of the 2014 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems Proceedings*, 17-18 September 2014.

[5] R.-H. Zhang, Z.-C. He, H.-W. Wang, F. You, and Ke-N. Li, "Study on self-tuning tyre friction control for developing main-servo loop integrated chassis control system," *IEEE Access*, vol. 5, pp. 6649–6660, 2017.

[6] Q. Yang, G. Shen, C. Liu, Z. Wang, K. Zheng, and R. Zheng, "Active fault-tolerant control of rotation angle sensor in steer-by-wire system based on multi-objective constraint fault estimator," *Journal of Intelligent and Connected Vehicles*, vol. 12, 2020.

[7] Y. Cai, T. Luan, H. Gao et al., "Yolov4-5d: an effective and efficient object detector for autonomous driving," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, no. 1–13, pp. 1–13, 2021.

[8] X. Zhao, X. Li, Y. Chen, H. Li, and Y. Ding, "Evaluation of fog warning system on driving under heavy fog condition based

on driving simulator," *Journal of intelligent and connected vehicles*, vol. 4, no. 2, pp. 41–51, 2021.

[9] K. Li Lim, J. Whitehead, D. Jia, and Z. Zheng, "State of data platforms for connected vehicles and infrastructures," *Communications in Transportation Research*, vol. 1, Article ID 100013, 2021.

[10] G. R. Gang and Z. Zhuping, "Traffic safety forecasting method by particle swarm optimization and support vector machine," *Expert Systems with Applications*, vol. 38, no. 8, pp. 10420–10424, 2011.

[11] Prc Bureau of Statistics, "China Statistical Yearbook in 2021," 2022, http://www.stats.gov.cn/tjsj/ndsj/2021/indexch.htm.

[12] W. Zhu, J. Wu, T. Fu, J. Wang, J. Zhang, and Q. Shangguan, "Dynamic prediction of traffic incident duration on urban expressways: a deep learning approach based on lstm and mlp," *Journal of intelligent and connected vehicles*, vol. 4, no. 2, pp. 80–91, 2021.

[13] D. Ma, X. Song, and P. Li, "Daily traffic flow forecasting through a contextual convolutional recurrent neural network modeling inter- and intra-day traffic patterns," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 5, pp. 2627–2636, 2021.

[14] Y. Liu, C. Lyu, Y. Zhang, Z. Liu, W. Yu, and X. Qu, "Deeptsp: deep traffic state prediction model based on large-scale empirical data," *Communications in Transportation Research*, vol. 1, Article ID 100012, 2021.

[15] Ho-C. Kho and S. Kho, "Predicting crash risk and identifying crash precursors on Korean expressways using loop detector data," *Accident Analysis & Prevention*, vol. 88, no. 9–19, pp. 9–19, 2016.

[16] J. Wang, W. Xie, B. Liu, S. Fang, and D. R. Ragland, "Identification of freeway secondary accidents with traffic shock wave detected by loop detectors," *Safety Science*, vol. 87, pp. 195–201, 2016.

[17] Y. W. Liyanage, D.-S. Zois, and C. Chelmis, "Near Real-Time Freeway Accident Detection," vol. 1, 2020 IEEE Transactions on Intelligent Transportation Systems.

[18] A. B. Parsa, A. Movahedi, H. Taghipour, S. Derrible, and A. K. Mohammadian, "Toward safer highways, application of xgboost and shap for real-time accident detection and feature analysis," *Accident Analysis & Prevention*, vol. 136, Article ID 105405, 2020.

[19] Z. Zhang, Q. He, J. Gao, and M. Ni, "A deep learning approach for detecting traffic accidents from social media data," *Transportation Research Part C: Emerging Technologies*, vol. 86, pp. 580–596, 2018.

[20] E. D'Andrea, P. Ducange, B. Lazzerini, and F. Marcelloni, "Real-time detection of traffic from twitter stream analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 4, pp. 2269–2283, 2015.

[21] Z. Zhang and Q. He, "On-site traffic accident detection with both social media and traffic data," in *Proceedings of the 9th Triennial Symp. Transp. Anal*, TRISTAN), China, June 19-26, 2022.

[22] Y. Li, W. Liu, and Q. Huang, "Traffic anomaly detection based on image descriptor in videos," *Multimedia Tools and Applications*, vol. 75, no. 5, pp. 2487–2505, 2016.

[23] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal, "Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, 1 20-25 June 2009.

[24] R. V. H. M. Colque, C. Caetano, M. T. L. de Andrade, and W. R. Schwartz, "Histograms of optical flow orientation and magnitude and entropy to detect anomalous events in videos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 3, pp. 673–682, 2017.

[25] Y. Zhang, H. Lu, L. Zhang, and X. Ruan, "Combining motion and appearance cues for anomaly detection," *Pattern Recognition*, vol. 51, pp. 443–452, 2016.

[26] L. Kratz and N. Ko, "Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1446–1453, IEEE, Miami, FL, USA, 1 20-25 June 2009.

[27] T. Hospedales, S. Gong, and T. Xiang, "A Markov clustering topic model for mining behaviour in video," in *Proceedings of the 2009 IEEE 12th International Conference on Computer Vision*, pp. 1165–1172, IEEE, Kyoto, Japan, 29 September 2009 - 02 October 2009.

[28] C. Yang, J. Yuan, and Ji Liu, "Sparse reconstruction cost for abnormal event detection," in *Proceedings of the CVPR 2011*, pp. 3449–3456, IEEE, Colorado Springs, CO, USA, 20-25 June 2011.

[29] W. Li, V. Mahadevan, and N. Vasconcelos, "Anomaly detection and localization in crowded scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, pp. 18–32, 2014.

[30] C. G. Blair and N. M. Robertson, "Event-driven dynamic platform selection for power-aware real-time anomaly detection in video," in *Proceedings of the 2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, pp. 54–63, IEEE, Lisbon, Portugal, 05-08 January 2014.

[31] M. Hasan, J. Choi, J. Neumann, K. Amit, and L. S. Davis, "Learning temporal regularity in video sequences," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 733–742, Las Vegas, June 2016.

[32] W. Luo, L. Wen, and S. Gao, "A revisit of sparse coding based anomaly detection in stacked rnn framework," in *Proceedings of the IEEE international conference on computer vision*, p. 341, Venice, Italy, 22-29 October 2017.

[33] D. Gong, L. Liu, V. Le et al., "Memorizing normality to detect anomaly: memory-augmented deep autoencoder for unsupervised anomaly detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1705–1714, Seoul Korea, Octomber 2019.

[34] T.-N. Nguyen and J. Meunier, "Anomaly detection in video sequence with appearance-motion correspondence," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1273–1283, Seoul Korea, Octomber 2019.

[35] H. Park, J. Noh, and B. Ham, "Learning memory-guided normality for anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14372–14381, Seoul Korea, 2020.

[36] H. Lv, C. Chen, Z. Cui, C. Xu, Y. Li, and J. Yang, "Learning normal dynamics in videos with meta prototype network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages*, pp. 15425–15434, Seoul Korea, 2021.

[37] L. Wen, W. Luo, D. Lian, and S. Gao, "Future frame prediction for anomaly detection–a new baseline," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6536–6545, Salt Lake City, June 2018.

[38] S. Lee, H. G. Kim, and Y. M. Ro, "Bman: bidirectional multi-scale aggregation networks for abnormal event detection," *IEEE Transactions on Image Processing*, vol. 29, pp. 2395–2408, 2020.

[39] F. Dong, Yu Zhang, and X. Nie, "Dual discriminator generative adversarial network for video anomaly detection," *IEEE Access*, vol. 8, pp. 88170–88176, 2020.

[40] K. Yilmaz and Y. Yilmaz, "Online anomaly detection in surveillance videos with asymptotic bound on false alarm rate," *Pattern Recognition*, vol. 114, Article ID 107865, 2021.

[41] S. Waqas, C. Chen, and M. Shah, "Real-world anomaly detection in surveillance videos,"pp. 6479–6488Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, June 2018.

[42] K. Doshi and Y. Yilmaz, "Continual learning for anomaly detection in surveillance videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 254-255, New Orleans, Louisiana, June 19th – 23rd.

[43] Z. Zhou, X. Dong, Z. Li, K. Yu, C. Ding, and Y. Yang, "Spatio-temporal feature encoding for traffic accident detection in vanet environment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 110 pages, 2022.

[44] M. Sabokrou, M. Khalooei, M. Fathy, and E. Adeli, "Adversarially learned one-class classifier for novelty detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3379–3388, Salt Lake City, 2018.

[45] Yi Zhu and S. Newsam, "Motion-aware feature for improved video anomaly detection," 2019, https://arxiv.org/abs/1907.10211.

[46] H. Zenati, C. S. Foo, L. Bruno, G. Manek, and V. R. Chandrasekhar, "Efficient gan-based anomaly detection," 2018, https://arxiv.org/abs/1802.06222.

[47] C. Vapnik and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[48] D. M. J. Duin and R. P. W. Duin, "Support vector data description," *Machine Learning*, vol. 54, no. 1, pp. 45–66, 2004.

[49] K. T. Chui, R. W. Liu, M. Zhao, and P. O. De Pablos, "Predicting sperformance with school and family tutoring using generative adversarial network-based deep support vector machine," *IEEE Access*, vol. 8, pp. 86745–86752, 2020.

[50] Y. Liu, Z. Li, C. Zhou et al., "Generative adversarial active learning for unsupervised outlier detection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 8, p. 1, 2019.

[51] P. Cuong Ngo, A. Aristo Winarto, S. Park, F. Akram, and H. Kuan Lee, "Fence gan: towards better anomaly detection," in *Proceedings of the 2019 IEEE 31St International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 141–148, IEEE, Portland, OR, USA, 04-06 November 2019.

[52] M.-I. Georgescu, A. Barbalau, R. Tudor Ionescu, F. S. Khan, M. Popescu, and M. Shah, "Anomaly detection in video via self-supervised and multi-task learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, p. 12742, Salt Lake City, Octomber 2021.

[53] S. Lin, H. Yang, X. Tang, T. Shi, and L. Chen, "Social mil: interaction-aware for crowd anomaly detection," in *Proceedings of the 2019 16th IEEE international conference on advanced video and signal based surveillance (AVSS)*, 18-21 September 2019.

[54] H. Lv, C. Zhou, Z. Cui, C. Xu, Y. Li, and J. Yang, "Localizing anomalies from weakly-labeled videos," *IEEE Transactions on Image Processing*, vol. 30, pp. 4505–4515, 2021.

[55] J.-C. Feng, Fa-T. Hong, and W.-S. Zheng, "Mist: multiple instance self-training framework for video anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14009–14018, Seoul Korea, June 2021.

[56] J. Wu, W. Zhang, G. Li et al., "Weakly-supervised spatio-temporal anomaly detection in surveillance video," 2021, https://arxiv.org/abs/2108.03825.

[57] J. Donahue, P. Krahenbuhl, and Trevor Darrell, "Adversarial Feature Learning," 2016, https://arxiv.org/abs/1605.09782.

[58] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *Lecture Notes in Computer Science*vol. 146, Springer157 pages, Springer, 2017.

[59] T. Schlegl, P. Seeböck, S. M. Waldstein, and G. Langs, "Schmidt-Erfurth: f-AnoGAN: f," *Medical Image Analysis*, vol. 54, pp. 30–44, 2019.

[60] J. Ryan Medel and A. Savakis, "Anomaly detection in video using predictive convolutional long short-term memory networks," 2016, https://arxiv.org/abs/1612.00390.

[61] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308, Salt Lake City, June 2017.

[62] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Salt Lake City, June 2019.

[63] C. Feichtenhofer, "X3d: expanding architectures for efficient video recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Soeul Korea, 2020.

[64] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, "Vivit: a video vision transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Soeul Korea, 2021.

[65] 2022, https://github.com/mx-markdsludslu/VideoTransformer-pytorch.

Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 2811961, 15 pages
https://doi.org/10.1155/2022/2811961

Research Article

MSASGCN: Multi-Head Self-Attention Spatiotemporal Graph Convolutional Network for Traffic Flow Forecasting

Yang Cao, Detian Liu, Qizheng Yin, Fei Xue, and Hengliang Tang

School of Information, Beijing Wuzi University, Beijing 101149, China

Correspondence should be addressed to Hengliang Tang; tanghengliangbwu@163.com

Received 6 February 2022; Revised 22 April 2022; Accepted 14 May 2022; Published 16 June 2022

Academic Editor: Yong Zhang

Traffic flow forecasting is an essential task of an intelligent transportation system (ITS), closely related to intelligent transportation management and resource scheduling. Dynamic spatial-temporal dependencies in traffic data make traffic flow forecasting to be a challenging task. Most existing research cannot model dynamic spatial and temporal correlations to achieve well-forecasting performance. The multi-head self-attention mechanism is a valuable method to capture dynamic spatial-temporal correlations, and combining it with graph convolutional networks is a promising solution. Therefore, we propose a multi-head self-attention spatiotemporal graph convolutional network (MSASGCN) model. It can effectively capture local correlations and potential global correlations of spatial structures, can handle dynamic evolution of the road network, and, in the time dimension, can effectively capture dynamic temporal correlations. Experiments on two real datasets verify the stability of our proposed model, obtaining a better prediction performance than the baseline algorithms. The correlation metrics get significantly reduced compared with traditional time series prediction methods and deep learning methods without using graph neural networks, according to MAE and RMSE results. Compared with advanced traffic flow forecasting methods, our model also has a performance improvement and a more stable prediction performance. We also discuss some problems and challenges in traffic forecasting.

1. Introduction

With the development of society and accelerated urbanization, the demand for urban transportation is growing. The problems arising from traffic congestion and road planning make it essential to have effective traffic management and planning. The rapid development of information technology makes intelligent transportation systems (ITS) gradually become an indispensable and critical part of urban transportation. It can bring efficient traffic management, accurate resource allocation, and traffic service support [1]. Advanced ITS needs efficient traffic data processing, and modeling of traffic data is the first task of ITS. Currently, there are multiple data collection methods in intelligent transportation, and the number of sensors deployed on the roadways has increased significantly. These sensors recorded information about vehicles passing through different road nodes' speed, flow, and size [2]. How to effectively process and analysis these multidimensional data to use them further for traffic prediction is an important research problem.

Traffic forecasting is an integral part of ITS, and timely and accurate traffic forecasting information helps managers make decisions and helps vehicle drivers choose smoother road trips, which can alleviate or avoid problems such as traffic congestion and traffic accidents [3]. Traffic flow forecasting is a crucial task, aiming to use historical traffic data from road networks to predict traffic flow in future time steps [4]. Traffic flow forecasting can be divided into short-term (within 30 min) and long-term (over 30 min) scales based on the future length of the forecast in the time dimension. Traditional forecasting approaches are ineffective in predicting medium- and long-term situations and only have some advantages in short-term forecasting [5]. In addition, traffic flow forecasting relies on sequential patterns in the time dimension and road networks in the spatial dimension. The connectivity relationships between different

road nodes can affect each other to influence the overall prediction accuracy [6]. Traffic flow is highly dynamic and spatial-temporal correlated as it changes with time and space and is a nonlinear problem that combines complexity and uncertainty.

For traffic flow forecasting problems, the existing research approaches can be divided into three categories which are classical statistics-based models, traditional machine learning-based models, and deep learning-based models. Due to the massive data generation and growth in the computing power of devices, the primary approaches for traffic flow forecasting are gradually evolving into data-driven deep learning methods [7]. Classical statistical-based forecasting models use limited data for analysis, regression, and optimization but fail to enable forecasting at large data scales and long-term forecasting. Traditional machine learning-based forecasting models mainly use machine learning methods to mine historical traffic flow data trends to predict future traffic status. But the complexity of historical traffic flow data is not effectively handled, making it impossible to achieve good prediction performance. Deep learning-based forecasting models often utilize neural network models, such as CNN and RNN, to model temporal and spatial dependencies. The overall performance of traffic forecasting is improved compared with the previous two approaches. Using deep learning has improved the overall performance of traffic forecasting models, but it is not the best solution yet. The main reason is the lack of adequate consideration of the spatiotemporal correlation of rapidly growing traffic data and the complexity of traffic networks.

Traffic flow forecasting is vital for intelligent transportation applications. Traffic flow data are mainly collected by sensors on the road, with the dynamic influence of the data collected by sensors between different location nodes in a specific time interval. Therefore, modeling the traffic flow forecasting problem is difficult due to the dynamic spatial-temporal correlation of traffic flows. It makes timely and accurate traffic flow forecasting very challenging. Exploring the nonlinear and complex traffic data to capture the temporal dependence and spatial dependence to get the potential spatiotemporal patterns is an essential issue in traffic flow forecasting [8].

Recently, graph neural networks (GNNs) [9] as a novel deep learning method have received a lot of research and attention due to their ability to directly model complex relationships. Representing non-Euclidean data as graphs with complex relationships and interdependencies between objects, GNNs can be effective methods for solving complex problems [10]. Graph neural networks are well suited for the field of traffic forecasting. The spatiotemporal correlation of traffic data can be effectively handled using graph neural networks, which can simultaneously deal with the temporal dynamics and the complexity of road networks, significantly improving the forecasting performance [11].

Although GNNs-based methods have achieved some advantages in traffic flow forecasting, the ability to model dynamic spatiotemporal correlation of traffic data is not perfect. Most current studies have not addressed the highly dynamic nonlinear spatiotemporal correlation challenges in traffic flow forecasting. The information on traffic data observed at different nodes is not entirely independent and is influenced by adjacent nodes and time steps, which are dynamically correlated. Figure 1 illustrates the complex spatial-temporal correlation, showing the spatial and temporal dynamics in traffic forecasting. In subfigure 1(a), three sensors in the road network are distributed in different ways, and even though they are geographically close in the road network, correlations do not always exist. The data information recorded by the sensors all differ. In subfigure 1(b), the correlation between traffic conditions at different time steps is different. For instance, sensor B is more correlated at time step $t + h + 1$ and $t - 1$ than with the nearest time step.

As mentioned above, traffic flow data exhibit strong dynamics and complexity in spatial and temporal dimensions. An accurate traffic flow forecast will depend on the effective treatment of spatiotemporal correlations in complex nonlinear traffic data. We propose a multi-head self-attention spatiotemporal graph convolutional network (MSASGCN) model to address these issues. Our model can effectively capture the potential spatial correlation and dynamic temporal features in the traffic road network. It can be adapted to the dynamic changes of the road network and used for traffic flow forecasting of different time lengths.

The main contributions of this article are as follows.

(1) To address the challenge of spatial-temporal correlation in traffic flow forecasting, we propose a novel deep learning model, the multi-head self-attention spatiotemporal graph convolutional network (MSASGCN). It can learn the temporal and spatial dependencies of dynamic traffic data and effectively forecast traffic flow in different periods.

(2) We use GCN to construct a spatial correlation model for road networks based on connection relations and a multi-head self-attention mechanism to capture the hidden spatial correlation between road networks and aggregate information among different nodes. A temporal convolution module is added to capture the dynamically changing temporal correlations. And based on the periodic characteristics of time series, an extended MSASGCN model is proposed to handle better the traffic flow forecasting problem with different temporal attributes.

(3) We conducted extensive experiments on real-world datasets to verify the proposed model validity and prediction performance. The experimental results show that our proposed model can achieve better prediction performance than the baseline model. In addition, this article concludes with a short description of some critical issues in traffic flow forecasting research.

The remainder of this article is as organized follows. Section 2 introduces related work on traffic flow forecasting and graph neural network models. Section 3 describes the traffic flow forecasting problem, and Section 4 illustrates our model (MSASGCN) in detail. Section 5 gives the experiment description and analysis of the results. Section 6 provides

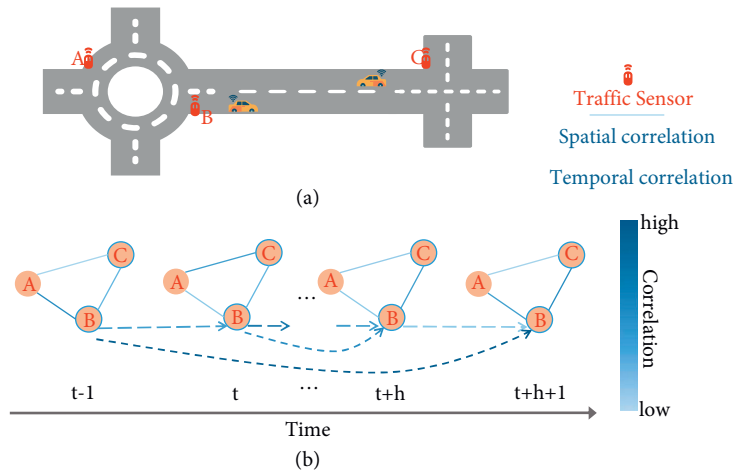


FIGURE 1: Dynamic spatial-temporal correlation in traffic forecasting. (a) Illustration of sensor distribution in the road network. (b) Dynamic spatial and temporal correlation.

some discussion to explain some potential problems in traffic flow forecasting. Finally, we conclude in Section 7.

2. Related Work

In this section, we first provide a summary of research on graph neural networks and then give an overview of recent traffic flow forecasting research.

2.1. Graph Neural Networks. Graph neural network is a novel model that captures graph dependencies through message passing between graph nodes to solve complex problems [12]. A new classification of graph neural networks was made in [10], respectively, recurrent GNNs (RecGNNs), convolutional GNNs (ConvGNNs), graph autoencoders (GAEs), and spatial-temporal GNNs (STGNNs). Reference [13] provided a comprehensive overview of the general design process, application classification, and some open problems for graph neural network models. Graph convolutional networks (GCNs) extend convolutional operations from traditional to graph data and are the foundation of many complex graph neural network models [14]. There are two categories of GCNs, spectral-based and spatial-based. The spectral-based approach introduced filters to define the graph convolution from the perspective of graph signal processing [15], while the spatial-based approach represents the graph convolution as aggregating feature information from the neighborhood [16]. To improve the effectiveness of long-range information dissemination, combining the gating mechanism of RNNs [17], such as GRU [18] or LSTM [19], with graph neural networks is an effective way. Gated graph neural networks (GGNNs) [20] use GRUs in forwarding propagation to expand RNNs in fixed time steps and compute gradients using a temporal backpropagation algorithm. As research on graph neural networks grows, combining them with other deep learning techniques is becoming a trend. Attention mechanisms [21] have been widely applied to sequence-based tasks, and combining attention mechanisms with graph neural networks yields better aggregation capabilities, integrating information from various components. Graph attention networks (GAT) [22] can efficiently handle the hidden states of nodes and perform well in tasks such as semi-supervised node classification. Apart from GAT, gated attention network (GAAN) [23] can assign different weights to different attention heads using additional soft gating computations. Graph neural network models can be widely used, but some methodological limitations, depth of model, dynamics, and heterogeneity need to be further explored.

2.2. Traffic Flow Forecasting. Research on traffic flow forecasting is an evolutionary process, [24] provided a detailed survey of urban traffic flow forecasting and analyzed some representative methods. The early traffic flow forecasting methods were mainly based on statistics, such as historical averages (HA) [25], time series methods [26], and Kalman filters [27]. Autoregressive integrated moving average (ARIMA) [28] and vector autoregressive (VAR) [29] are two classical methods that both have good performance for time series processing. Although these methods are helpful for traffic flow forecasting, all of them have some limitations. The road network's dynamic time dependence and spatial dependence in traffic data cannot be effectively exploited. Therefore, data-driven deep learning-based forecasting methods gradually become popular and bring good performance improvements. When a large amount of traffic data is accumulated, [4] used deep neural networks to explore the intrinsic relationships hidden in them and improve forecasting accuracy. Reference [30] proposed deep spatial-temporal convolutional network (DSTCN) to learn the spatial features of convolutional neural network and the temporal features of LSTM, but it needs to convert the traffic data into grid data. Reference [31] analyzed the data loss problem in traffic data collection and proposed a reconstruction method with low-rank matrix decomposition to reconstruct road traffic data accurately. Reference [32]

designed an enhanced graph convolutional network based on cross-attention fusion with better performance superiority and robustness.

Due to the characteristics of road networks, graph neural networks can directly model the road network and better capture spatial-temporal correlations. Seo et al. [33] proposed graph convolutional recurrent network (GCRN) to extract the topology of traffic networks and find dynamic patterns to optimize traffic forecasting. In [34], a combination of LSTM with graph convolutional networks is proposed, the streaming graph convolutional long short-term memory neural network (TGC-LSTM), which can address the dynamic time variation and complex spatial constraints of road networks. Reference [35] converted the dynamic traffic flow modeling into a diffusion process that can capture spatial dependencies using diffusion convolution operations. Reference [36] proposed the STGCN method, which can model multi-scale traffic networks, effectively capture comprehensive spatial-temporal correlations, and obtain better traffic forecasting results. Reference [37] designed a learnable position attention mechanism that can effectively aggregate information from adjacent roads and better exploit local and global spatial-temporal correlations. Guo et al. [38] used the attention mechanism for traffic flow prediction and proposed an attention mechanism spatial-temporal graph convolutional network (ASTGCN), which can extract temporal features more effectively to improve forecasting performance. Inspired by the low-rank representation and dynamic decomposition model, a low-rank dynamic decomposition model for traffic flow forecasting [39] is proposed for effective short-term traffic flow forecasting. An optimized graph convolutional recurrent network for traffic forecasting was proposed in [40] to improve the forecasting performance by learning the optimized graph data-driven during the training phase to reveal the potential relationships between road segments. Reference [41] considered road scalable and changing road networks, combining continuous learning with GNNs, and proposed the TrafficStream method. Reference [42] proposed a hierarchical graph convolutional network (HGCN) for traffic forecasting, which uses the road network's natural hierarchy to operate on micro and macro traffic maps to achieve traffic forecasting. Reference [43] proposed the spatial-temporal fusion graph neural network (STFGNN), which integrates the fusion graph module with the gated convolution module into one layer and can learn more spatiotemporal dependencies to handle long sequence situations. Reference [44] proposed the transformer network for traffic flow forecasting, which can jointly exploit dynamic directional spatial dependence and long-term temporal dependence to improve the forecasting accuracy. Considering the limitations of acquiring temporal and spatial dependencies separately, [45] designed a spatiotemporal synchronous modeling mechanism to construct the spatial-temporal synchronous graph convolutional network (STSGCN) to acquire complex local spatiotemporal correlations. Reference [46] proposed the STGSA, a spatial-temporal graph self-attention model, to learn graph-level spatial embeddings using graph self-attention layers and gated cyclic units integrated with RNN units to learn temporal embeddings. Reference [47] proposed the graph multi-attention network (GMAN) method, which applies an encoder-decoder structure and can solve the error propagation problem in forecasting. A multi-sensor data-correlated graph convolutional network model is proposed in [48], named MDCGCN, mainly designed with an adaptive benchmark mechanism and multi-sensor data-correlated convolutional blocks that can eliminate the differences between periodic data and capture dynamic spatial-temporal correlations. Reference [49] proposed a multi-range attention bicomponent graph convolutional network that uses bicomponent graph convolution to implement node and edge interaction aggregate information about different neighbors with a multi-range attention mechanism, and automatically learns the importance of different ranges. Therefore, the research of traffic forecasting approach based on graph neural networks can be effective for the time and spatial dependence and has some advantages for dynamic changes. Our work is based on the attention mechanism and combined with graph convolutional neural networks for traffic flow forecasting.

3. Preliminaries

In our work, traffic flow forecasting issues using the graph neural network approach require building the traffic network and defining the forecasting problem representation first. In addition, graph convolutional networks and attention mechanisms are the essential parts of our approach, and we give a brief description of them here.

3.1. Problem Statement. We first need to construct the traffic road network as a graph and illustrate the traffic flow forecasting problem. Define the traffic network as an undirected graph $G = (V, E, A)$, where V denotes a finite set of nodes corresponding to the observations of N sensors in the traffic network, $|V| = N$ nodes, and E is the set of edges to represent the connectivity of nodes. The graph structure of traffic data is shown in Figure 2, and each data node can be considered a graph signal defined on Graph G. If v_i and v_j are two nodes in V with a connection, then (v_i, v_j) is an edge in set E. These connections between nodes can be described by the adjacency matrix $A = (A_{ij})^{N\times N} \in \mathbb{R}^{N\times N}$. A is the adjacency matrix constructed based on the distance relationship between different sensor distributions, which can define and describe the relationships between different nodes in the graph G. The threshold Gaussian kernel approach is used to process the adjacency matrix A, where A_{ij} is the i, j − th element.

$$A_{ij} = \begin{cases} \exp\left(-\frac{d_{ij}^2}{\sigma^2}\right), & \text{if } \exp\left(-\frac{d_{ij}^2}{\sigma^2}\right) \geq \varepsilon \\ 0, & \text{otherwise} \end{cases}, \tag{1}$$

where d_{ij} denotes the distance between the sensors v_i and v_j, σ is the standard deviation of the distance between each

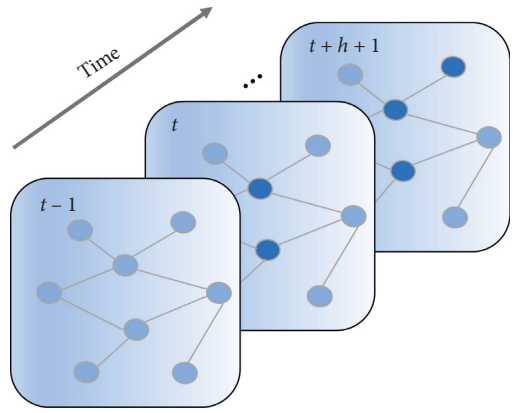


Figure 2: Graph-structured traffic data.

sensor node, and ε is a preset threshold value, which is taken as 0.1.

In the traffic network, each node on G samples F observations at the same frequency, which means that each node generates a feature vector of length F at each time step. We use $x_t^i \in \mathbb{R}^F$ as the value of all F features of node i at time t, and $X_t = (x_t^1, x_t^2, \ldots, x_t^N)^T \in \mathbb{R}^{N\times F}$ denotes the values of all F features of all nodes at time t. Therefore, we can denote the target feature value of all nodes predicted at time step t as $Y_t = (y_t^1, y_t^2, \ldots, y_t^N)^T \in \mathbb{R}^{N\times 1}$. The traffic flow forecasting problem is to predict future traffic conditions based on historical traffic data and the topology of the road network, which can be summarized as follows. The graph structure of traffic data is shown in Figure 2, and sensors are represented by nodes in the graph, allowing the acquisition of information on traffic conditions at different times and spaces.

$$f\left(X_{t-F+1}, X_{t-F+2}, \ldots, X_t, A\right) = \left(Y_{t+1}, Y_{t+2}, \ldots, Y_{t+M}\right), \quad (2)$$

where $X_{t-F+1}, X_{t-F+2}, \ldots, X_t$ denotes historical traffic data, and with the processing of function f, future traffic data series $Y_{t+1}, Y_{t+2}, \ldots, Y_{t+M}$ can be obtained, and A is the adjacency matrix of graph G.The crucial to the traffic forecasting problem is the need to find the function f, the traffic forecasting model, which maps the data series to the future traffic data series.

3.2. Graph Convolutional Networks. Graph convolutional network is a feature extractor for processing unstructured data with strong advantages for non-Euclidean graph-structured data processes. Suppose that a graph containing each node of K is given and the adjacency matrix $A \in \mathbb{R}^{k\times k}$ of this graph is obtained. The output of node i at layer l of the GCN is represented here as h_i^l, and h_i^0 represents the initial state of node i at the time of input to layer 1 of the GCN. For one l-layer GCN, $l \in [1, 2, \ldots, L]$, the final state of node i can be expressed as h_i^L. The following (3) is the computational procedure for the graph convolution of node i.

$$h_i^l = \sigma\left(\sum_{j=1}^{k} A_{ij} W^l h_i^{l-1} + b^l\right), \quad (3)$$

where W^l is the linear transformation weight, b^l is the deviation term, σ represents the activation function, commonly used activation functions such as *ReLU*.

3.3. Attention Mechanism. There are three matrix inputs, key $K \in \mathbb{R}^{n\times d_k}$, query $Q \in \mathbb{R}^{m\times d_q}$, and value $V \in \mathbb{R}^{m\times d_v}$, where n and m represent the lengths of these two inputs, and d_k and d_v represent the dimensional dimensions of the key and value. The attention mechanism is also set up with multiple heads, each of which can pay attention to different location information and learn different features. It computes the weighted sum by calculating the key and value dot product, and then normalizes it by *SoftMax*. And finally using the value projection (V) output. The concrete expression of the formula is as follows.

$$\text{Attention}\, Q, K, V = \text{SoftMax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V. \quad (4)$$

The shape of Q is $N \times d_q$, which represents the matrix consisting of query vectors of N nodes. The shape of K is $N \times d_k$, representing the matrix consisting of key vectors of N nodes. The shape of V is $N \times d_v$, representing the matrix consisting of value vectors of N nodes. In traffic flow forecasting with a spatial attention mechanism, it is necessary to aggregate node information in the spatial dimension, for which parameters are shared between different time steps.

3.4. Multi-Head Self-Attention Mechanism. The multi-head self-attention mechanism is mainly a process of multiple groups of self-attention on the original input sequence. It is worth noting that the process can be computed in parallel, improving the efficiency of feature extraction. Then, each group of self-attention results is concatenated, and then a linear transformation is performed to obtain the final output results.

$$\text{Multi Head}\, Q, K, V = \text{Concat}\left(\text{head}_1, \ldots, \text{head}_h\right)W^O. \quad (5)$$

$$\text{head}_i = \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right). \quad (6)$$

The specific calculation is expressed as shown in (5) and (6), where $W_i^Q \in \mathbb{R}^{p_q\times d_q}$, $W_i^K \in \mathbb{R}^{p_k\times d_k}$, and $W_i^V \in \mathbb{R}^{p_v\times d_v}$. The output of multi-head attention requires a linear transformation, which corresponds to the result after concatenate h heads, and therefore, its learnable parameter is $W_i^O \in \mathbb{R}^{p_o\times h_{pv}}$.

$$W_O \begin{bmatrix} \text{head}_1 \\ \vdots \\ \text{head}_n \end{bmatrix} \in \mathbb{R}^{p_O}. \quad (7)$$

Based on this design, each of the heads may focus on a different part of the input and can represent more complex functions than a simple weighted average.

4. Multi-Head Self-Attention Spatiotemporal Graph Convolutional Neural Network

This section describes our forecasting method by detailing the modules that make up the multi-head self-attention spatiotemporal graph convolutional network (MSASGCN) model. We provide a detailed description of the multi-head self-attention and graph convolution module, temporal convolution module, and the extended MSASGCN. It can effectively handle different temporal periods in historical data.

4.1. Architecture of Model. The architecture of our proposed model is illustrated in Figure 3. In addition, based on the research idea of ASTGCN [38], we added parallel sub-models of the same structure to improve the accuracy of prediction. Figure 4 illustrates the overall architecture with the addition of sub-models capturing the daily and weekly characteristics of traffic flow data. We also call the overall architecture an extension of the MSASGCN model, extended MSASGCN.

The multi-head self-attention mechanism and graph convolutional network are combined to capture local and global spatial dependencies, and the information obtained is fused using a gating mechanism. Meanwhile, the temporal convolution is used to capture the temporal dependence to get different influence levels at different times to improve forecasting accuracy. This structure is stacked to obtain more substantial processing power for long sequences or large-scale data. Extend MSASGCN model by adding weekly and daily periods to traffic flow forecasting. Each of these components has the same structure and has the same capability to handle spatial-temporal correlation.

MSA refers to the multi-head self-attention mechanism, GCN denotes graph convolutional network, Temp-Conv denotes temporal convolution, Gated Fusion denotes the gating mechanism to fuse spatial information, and Conv is the convolution operation. GCN is used to acquire local spatial information, and MSA is used to acquire global spatial correlations. The gating mechanism can fuse the extracted spatial correlations. A simple fully connected layer is used at the input layer to map the information to a high-dimensional space to improve the expressiveness of the model. Two convolution layers are used in the output layer for the decay of feature dimensions and the transformation of time series length. More details about the significant component modules of the model are described as follows.

4.2. Graph Convolution and Multi-Head Self-Attention Module. Traffic conditions on a road segment are influenced not only by the road segments that are spatially connected to it but also by other factors, and two road nodes that are far apart may still exhibit similar traffic patterns. The spatial correlation of traffic conditions can be influenced by the connectivity between road segments and geographic position attributes. Therefore, local spatial correlation and global correlation need to be considered. We use GCN to aggregate node information from local based on the connectivity

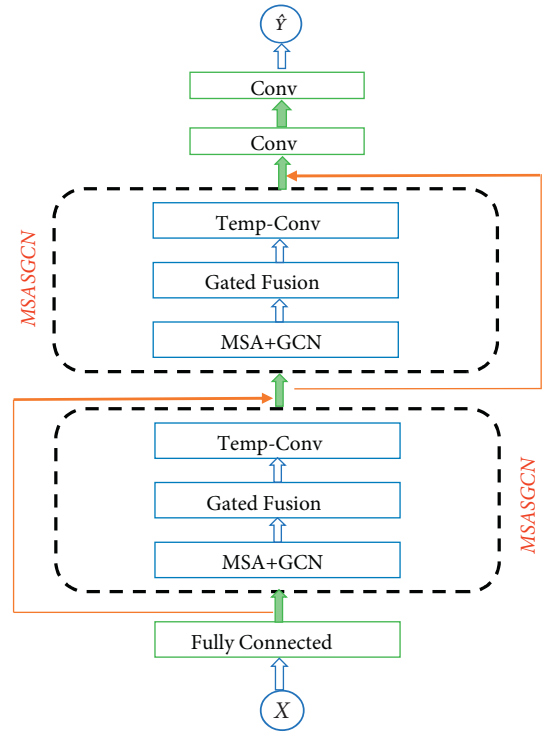


Figure 3: Architecture of MSASGCN. MSA: Multi-head self-attention; GCN: graph convolutional network; Temp-Conv: temporal convolution; Conv: convolution.

between roads and use the multi-head self-attention mechanism to aggregate the hidden global correlations.

Initially, the features of each node are considered as signals on the graph, and then spectral graph-based graph convolution is used to capture the spatial patterns in the traffic network. According to the spectral theory, the traffic graph is represented by the normalized Laplace matrix L in the graph. It can be defined as follows.

$$L = I_N - D^{-1/2} A D^{-1/2}, \tag{8}$$

where I_N is an $N \times N$ unit matrix, N denotes the number of nodes, and A is the adjacency matrix. D is the degree matrix, which is a diagonal matrix with diagonal elements of $D_{ii} = \sum_{j=1}^{N} A_{ij}$, and A_{ij} is the element of the $i-$th row and $j-$th column of the adjacency matrix A. The graph convolution can be defined as follows:

$$\theta_{*G} x \approx \sum_{K=0}^{K-1} \theta_K T_K(\tilde{L}) x, \tag{9}$$

where θ_{*G} denotes the graph convolution operation on the signal x in the graph G, $\tilde{L} = 2/\lambda_{\max} L - I_N$ is the normalized Laplace matrix after scaling, $\lambda_{\max}$ is the maximum feature value of L, θ_k is the coefficient of the $k-$th term of the Chebyshev polynomial, and T_K is $k-$th order Chebyshev polynomial. Graph convolution with Chebyshev polynomials is used to aggregate information from neighbor nodes to capture local spatial correlations.

To capture the global spatial correlation, it is necessary to consider the changes in the road network structure and the

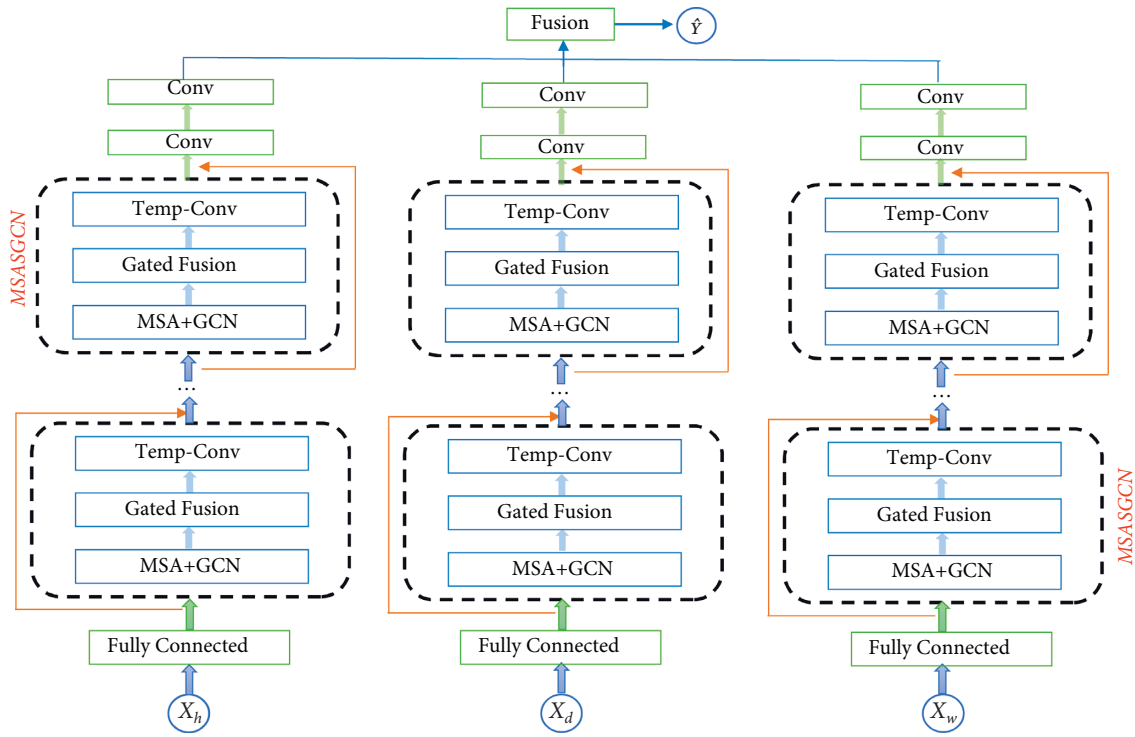


FIGURE 4: Extended MSASGCN architecture with weekly and daily period time dependencies. MSA : Multi-head self-attention; GCN : graph convolutional network; Temp-Conv : temporal convolution; Conv : convolution.

hidden spatial correlation in the road network, and we employ a multi-head self-attention mechanism to aggregate the information. Firstly, the feature vectors of each node are mapped with three different matrices W^Q, W^K, and W^V. Three vectors can be obtained, as described in preliminaries, as Query, Key, and Value. W^Q, W^K, and W^V are learnable parameters that are continuously optimized and updated during the training of the model. With the inner product of the Query vector of each node and the Key vector of all nodes, the *SoftMax* function can compress the vector to between 0 and 1. After normalization by the *SoftMax* function, the attention score of this node with all nodes can be obtained. The *SoftMax* function is defined as follows, where z_i denotes the i – th dimension of the vector and K denotes the dimension of the vector.

$$\text{SoftMax}(z_i) = \frac{e^{z_i}}{\sum_{K=1}^{K} e^{z_k}}. \tag{10}$$

We represent the attention mechanism in matrix form, which can be calculated using (4). A multi-head self-attention mechanism can aggregate information in several different feature subspaces simultaneously, with different subspaces expressing different implicit spatial correlations. The multi-head self-attention mechanism is performed by linearly mapping Query, Key, and Value n times (n is the number of heads) to get multiple sets of different subspace representations, then performing the attention mechanism on each set, and then stitching them together to get a final result by doing another linear mapping. The following equation can express the multi-head self-attention mechanism.

$$\begin{aligned} h_i &= \text{Attention}\left(XW_i^Q, XW_i^K, XW_i^V\right), \\ \text{Multihead} &= \text{Concat}\left(h_1, h_2, h_3, \ldots, h_n\right)W^O. \end{aligned} \tag{11}$$

The h_i denotes the output of the i – th group of the self-attention mechanism, n denotes the number of heads, Multihead denotes the output of the multi-head self-attention mechanism, Concat denotes the stitching operation on the tensor along the feature dimension that is the i – th group of linear mapping matrices, and W^O is the matrix that maps the result of the stitching. The spatial multi-head self-attention mechanism can learn the implied spatial correlation between nodes based on the features of each node in the input data. It is practical to capture their correlations in the spatial dimension using the multi-head self-attention mechanism. Meanwhile, it can capture when the topology of the road network changes because the attention scores among nodes are dynamically calculated based on the input. In addition, it is also able to capture the spatial correlation of the road network globally since the spatial self-attention aggregates the information of all nodes.

After obtaining local spatial correlation and global correlation, the information gained should be fused using a gating mechanism. The gating mechanism can be used to learn the importance of two kinds of spatial information and fuse the two kinds of information based on the learned weights, represented by the following equations.

$$\begin{aligned} g &= \sigma\left(H_{GCN}^{(l)}W_1 + H_{\text{Att}}^{(l)}W_2 + b\right), \\ H^{(l)} &= g \odot H_{GCN}^{(l)} + (1-g) \odot H_{\text{Att}}^{(l)}, \end{aligned} \tag{12}$$

where $H_{GCN}^{(l)}$ denotes the output of the first graph convolution module, $H_{\text{Att}}^{(l)}$ denotes the output of the first multi-head self-attention module, W_1 and W_2 are the mapping matrices, and b is the bias value. $\odot$ denotes the Hadamard product, where the corresponding position elements of the matrix are multiplied together. Moreover, g denotes the output of the gate, using the sigmoid activation function. $H^{(l)}$ is the result of the fusion of two spatial information.

4.3. Temporal Convolution Module. The improved convolution operation captures the temporal correlation in the time dimension. We combine dilated convolution with causal convolution methods for time series correlation prediction. Causal convolution can abstract the sequential problem and make the predicted value closer to the actual value, but it requires many layers or a large filter to increase the perceptual field of the convolution. Dilated convolution can make the filter apply to regions larger than filter length by skipping some inputs and expanding the receptive field without increasing the model complexity. The combination of these two convolution methods forms the temporal convolution (Temp-Conv) module, which facilitates the acquisition of long-term temporal correlation, improves processing efficiency, and can avoid information forgetting when the sequence is too long. Node i has an output value for the q channel at time t that can be expressed by the following equation.

$$Y_{i,t,q} = \sum_{k=1}^{\tau} \sum_{p=1}^{p} W_{k,p,q} * x_{i,t-d(k-1),p}, \tag{13}$$

where $W_{k,p,q}$ is the element in the convolution kernel, $x_{i,t-d(k-1),p}$ is the element of the input feature, p is the number of input channels, τ is the convolution kernel size, and d is the dilation rate. If the number of output channels is denoted by S, then S sets of convolution kernels are needed. The parameters of these S sets of convolution kernels can be expressed as a tensor $W^{\tau \times P \times S}$ of shape $\tau \times P \times S$, which are learnable parameters that are continuously updated iteratively by minimizing the loss function during the model training.

In Temp-Conv, to maintain the length of the input time series unchanged, a complementary 0 operation is required, but complementary 0 at both sides of the sequence will increase the length of the sequence, so the sequence ends will be cropped before proceeding to the next layer. Moreover, Temp-Conv contains multiple layers of dilated causal convolution, where the parameters of the convolution kernel are shared among different nodes. The tensor H_t of shape $N \times F \times P$ is used to denote the features of N nodes F time steps, and d denotes the dilated causal convolution operation with expansion rate d_*. Thus, the Temp-Conv operation for H_t can be shown as follows.

$$T = W_{d_*} H_t. \tag{14}$$

The T is result after convolution, and to expand the receptive field more, it is necessary to stack multiple layers of dilated causal convolution. Each layer's expansion rate increases exponentially, and the expansion rate of the $l-$th layer is $d_*^l = 2^{l-1}$. Hence, the output of the $l-$th layer can be expressed as follows.

$$T^l = \text{ReLu}\left(W_{d_l^*}^l Y^{l-1}\right). \tag{15}$$

Different layers get different outputs with different receptive fields, with shallow layers to obtain short-term temporal correlation and deep layers to obtain long-term temporal correlation. Then, the output features of each layer are concatenated according to their dimensions, and the output channels are transformed using a 1×1 convolutional layer to form the final output of Temp-Conv.

$$T = \text{Conv}\left(\text{Concat}\left(T^1, T^2, \ldots, T^c\right)\right), \tag{16}$$

where Concat denotes concatenation along the feature dimension, Conv denotes a 1×1 convolutional layer, and c denotes the number of layers of the dilated causal convolution.

4.4. Framework of Extended MSASGCN. The extended MSASGCN model is designed to model and process the dependencies of recent, daily, and weekly periods in historical data rather than a single time series input. As shown in Figure 5, we intercept three time series segments of lengths T_h, T_d, and T_w along the time axis as inputs for the recent, daily, and weekly period components, respectively, where T_h, T_d, and T_w are all multiples of the integer T_p, and T_p is the target time to be predicted.

We assume that the data sampling frequency is m times per day and the current time is t_0. The time series input for different time periods is denoted by X_h, X_d, X_w. The recent time segment is X_h, $X_h = \left\{X_{t_{0-T_h+1}}, X_{t_{0-T_h+2}}, \ldots, X_{t_0}\right\} \in \mathbb{R}^{N \times F \times T_h}$, a segment of the historical time series that is directly adjacent to the forecast period T_p. The daily periodic time segment is X_d, $X_d = \left\{X_{t_{0-(T_d/T_p-1)*q+1}}, \ldots, X_{t_{0-(T_d/T_p-1)*q+T_p}}, X_{t_{0-(T_d/T_p-1)*q+1}}, \ldots, X_{t_{0-(T_d/T_p-1)*q+T_p}}, \ldots, X_{t_{0-q+1}}, \ldots, X_{t_{0-q+T_p}}\right\} \in \mathbb{R}^{N \times F \times T_d}$, and consists of the same segments of the past few days as the prediction period. The weekly periodic time segment is X_w, $X_w = \left\{X_{t_{0-7*(T_w/T_p)*q+1}}, \ldots, X_{t_{0-7*(T_w/T_p)*q+T_p}}, X_{t_{0-7*(T_w/T_p-1)*q+1}}, \ldots, X_{t_{0-7*(T_w/T_p-1)*q+T_p}}, \ldots, X_{t_{0-7*q+1}}, \ldots, X_{t_{0-7*q+T_p}}\right\} \in \mathbb{R}^{N \times F \times T_w}$, and consists of time segments from the most recent weeks, with the same weekly attributes and time intervals as the forecast period.

Therefore, in the extended MSASGCN model architecture, the model components dealing with different periods have the same network structure, all with the same setup as in MSASGCN. Finally, the outputs of the three components are combined based on the parameter matrix to obtain the final prediction results, which can better predict the dynamic traffic flow. Extended MSASGCN is a multi-component fusion model, and the correlation of different periods needs to be handled. The sensitivity of the input traffic flow data to the components is inconsistent, so the sub-models within

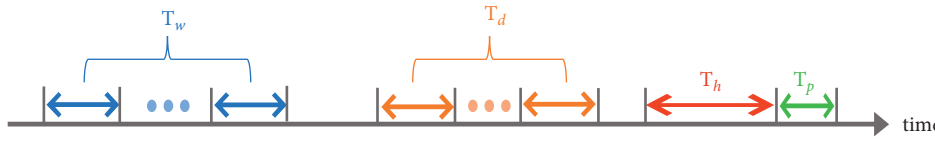


Figure 5: Example of time series segment input.

Table 1: Dataset profiles.

Attributes	PeMSD4	PeMSD8
Time periods	January-February 2018	July-August 2016
Detectors (nodes)	3848	1979
Distance information of sensors (edges)	340	295
Selected detectors	307	170
Sequence length	16,992	17,856
Selected features	3	3

the model have varying degrees of influence on the results. Different time components have different impact levels on each node and should be learned from the historical data and fused according to the different weights to obtain the forecasting results. The updated formula of the forecasting results is as follows.

$$\widehat{Y} = W_w \odot \widehat{Y_w} + W_d \odot \widehat{Y_d} + W_h \odot \widehat{Y_h}, \tag{17}$$

where $\odot$ is Hadamard product, and W_w, W_d, and W_h are learnable parameters that reflect the degree of influence of the three different time-dimensional components on the predicted target.

In some regions, traffic flows may have significant peaks in the morning or evening, making the output of the daily and weekly period components more critical. However, for some other regions, there is no prominent traffic period. Therefore, by fusing the outputs of different components from the above equation, we can obtain traffic flow forecasting results suitable for different regions or periods with different weights.

5. Experiments and Analysis

In this section, to verify the efficacy of our proposed model, we conduct experiments on two real datasets. Firstly, we describe and introduce the experimental datasets and the baseline method of comparison and then define the metrics for the experiments. Finally, the experimental settings and results analysis are given.

5.1. Datasets. PeMSD4 and PeMSD8 are two freeway traffic datasets from California on which we validate our model. The datasets are collected in real time every 30 seconds by the Caltrans Performance Measurement System (PeMS) [49, 50]. Traffic flow data are aggregated from the raw data into intervals of every 5 minutes. The system has over 39,000 detectors deployed on freeways in major metropolitan areas in California. The geographic information of the sensor stations is recorded in the dataset. Three traffic measures are considered in our experiments, including total flow, average speed, and average occupancy. Future traffic flow is our forecasting target. PeMSD4 and PeMSD8 are from different regions, for which details of the dataset are given in Table 1.

PeMSD4 is the San Francisco Bay Area traffic data and contains 3,848 detectors on 29 roads. The periods of this dataset span from January to February 2018. The first 50 days of data were chosen as the training set and the rest as the test set. PeMSD8 is the traffic data of San Bernardino from July to August 2016, which contains 1979 detectors on 8 roads. The first 50 days of data are used as a training set, and the last 12 days of data as the test set.

$$X' = \frac{X - \text{mean}(X)}{\sigma_x}. \tag{18}$$

The selection of detectors required the distance between adjacent detectors to be greater than 3.5 miles. In addition, the missing data are filled linearly. Data processing was performed using zero-mean normalization to make the training process more stable. As shown in (18), $\text{mean}(X)$ denotes the mean of the original data, σ_x is the standard deviation of the original data X, and X' is the normalized data.

5.2. Baselines and Experiment Metrics. We compare our model with the following baselines:

(i) HA [25]: Historical average, using the average of historical data to predict the next value.

(ii) VAR [29]: Vector autoregressive, capture pairwise relationships in traffic flow sequences for prediction.

(iii) ARIMA [28]: Autoregressive integrated moving average method is a classical time series forecasting algorithm that combines autoregressive models, moving average models, and differencing methods.

(iv) LSTM [19]: Long short-term memory network, a variant of RNN.

(v) GRU [18]: Gated recurrent unit network, a variant of RNN.

(vi) DCRNN [35]: Diffusion convolutional recurrent neural network, modeling traffic flow as a diffusion question process on a directed graph.

(vii) STGCN [36]: Spatial-temporal graph convolutional networks that model temporal and spatial dependencies.

(viii) ASTGCN [38]: Attention-based spatial-temporal graph convolutional networks, exploiting spatial-temporal attention mechanisms to model spatio-temporal correlations.

(ix) GeoMAN [51]: An attention-based multilevel recurrent neural network model for geo-aware time series prediction problems.

The baseline and MSASGCN method are compared with the same metrics in our experiments. We use the mean absolute error (MAE) and root mean square absolute error (RMSAE) as performance metrics for experimental evaluation, expressed in the following equations.

$$\begin{aligned} \mathrm{MAE} &= \frac{1}{n}\sum_{i=1}^{N}\left|X_i - X_i'\right|, \\ \mathrm{RMSE} &= \sqrt{\frac{1}{n}\sum_{i=1}^{N}\left(X_i - X_i'\right)^2}. \end{aligned} \tag{19}$$

In addition, we also compared with the mean absolute percentage error (MAPE) in some of the baselines, with the following definition.

$$\mathrm{MAPE} = \frac{100\%}{n}\sum_{i=1}^{N}\left|\frac{X_i - X_i'}{X_i}\right|, \tag{20}$$

where X_i and X_i' denote the $i-$th element in the true and predicted values, respectively, and n denotes the total number of elements.

5.3. Experiment Settings. We have implemented our model using the PyTorch deep learning framework. Future traffic flow is our forecasting target. On the one hand, we use the 1-hour historical traffic flow to forecast the future 1-hour traffic flow situation. Both the input time series and output time series lengths are set to 12, and the time series input length can be adjusted depending on the prediction time. We set the batch size to 64, the learning rate to 0.001, and the Chebyshev polynomial K to 3. The dimensions of the input layer, the implicit layer, and the output layer of the graph convolution module are taken to be 16, 64, and 128, and the input dimension, the dimension of key and value, and the number of heads of the multi-head self-attention module are taken as 16, 128, 128, and 4, respectively. The L_1 loss function is used to minimize the difference between the predicted results and the true value, and the L_1 loss for multi-step prediction is defined as follows.

$$L_1(W_\theta) = \sum_{i=t+1}^{t=P}\left|X_{:,i} - X_{:,i}'\right|. \tag{21}$$

Table 2: Average performance comparison of future 1-hour traffic flow prediction experiments.

Model	PeMSD4		PeMSD8	
	MAE	RMSE	MAE	RMSE
HA	36.76	54.14	29.52	44.03
VAR	33.76	51.73	21.41	31.21
ARIMA	32.11	68.13	24.04	43.30
LSTM	29.45	45.82	23.08	37.06
GRU	28.65	45.11	22.22	36.95
DCRNN	22.93	33.44	16.82	28.06
STGCN	25.15	38.29	17.51	27.09
ASTGCN	21.80	32.84	16.63	26.51
GeoMAN	23.64	37.84	17.84	28.91
MSASGCN (ours)	**21.22**	**32.09**	**16.23**	**26.24**

Bold is to highlight our experimental results.

The purpose of training the model is to continuously and iteratively update W_θ to minimize L_1, and $X_{:,i}$ and $X_{:,i}'$ denote the labels and predicted values of all nodes at time step i, respectively. On the other hand, we verified the efficiency of our method for traffic flow forecasting in different time prediction intervals. We conducted experiments to predict the future 10, 20, 30, and 40 minutes and analyzed the performance evaluation metrics.

5.4. Comparison and Result Analysis. We compared our model with the baseline approaches on PeMSD4 and PeMSD8. The average results of the future one-hour traffic flow prediction performance are shown in Table 2. It could be seen that our method achieves excellent performance in both datasets for MAE and RMSE evaluation metrics. In the case of traditional time series forecasting methods, they have limited analytical power to deal with spatial-temporal dependence, and the forecasting results are not very satisfactory.

Through comparison and analysis, it is clear that the performance of the deep learning-based approach is significantly better than that of the traditional approach. However, methods such as LSTM and GRU, which fail to handle temporal and spatial correlations effectively, also perform much weaker than methods that capture spatial-temporal correlations. Therefore, it can be concluded that the use of graph neural networks and their variants is effective in handling traffic flow forecasting. The algorithms GeoMAN and ASTGCN, which apply the attention mechanism, also outperform the other algorithms, demonstrating the effectiveness of using the attention mechanism to obtain spatial-temporal correlations.

Based on the experimental results obtained on the PeSMD4 dataset, a detailed description is given in Table 3. The analysis of the traffic flow prediction results for the next 10, 20, 30, and 40 minutes shows the effectiveness of our proposed method for different prediction intervals. Our method's MAE and RMSE evaluation metrics constantly change with increasing time intervals, which is a normal trend. Despite slight fluctuations, the performance is still within an average and the excellent band as the prediction interval increases.

Table 3: Average performance of experiments with different prediction intervals on PeMSD4.

Dataset	Model	Prediction interval (min)	MAE	RMSE
PeMSD4	MSASGCN	10	19.73	29.16
		20	20.42	30.24
		30	20.78	31.33
		40	21.01	31.82

Table 4: Average performance of experiments with different prediction intervals on PeMSD8.

Dataset	Model	Prediction interval (min)	MAE	RMSE
PeMSD8	MSASGCN	10	15.31	24.01
		20	15.72	25.24
		30	15.93	25.93
		40	16.01	26.13

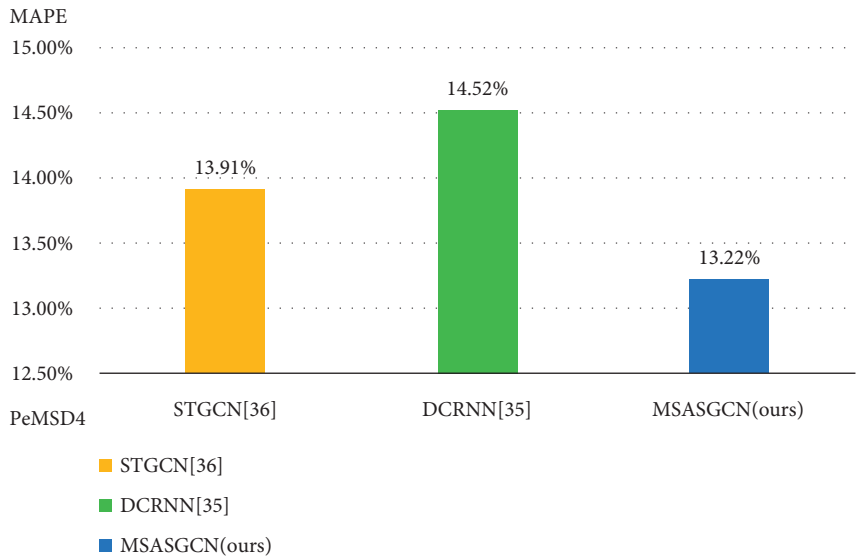


Figure 6: Comparison of different algorithms MAPE on PeMSD4.

Similarly, experimental results were obtained on the PeMSD8 dataset, detailed in Table 4. The experimental results on PeMSD8 are better than the PeMSD4 dataset, and although different datasets present different results, the overall trend is considered similar. MAE and RMSE performance evaluation metrics show that our proposed method can cope with traffic flow situations with different prediction intervals. According to the experimental results of different prediction intervals performed on two datasets, our proposed method can solve the traffic flow forecasting issue in the short or long term. Our proposed method has some advantages and reasonableness to capture the temporal and spatial characteristics nicely.

Moreover, we also compared the mean absolute percentage error (MAPE) of the MSASGCN model with STGCN and DCRNN, which also obtained better results than these two methods. This indicated that MSASGCN could better handle spatial-temporal correlations to capture the dynamically changing temporal and spatial dependence. The results obtained from the experiments on the PeMSD4 and PeMSD8 datasets are shown in Figures 6 and 7, respectively. Our method uses a multi-head self-attention mechanism to effectively fuse local and global spatial correlations, aggregate information from multiple nodes, and extract implicit information to improve prediction accuracy.

5.5. Validation of Module Effectiveness. To better represent the effectiveness of our proposed model, we modify the MSASGCN model by removing the MSA module and using only GCN to process the spatial dependencies, and all other experimental settings are consistent with the original

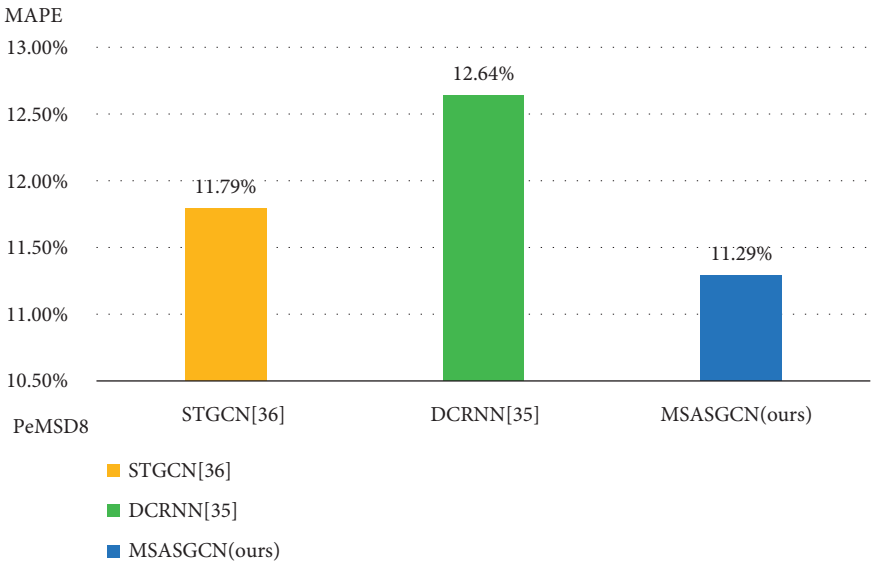


Figure 7: Comparison of different algorithms MAPE on PeMSD8.

Table 5: Introduction of the model name.

Name	Description
MSASGCN	Our original method
MSASGCN-s	Method for removing the multi-head self-attention mechanism

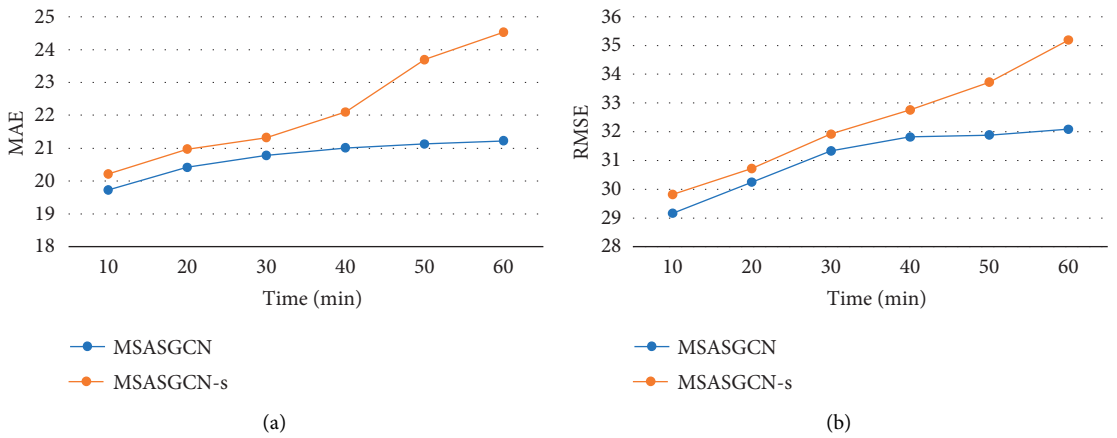


Figure 8: Module validation on the dataset PeMSD4. (a) The evolution of MAE. (b) The evolution of RMSE.

method. Table 5 shows our description of removing the MSA module. We have conducted experiments on PeMSD4 and PeMSD8 to remove the multi-head self-attention mechanism, respectively, and the MAE and RMSE metrics have a significant change in magnitude, which is not as good as the performance of the original MSASGCN method. The experimental results are shown in Figure 8 and Figure 9. Therefore, the multi-head self-attention mechanism plays an essential role in our method.

6. Discussion on Traffic Flow Forecasting

Although the performance of traffic flow forecasting has been significantly improved by applying graph neural networks, there are still some challenges for traffic flow forecasting. Reference [52] provided a summary of the challenges and future directions of traffic forecasting.

(1) From the data perspective, traffic data are heterogeneous and involve spatial-temporal factors and external factors. How well the heterogeneous data are handled can directly affect the forecasting accuracy. Data quality issues can also bring additional challenges.

(2) The timely accuracy of traffic forecasting is critical, but most graph neural network models require much computation and cannot make some real-time forecasting. Building a lightweight and general

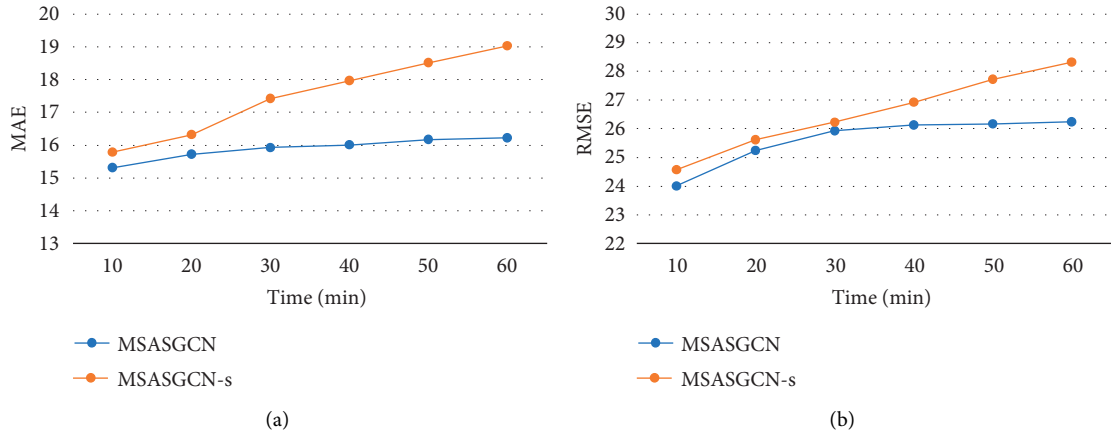


Figure 9: Module validation on the dataset PeMSD8. (a) The evolution of MAE. (b) The evolution of RMSE.

framework is a challenge for traffic forecasting and an essential requirement for intelligent transportation systems.

(3) Intelligent transportation systems need to integrate a different traffic information for analysis and processing simultaneously. Traffic forecasting models not only need to be able to process a specific task demand, but more importantly, they may process multiple tasks at the same time, which is a crucial challenge for multi-task forecasting.

(4) Privacy and security issues in traffic forecasting. The large-scale traffic data collection by IoT devices such as sensors has potential data security and privacy threats. The use of federated learning for graph neural network models in traffic forecasting in [53] is an approach of future interest, applying the distributed structure of federated learning, which allows some data protection.

7. Conclusions

In this article, we propose the multi-head self-attention spatiotemporal graph convolutional network (MSASGCN) model. Combining the multi-head self-attention mechanism with graph convolutional network can effectively handle the spatial-temporal correlation of traffic data. Our model can accommodate both the road network's dynamic time dependence and spatial dependence and performs better in capturing the spatial-temporal characteristics. Experiments on two real-world datasets showed that the prediction accuracy of our proposed model outperformed the baseline model. Our model verified the ability of processing spatial-temporal features simultaneously to construct the graph convolutional module and the spatiotemporal attention module to improve the prediction performance. In this article, we also summarize some of the challenges of traffic forecasting, and in the future, we will investigate the critical challenges of traffic flow forecasting intensively.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the General Program of Science and Technology Development Project of Beijing Municipal Education Commission of China (No. KM202110037002), Humanity and Social Science Research of Ministry of Education (No. 20YJCZH200), Research on Intelligent Inventory Optimization Decision Driven by Data (No.2021XJKY01), Beijing Intelligent Logistics System Collaborative Innovation Center Open Topic (No. BILSCIC-2019KF-05).

References

[1] A. Boukerche, Y. Tao, and P. Sun, "Artificial intelligence-based vehicular traffic flow prediction methods for supporting intelligent transportation systems," *Computer Networks*, vol. 182, Article ID 107484, 2020.

[2] D. A. Tedjopurnomo, Z. Bao, B. Zheng, F. Choudhury, and A. K. Qin, "A survey on modern deep neural network for traffic prediction: trends, methods and challenges," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 4, p. 1, 2020.

[3] I. Lana, J. Del Ser, M. Velez, and E. I. Vlahogianni, "Road traffic forecasting: recent advances and new challenges," *IEEE Intelligent Transportation Systems Magazine*, vol. 10, no. 2, pp. 93–109, 2018.

[4] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[5] E. Bolshinsky and R. Friedman, "Traffic Flow Forecast survey," No. CS Technion Report CS-2012-06, Computer Science Department, Technion, Haifa, Israel, 2012.

[6] M. Treiber and K. Arne, "Traffic flow dynamics," *Traffic Flow Dynamics: Data, Models and Simulation*, Springer-Verlag, Berlin Germany, pp. 983–1000, 2013.

[7] H. Zhu, Y. Xie, W. He et al., "A novel traffic flow forecasting method based on RNN-GCN and BRB," *Journal of Advanced Transportation*, vol. 202011 pages, Article ID 7586154, 2020.

[8] J. An, L. Guo, W. Liu et al., "IGAGCN: information geometry and attention-based spatiotemporal graph convolutional networks for traffic flow prediction," *Neural Networks*, vol. 143, pp. 355–367, 2021.

[9] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How Powerful Are Graph Neural Networks?," 2018, https://arxiv.org/abs/1810.00826.

[10] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[11] J. Ye, J. Zhao, K. Ye, and C. Xu, "How to build a graph-based deep learning architecture in traffic domain: a survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 5, pp. 3904–3924, 2022.

[12] F. Scarselli, M. Gori, and A. C. Tsoi, M. Hagenbuchner and G. Monfardini, The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[13] J. Zhou, G. Cui, S. Hu et al., "Graph neural networks: a review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[14] T. N. Kipf and M. Welling, "Semi-supervised Classification with Graph Convolutional Networks," 2016, https://arxiv.org/abs/1609.02907.

[15] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, "Spectral Networks and Locally Connected Networks on Graphs," 2013, https://arxiv.org/abs/1312.6203.

[16] A. Micheli, "Neural network for graphs: a contextual constructive approach," *IEEE Transactions on Neural Networks*, vol. 20, no. 3, pp. 498–511, 2009.

[17] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent Neural Network Regularization," 2014, https://arxiv.org/abs/1409.2329.

[18] K. Cho, B. Van Merriënboer, C. Gulcehre et al., "Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation," 2014, https://arxiv.org/abs/1406.1078.

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

[20] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated Graph Sequence Neural Networks," 2015, https://arxiv.org/abs/1511.05493.

[21] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[22] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," 2017, https://arxiv.org/abs/1710.10903.

[23] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung, "Gaan: Gated Attention Networks for Learning on Large and Spatiotemporal Graphs," 2018, https://arxiv.org/abs/1803.07294.

[24] P. Xie, T. Li, J. Liu, S. Du, X. Yang, and J. Zhang, "Urban flow prediction from spatiotemporal data using machine learning: a survey," *Information Fusion*, vol. 59, pp. 1–12, 2020.

[25] A. G. Hobeika and C. K. Kim, "Traffic-flow-prediction systems based on upstream traffic," in *Proceedings of the VNIS'94 - 1994 Vehicle Navigation and Information Systems Conference*, pp. 345–350, Okohama, Japan, September 1994.

[26] M. S. Ahmed and A. R. Cook, *Analysis of Freeway Traffic Time-Series Data by Using Box-Jenkins Techniques*, Transportation Research Board, Washington, D.C, USA, 1979.

[27] I. Okutani and Y. J. Stephanedes, "Dynamic prediction of traffic volume through Kalman filtering theory," *Transportation Research Part B: Methodological*, vol. 18, no. 1, pp. 1–11, 1984.

[28] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: theoretical basis and empirical results," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[29] E. Zivot and J. Wang, "Vector Autoregressive Models for Multivariate Time Series," *Modeling Financial Time Series with S-Plus®*, pp. 385–429, Springer, Berlin, Germany, 2006.

[30] J. Zhang, Y. Zheng, and D. Qi, "Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction," in *Proceedings of the Thirty-first AAAI conference on artificial intelligence*, pp. 1655–1661, San Francisco, CA, USA, February 2017.

[31] Y. Wang, Y. Zhang, X. Piao, H. Liu, and K. Zhang, "Traffic data reconstruction via adaptive spatial-temporal correlations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 4, pp. 1531–1543, April 2019.

[32] G. Huo, Y. Zhang, J. Gao, B. Wang, Y. Hu, and B. Yin, "CaEGCN: cross-attention fusion based enhanced graph convolutional network for clustering," *IEEE Transactions on Knowledge and Data Engineering*, p. 1, 2021.

[33] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson, "Structured sequence modeling with graph convolutional recurrent networks," *Structured Sequence Modeling with Graph Convolutional Recurrent Networks*, Springer, Cham, Switzerland, pp. 362–373, 2018.

[34] 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 Transactions on Intelligent Transportation Systems*, vol. 21, no. 11, pp. 4883–4894, Nov. 2020.

[35] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting," 2017, https://arxiv.org/abs/1707.01926.

[36] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting," 2017, https://arxiv.org/abs/1709.04875.

[37] X. Wang, Y. Ma, Y. Wang et al., "Traffic flow prediction via spatial temporal graph neural network," in *Proceedings of the Web Conference 2020*, pp. 1082–1092, Taipei Taiwan, April 2020.

[38] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 922–929, Honolulu, Hawaii, February 2019.

[39] Y. Yu, Y. Zhang, S. Qian, S. Wang, Y. Hu, and B. Yin, "A low rank dynamic mode decomposition model for short-term traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 10, pp. 6547–6560, 2021.

[40] K. Guo, Y. Hu, Z. Qian et al., "Optimized graph convolution recurrent neural network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 1138–1149, 2021.

[41] X. Chen, J. Wang, and K. Xie, "TrafficStream: A Streaming Traffic Flow Forecasting Framework Based on Graph Neural Networks and Continual Learning," 2021, https://arxiv.org/abs/2106.06273.

[42] K. Guo, Y. Hu, and Y. Sun, "Hierarchical graph convolution networks for traffic forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 1, pp. 151–159, February 2021.

[43] M. Li and Z. Zhu, "Spatial-temporal fusion graph neural networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 5, pp. 4189–4196, February 2021.

[44] M. Xu, W. Dai, C. Liu et al., "Spatial-temporal Transformer Networks for Traffic Flow Forecasting," 2020, https://arxiv.org/abs/2001.02908.

[45] 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 1, pp. 914–921, New York, NY, USA, February 2020.

[46] Z. Kang, H. Xu, J. Hu, and X. Pei, "Learning Dynamic Graph Embedding for Traffic Flow Forecasting: A Graph Self-Attentive Method," in *Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference*, pp. 2570–2576, ITSC, Auckland, New Zealand, October 2019.

[47] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: a graph multi-attention network for traffic prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 1, pp. 1234–1241, 2020.

[48] W. Li, X. Wang, Y. Zhang, and Q. Wu, "Traffic flow prediction over muti-sensor data correlation with graph convolution network," *Neurocomputing*, vol. 427, pp. 50–63, 2021.

[49] W. Chen, L. Chen, Y. Xie, W. Cao, Y. Gao, and X. Feng, "Multi-range attentive bicomponent graph convolutional network for traffic forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 4, pp. 3529–3536, New York, NY, USA, February 2020.

[50] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, "Freeway performance measurement system: mining loop detector data," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1748, no. 1, pp. 96–102, 2001.

[51] Y. Liang, S. Ke, J. Zhang, X. Yi, and Y. Zheng, "Geoman: multi-level attention networks for geo-sensory time series prediction," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, vol. 2018, pp. 3248–3434, Stockholm, Sweden, July 2018.

[52] W. Jiang and J. Luo, "Graph Neural Network for Traffic Forecasting: A Survey," 2021, https://arxiv.org/abs/2101.11174.

[53] C. Meng, S. Rambhatla, and Y. Liu, "Cross-node federated graph neural network for spatio-temporal data modeling," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (KDD '21)*, pp. 1202–1211, Singapore, August 2021.

Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 2238095, 10 pages
https://doi.org/10.1155/2022/2238095

Research Article

The Improvement of Automated Crack Segmentation on Concrete Pavement with Graph Network

Jiang Chen, Ye Yuan, Hong Lang, Shuo Ding, and Jian John Lu

The Key Laboratory of Road and Traffic Engineering of the Ministry of Education, Tongji University, Shanghai 201804, China

Correspondence should be addressed to Hong Lang; honglang@tongji.edu.cn

Received 25 January 2022; Revised 7 April 2022; Accepted 23 May 2022; Published 11 June 2022

Academic Editor: Yong Zhang

Crack is a common concrete pavement distress that will deteriorate into severe problems without timely repair, which means the automated detection of pavement crack is essential for pavement maintenance. However, automatic crack detection and segmentation remain challenging due to the complex pavement condition. Recent research on pavement crack detection based on deep learning has laid a good foundation for automated crack segmentation, but there can still be improvements. This paper proposes an automatic concrete pavement crack segmentation framework with enhanced graph network branch. First, the nodes of the graph and nodes' attributions are generated based on the image dividing. The edges of the graph are determined based on Gaussian distribution. Then, the graph from the image is input into the graph branch. The graph feature map of the graph branch output is fused with the image feature map of the encoder and then enters the decoder to recover the image resolution to obtain the crack segmentation result. Finally, the method is tested on a self-built 3D concrete pavement crack dataset. The proposed method achieves the highest F1 and IoU (Intersection over Union) in the comparison experiments. And the graph branch addition improves 0.08 on F1 and 0.06 on IoU compared with U-Net.

1. Introduction

Road infrastructure is an essential asset for a country, and it can contribute to the economic development and bring significant social benefits. Road density is adopted as a rating criterion by the World Bank to evaluate low-income, middle-income, and high-income economies [1]. Concrete pavement is one of the main pavement types. The concrete pavement in the United States highway network accounts for about 49 percent, and in Belgium they occupy 50 percent. However, due to the severe traffic loading and the variable environment, concrete pavement distress always appears over the road operation time. Maintaining an acceptable level of service for the whole road network is a challenge to the transportation agency officials.

Pavement distress evaluation is the essential work for pavement maintenance. The transportation agency officials regard pavement data collection as a regular work to grasp the evolution of road conditions and make opportune work to stop the deterioration of the distress. Efficient pavement condition inspections and reasonable repair strategies can lead to a significant reduction in life-cycle pavement maintenance cost [2].

Pavement distress evaluation has undergone a long period of development with the continuous advancement of computer technology. Traditional distress inspections are based on the manual visual survey, which is time-consuming. After that, a collection vehicle equipped with a high-speed digital camera is invented to acquire the pavement surface images at a high speed [3, 4]. This method causes little influence on traffic operation and is widely accepted by the transportation agency officials. Recently, 3D technology has attracted much attention. Compared with the 2D technology, the pixels in the image captured by 3D technology describe the depth change relative to the reference surface. Therefore, the 3D images of concrete pavement can reduce the influence of surface oil and lighting conditions and present more information on pavement distress [5–7].

Once the concrete pavement surface images are obtained, the processing can be conducted to detect pavement distress using various algorithms. Over the past decade, there have been sufficient methods based on computer vision

proposed to detect pavement distress automatically and achieve excellent results, such as methods based on threshold [8, 9], methods based on edge detection [10, 11]. However, the effects of most methods are easily influenced by different pavement detection environments due to the feature extraction based on manual design. Therefore, semiautomatic pattern to pavement crack detection is in current practice. In the semiautomatic approach, crack detection algorithms are applied first, and then a series of human interventions are conducted to manually adjust the crack information and incorrect results. It is also time-consuming.

Recently, with the success of deep learning methods, especially Convolutional Neural Network (CNN) in computer vision tasks [12], applying deep learning to automatic pavement distress detection has become a spotlight. CNN can automatically extract the features of objects in the images with a structure similar to the human brain compared to manually designed feature extraction in traditional methods. In the current application, the deep learning based pavement crack detection method can be divided into three categories, e.g., patch classification [13], object detection [14], and semantic segmentation [15]. The patch classification methods divide the pavement image into several blocks of the same size and then classify each block into the corresponding category. The object detection methods frame the crack in the pavement image using a bounding box. Furthermore, the semantic segmentation methods classify each pixel in the pavement image. Hence, the semantic segmentation methods can achieve the pixel-level inspection result and obtain more detailed characteristics of distress, such as precise length and area of the crack. However, the low accuracy and high false positives of the semantic segmentation when the pavement conditions change limit the promotion in practice.

Concrete pavement is a rigid pavement, while asphalt pavement is a flexible pavement. Cracks have different characteristics in different pavements. The crack on the concrete pavement has a more obvious and complex boundary compared to the crack on the asphalt pavement and often has a jump down in pavement depth changes. And joints between the concrete blocks and the indentations in concrete pavement cause the more complex surface texture than asphalt pavement. The complex texture will bring interference to the identification of crack. Most of the research on pavement distress and the pavement datasets constructed nowadays focus on asphalt pavement distress [16, 17]. The effectiveness of transferring the method applied for asphalt pavement distress detection to the concrete pavement is substantially reduced. It is necessary to establish a concrete pavement distress dataset and a method to detect cracks in the concrete pavement.

In this work, a feature extraction branch based on graph neural network is added to a typically semantic segmentation network to form a new end-to-end network structure. And experiments are conducted on concrete pavement crack segmentation to evaluate the performance improvement. In this regard, the main contributions of this work can be summarized as follows:

(i) A semantic segmentation network framework with graph neural network branch is proposed to segment the concrete pavement crack. The performance of crack segmentation is significantly improved based on the original segmentation network. In addition, the inclusion of the graph branch improves the continuity of crack segmentation.

(ii) A generation method to convert images into graphs is designed, which enriches the feature map dimension of images.

(iii) A new dataset of 3D concrete pavement crack images is established and applied to evaluate the proposed network.

The rest of this paper is organized as follows. Section 2 describes the related research on pavement crack detection and the development of graph neural networks. Section 3 introduces the detailed architecture of the segmentation network with graph neural network branch. Section 4 represents the experiment setting. Section 5 discusses the experiment results. Finally, Section 6 concludes the work and presents the findings of this research.

2. Related Work

2.1. Semantic Segmentation. The semantic segmentation method is the classification of the category for each pixel in the image. The fully convolution network (FCN) proposed by Long et al. is the milestone for semantic segmentation based on deep learning [18]. They apply a 1×1 convolution layer instead of a fully connected layer as a classifier. Hence, the output of the network is transformed from a vector to a matrix, where the value of each pixel represents the probability that the corresponding pixel of the input image belongs to a specific class. Moreover, an encoder-decoder structure is added to the network design for semantic segmentation [19]. This structure can improve computational efficiency and reduce the overfitting problem. In simple terms, the encoder process extracts the feature of the input image by convolutional computation and pooling, and the decoder process restores the feature map to a matrix with the same size as the input by upsampling.

The semantic segmentation method used in pavement crack detection is to classify each pixel of the image into two categories, crack pixel and none crack pixel. Due to the easy obtaining of pavement crack's geometric characteristics such as length, area, and bounding box, semantic segmentation is widely popular. Yang et al. offer an FCN-based method to segment crack pixel in the pavement image and acquire the length, width, and mean width of crack [20]. Liu et al. develop a U-Net based model to segment concrete cracks [15]. Qu et al. improve crack segmentation performance with attention mechanism and apply their model in asphalt pavement and concrete pavement crack segmentation [21].

2.2. Graph Neural Network. Graphs are all around us. The graph has two elements consisting of nodes and edges, which represent a set of objects and the connection between them,

respectively [22]. Anything with a connection relationship can be described as a graph, e.g., image, text, and social network. Motivated by the neural network and deep learning, new significant operations have rapidly developed over the past few years to handle the complexity of graph data. Compared to the other networks, the graph neural networks (GNNs) need two vectors or matrices as input, representing the node and the edge attributes of the graph. Sanchez-Lengeling et al. propose the Spectral network and use a learnable diagonal matrix as the filter to process graph [23]. However, the operation is computationally inefficient and the filter is nonspatially localized. Inspired by the 2D convolution in image computing, Kipf and Welling develop the graph convolution operation to alleviate the overfitting problem and promote the computationally efficiency [24]. To address the large-scale graph computation problem, spatial approaches based on the graph convolution are developed to adjust to different sized neighborhoods and maintain the local invariance [25–27]. Zhang et al. improve the graph network's ability to extract node relationships by adding a cross attention module and apply the graph network to metro passenger flow prediction, achieving state-of-the-arts performance [28, 29].

2.3. 3D Technology in Pavement Detection. The 2D images describe the grey-scale feature of the pavement surface, which is the most used method in traditional pavement distress detection. However, detection on 2D images is susceptible to surface oil, pavement texture, lighting condition, etc. 3D images describe the depth changes of pavement surface, which can overcome the shortage above and usually present more detail of distress like depth. With the development of 3D sensors and image processing technology, the potential of 3D measurement in pavement detection has earned widespread attention. 3D technologies applied in pavement inspection include 3D structure light [30, 31], laser scanning [32], and binocular stereo vision [33]. There have been attempts to combine 3D techniques and deep learning methods for pavement inspection. Zhang et al. propose a model called CrackNet based on a convolution neural network to detect crack on 3D asphalt pavement image and significantly outperforms the traditional approaches in terms of F-measure [34]. Lang et al. develop a multiscale clustering model for detecting different types of cracks, including linear and netted types on the 3D pavement surface [35]. However, most of the existing studies have focused on detecting asphalt pavements and less on the detection of concrete pavement distress. In this work, a concrete pavement dataset with 3D images is built for pavement crack detection and validates the accuracy of our proposed method.

3. Methodology

In this section, the graph neural network feature extraction branch and the main body of semantic segmentation are first introduced, respectively. Then, the proposed network structure for crack detection on the concrete pavement is described.

3.1. Graph Neural Network Branch. Adding new feature extraction branches to enrich feature map information is a common approach to improving the accuracy of semantic segmentation networks. Image is similar to graph data, and each pixel in the image can be regarded as a node in the graph. The relationship between every pixel can be considered as an edge in the graph, as shown in Figure 1. Note that the generation of nodes and edges in a graph is designed according to the realistic task. In this work, the node and its attribute generate from a group of pixels in a region. The image with the size of 512×512 is divided into 1,024 (32×32) patches with 16×16 size. Each patch forms a node, and the mean value of the pixels in the patch is calculated as the attribution of the node. In general, the neighbors of a node can be the nearest neighbor node or the node in the interval, even all other nodes. In this work, we assume that each node connects to the node with the interval of D. The connection means that the nodes attribution can be transferred by the edge in the graph neural network. There is an instance to describe the neighbors of a node when the D is 2, as illustrated in Figure 2. The edge information will be respected by the adjacent matrix as the input to the graph neural network.

The transform methods from image to graph including the node generation and the edge generation are determined. Then, the feature extraction branch is described. This part of the work is related to the GraphSAGE (Graph SAmple and aggreGatE) method proposed by Hamilton et al. [25]. GraphSAGE is an inductive learning framework that can use the vertice attribution to generate unknown node embedding. The feature extraction based on GraphSAGE can be divided into three steps as illustrated in Figure 3. In the first step, sample the local neighborhood and generate the embeddings for nodes. Considering the computing efficiency, sampling range K is proposed to control the number of neighboring nodes sampled. According to the edge generation, a node has at least 5 neighbors, and at most 12 nodes in this work. When K is larger than the number of node neighbors, the sampling with put-back is completed until K nodes are sampled, when not, the sampling without put-back is used. In the second step, choose an aggregator and aggregate feature information from neighbors. Since the neighbors of the node in the graph are disordered, the aggregator function needs to be symmetric, which means that the output of the function remains the same when the order of the inputs changes. A mean aggregator is applied in this step to connect the node and its neighbors and calculate the mean value of each dimension of the node attribution vectors. Through an activate function layer, the target representing vector of node is obtained. This step is equivalent to the convolutional computing for feature extraction in a convolutional neural network. In the third step, the aggregate information output from step 2 can be applied to the downstream tasks such as classification and prediction.

3.2. Semantic Segmentation Main Body. The semantic segmentation task acquires a combination of local information based on high-resolution images and global information based on low resolution to classify each pixel. Common

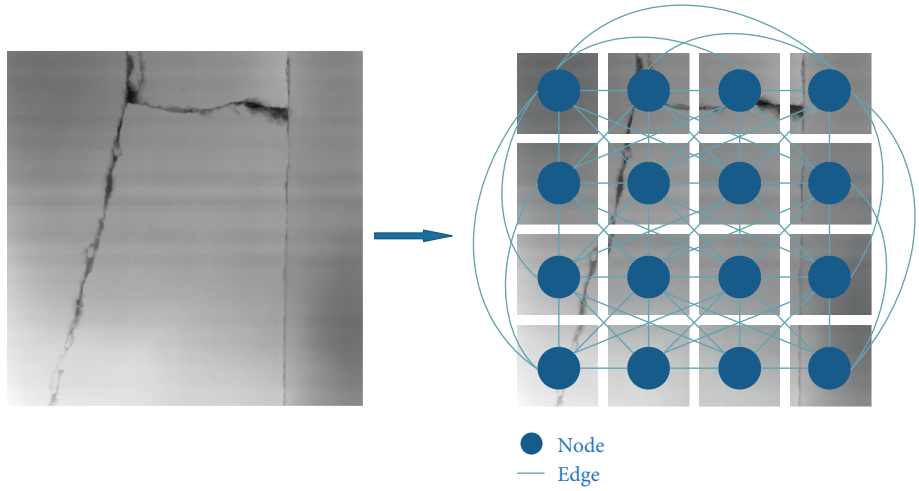


Figure 1: Schematic of transforming an image into a graph.

segmentation networks utilize an encoder-decoder framework to obtain the features of different levels of different scales. The main body of the network structure proposed in this work is related to U-Net with an encoder and decoder framework [19]. The U-Net structure is simple and easy to modify, as shown in Figure 4. Symmetry is one of the characteristics of U-Net. The left of Figure 4 is the encoder, while the right is the decoder. The encoder is responsible for the extraction of the image feature and the decoder is responsible for recovering the image resolution. In the encoder, there are five encoding blocks. Each encoding block consists of one convolutional layer with kernel size of 3×3 (deep blue arrow) and one maximum pooling layer (red arrow). The rotated numbers represent the width and the height of the images or the feature maps, while the normal number represents the number of channels. The convolutional layers do not change the sizes and channel numbers, but the maximum pooling layers do. After the maximum pooling layer, the output is halved in width and height but doubled in the number of channels compared to the input. The flow of the size and channel number is listed in order, $512 \times 512 \times 16$, $256 \times 256 \times 32$, $128 \times 128 \times 64$, $64 \times 64 \times 128$, $32 \times 32 \times 256$, $16 \times 16 \times 512$ (weight $\times$ height $\times$ channel number). In the decoder, there are five decoding blocks correspondingly. Each decoding block consists of one convolutional layer and one deconvolution layer (light blue arrow). The effect of the convolutional layer in the decoder is the same as in the encoder. The deconvolution layer is to recover the image resolution in the contrast to the pooling layers. After the deconvolution layer, the output is doubled in width and height but halved in channel number compared to the input. The flow of the size and channel number in 'decoder is list in order, $16 \times 16 \times 512$, $32 \times 32 \times 256$, $64 \times 64 \times 128$, $128 \times 128 \times 64$, $256 \times 256 \times 32$, $512 \times 512 \times 16$ (weight $\times$ height $\times$ channel number). Different from the encoding block, the input in the decoding block is not only the output of the upper decoding block but also includes the output from the encoding block at the same level. This design facilitates the integration of high-resolution detailed features and the low-resolution semantic feature to promote performance. And the ReLu activate function is added after each convolutional layer to boost the nonlinearity of the network. In the end, a convolutional layer with a kernel size of 1×1 is applied to classify the pixel into the corresponding class. The size of the output is 512×512 the same as the input. The number of channels is 2. The value of each pixel in output represents the probability of the corresponding pixel in the input belonging to a certain category.

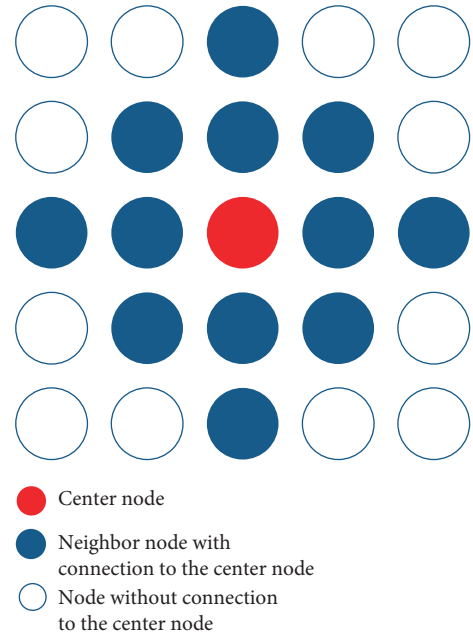


Figure 2: Nodes and neighbor nodes. Blue represents the node has edge to the red node. White represents the node has no edge to the red node.

3.3. Network Structure. By integrating the graph network branch in the U-Net, the network is developed, namely, GA-Unet (Graph branch Added Unet). The network structure is illustrated in Figure 5. Firstly, the input image is processed through the encoder in semantic segmentation main body

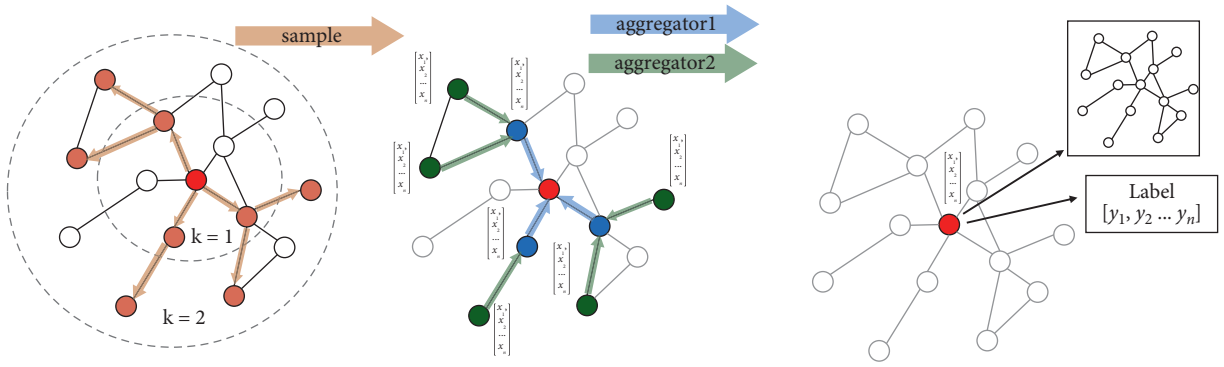


Figure 3: The process of feature extraction based on GraphSAGE(25). (a) Sample neighborhood, (b) Aggregate feature information from neighbors, (c) Predict graph context and label using aggregated information.

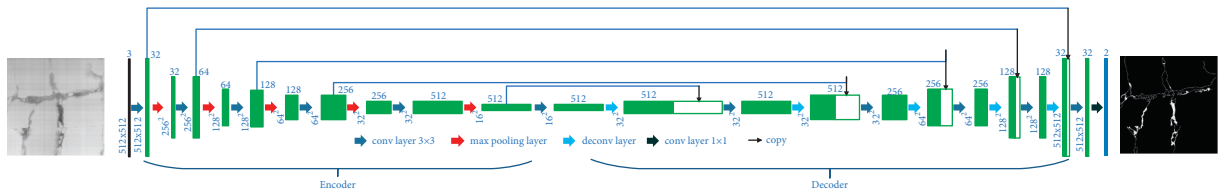


Figure 4: The structure of U-Net.

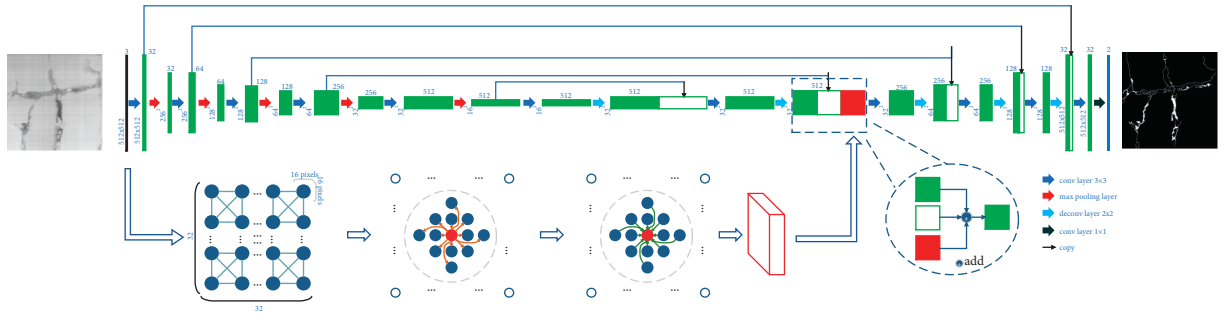


Figure 5: The structure of the whole network. The top-line describes the processing of the image in the U-Net. The number above the feature map presents the number of channels. The number beside the feature presents the size of the feature map. The following line describes the processing in the graph neural network branch. And the feature map in the U-Net and the graph network branch are fused after the first decoding.

and the graph network branch, respectively. In graph network branch, the image input with 512×512 size is transferred into a graph with 1024 (32×32) nodes and 5,174 edges. Through sampling, aggregating, and predicting in the graph network branch, the feature map with the size of 32×32 is obtained at the graph level. Then, the feature maps obtained by the graph branch and by the encoder are fused after the first decoding block and input to the subsequent decoding blocks.

4. Experiments and Results

The proposed method was evaluated on the self-captured concrete pavement crack dataset, namely, the CPC dataset. The performance of the proposed graph network branch was evaluated by comparing it with U-Net methods. Furthermore, the proposed network was implemented using Pytorch on a personal computer with an Intel i7-11700K CPU @3.60 GHz, 64 GB memory, and an NVIDIA RTX3090 GPU with 24 GB memory.

4.1. CPC Dataset. The CPC dataset consisting of 3D concrete pavement crack images is built to train and test the proposed network in this work. The detection vehicle can scan the pavement at different collection speeds ranging from 35 to 100 km/h (20 to 60 mi/h). The pixel resolutions of the 3D pavement data are both 2048×2048, covering pavement surfaces of more than 2 m in width and 2 m in length. Moreover, the CPC dataset contains images with various changes in pavement conditions aiming at the accuracy of crack recognition. There is no overlap between any

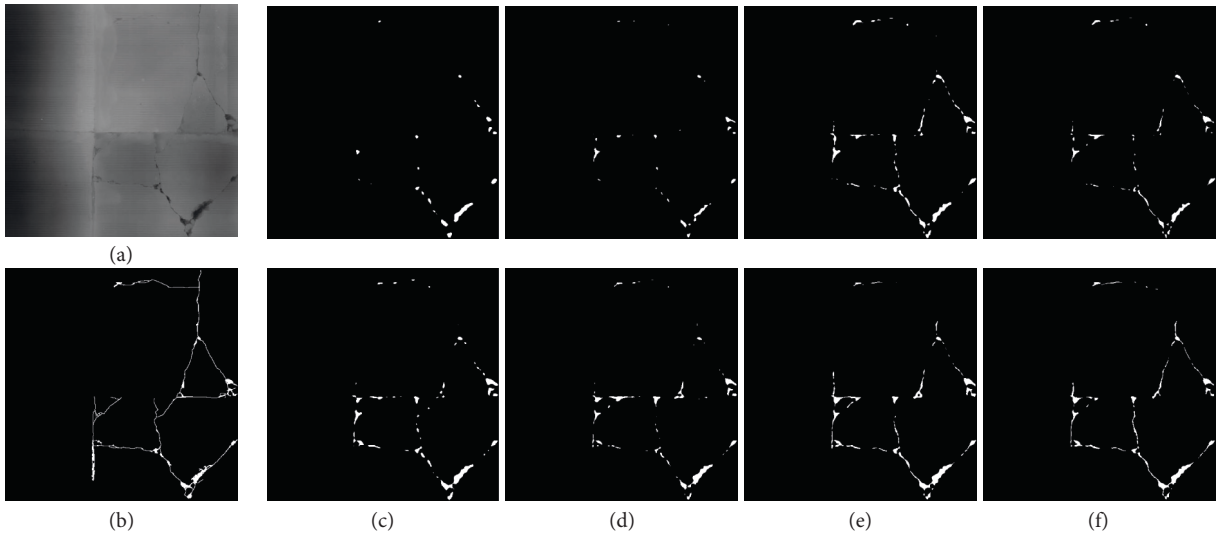


Figure 6: The result of GA-Unet and U-Net at different epoch. (a) original pavement image. (b) ground truth. (c) the results at epoch 50. (d) the results at epoch 100. (e) the results at epoch 200. (f) the results at epoch 300. In (c), (d), (e), (f) images, the upper line is the result of U-Net, and the down line is the results of GA-Unet.

Table 1: The comparison of the test results on the concrete pavement image dataset.

	Precision	Recall	F1	IoU
AutoEncoder	0.38	0.53	0.42	0.28
PSPNet [37]	0.65	0.44	0.52	0.35
U-Net [19]	**0.67**	0.37	0.45	0.31
KiU-Net [38]	0.38	**0.69**	0.46	0.31
GA-Unet(ours)	0.63	0.49	**0.53**	**0.37**

2 images, and no more than 50 images are from the same pavement section. The 3D pavement image input into the network will be resized into 512×512 to reduce the computational effort. The final dataset consists of 1,452 images. After collection, the labeling work is conducted. The ground truth of cracks on all pavement images is manually labeled on pixel level by our research team. To ensure the accuracy of the ground truth, three rounds of labeling work were applied. In the first round, several well-trained operators label cracks manually on the pavement image. In the second round, the operators in the first round exchange their labeling results and check them. In the third round, the experts further confirm the availability of ground truth in each pavement image of the entire dataset. And finally we get accurate ground truth images. The ground truth image is a binary image, in which 0 represents the pavement background pixel and 1 represents the crack pixel. Then, the CPC dataset with ground truth is divided into two parts, 1,352 images for training and 100 images for testing.

4.2. Training Settings. The input image size of the network is resized into 512×512. The epoch number is 300. And Adam [36] is chosen as the optimizer with a batch-size of 1 and weight decay of 0.00001. Training is started with a learning rate of 0.00005. The cross-entropy loss function is chosen as the loss function in training, and the definition is shown in the following equation

$$\text{loss function} = -\frac{1}{N}\sum_{i}^{N} Y_i \log\left(y_i\right), \tag{1}$$

where N means the category number which is 7, Y_i is the ground truth representing whether the pixel belongs to category i, 1 for yes and 0 for no. And y_i is the prediction probability that the pixel belongs to category i.

4.3. Evaluation Criteria. Four metrics are introduced to evaluate the performance of crack semantic segmentation, Precision (Pr), Recall (Re), F1, and Intersection over Union (IoU). Precision describes the ratio of all pixels predicted to be the crack type that is actually positive. Recall shows the completion of crack prediction, which is a ratio of all crack pixels in the image which is predicted to be crack. F1 is the metric combining precision and recall. IoU calculates a ratio between the number of true positives and the sum of the true positives, false negatives, and false positives. The definition of Pr, Re, F1, and IoU are as follows:

$$\begin{aligned} \text{Precision} &= \frac{TP}{TP + FP}, \\ \text{Recall} &= \frac{TP}{TP + FN}, \\ F1 &= 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \\ IoU &= \frac{\text{GroundTruth} \cap \text{Prediction}}{\text{GroundTruth} \cup \text{Prediction}}, \end{aligned} \tag{2}$$

where TP (True Positive) means the number of crack pixel predicted to be cracks, FP (False Positive) means the number of pavement pixel wrongly predicted to be cracks, FN (False

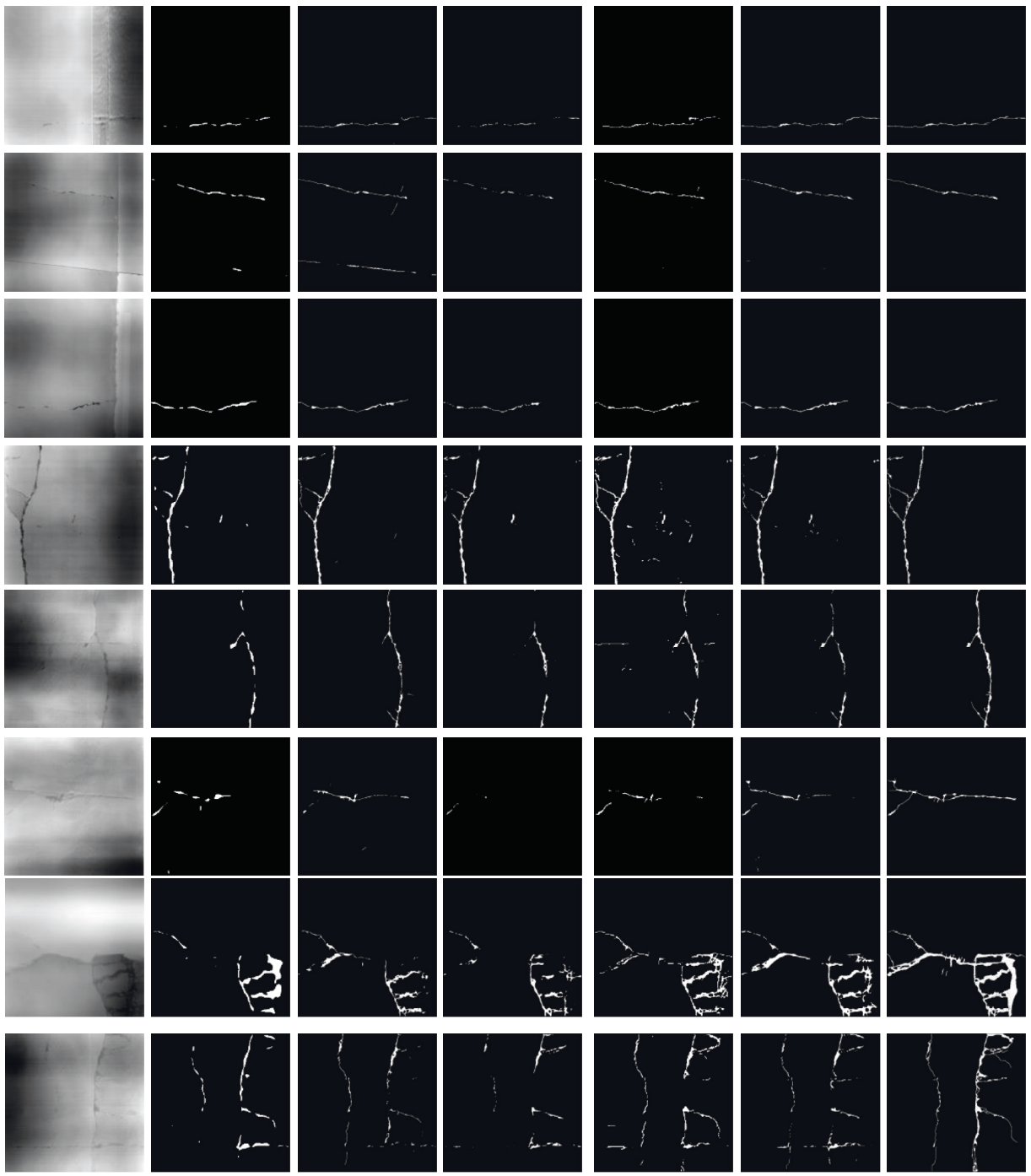


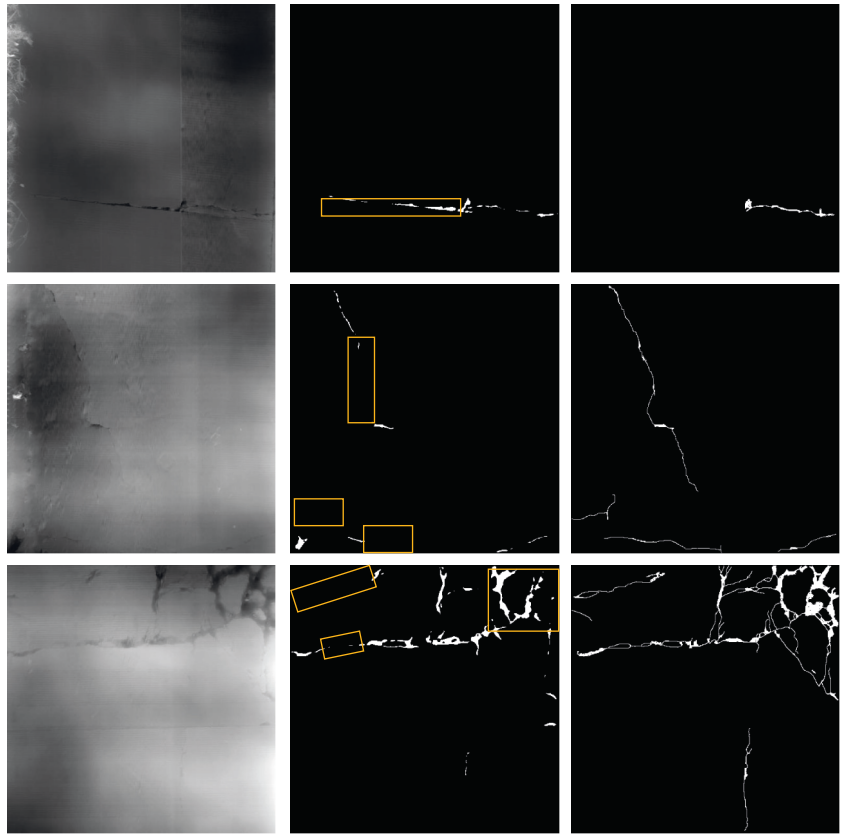
Figure 7: The segmentation results. (a) the original image, (b) the results of AutoEncoder, (c) the results of PSPNet, (d) the results of U-Net, (e) the results of KiU-net, (f) the results of GA-Unet, (g) ground truth of crack in pavement image.

Negative) means the number of crack pixel wrongly predicted to be pavement pixel.

5. Results and Discusses

To evaluate the performance of the proposed GA-Unet, the test dataset selected from the CPC dataset is applied to evaluate the network. The following are the results and discussion of the experiment.

5.1. Learning Process Experiment. Figure 6 shows the result of U-Net and GA-Unet at different epochs. The effort of concrete pavement crack segmentation is improving and the



(a) (b) (c)

Figure 8: Typical errors resulting from PDSNet. (a) pavement images, (b) GA-Unet, and (c) ground truth.

results become closer to the ground truth with the epoch increasing regardless of the U-Net or GA-Unet. However, the GA-Unet is more accurate than U-Net for the same training epoch. The addition of the graph branch can improve the learning ability, enhance feature extraction capability, and boost the convergence speed.

5.2. Comparison Experiment. The comparison experiment between the AutoEncoder, PSPNet [37], U-Net [19], KiUnet [38], and GA-Unet is conducted, and the results are illustrated in Table 1 and Figure 7. AutoEncoder is the simplest segmentation network with only an encoder and decoder structure. U-Net is the segmentation backbone in KiU-Net and GA-Unet. KiU-Net adds an over-complete representation branch based on U-Net to promote the performance. GA-Unet adds the graph network branch to enrich the feature represents. The U-Net can be regarded as the original semantic segmentation network compared to the GA-Unet. The comparison result between U-Nnet and GA-Unet can verify the validity of graph network branch. The performance is represented by four metrics, and the optimal results have been highlighted in bold in Table 1. GA-Unet achieves the optimal results in the metrics of F1, and IoU, which are 0.53 and 0.37, respectively. In addition, GA-Unet has a significant improvement in Recall, F1, and IoU metrics compare to the U-Net, which is increased by 0.12, 0.08 0.06. Although GA-Unet is weaker than U-Net in terms of Precision and KiU-Net in terms of Recall, GA-Unet achieves better performance in segmenting cracks in concrete pavement in general. Figure 7 shows the comparison between the segmentation image of PSPNet, U-Net, and GA-Unet. The quality of the crack segmentation conducted by GA-Unet achieved better results than U-Net under different conditions.

5.3. Discussion. Convolutional computation is a common image processing method widely used in computer vision as a feature extractor for images. However, the convolutional network often uses the convolutional kernel with a small size (usually 3×3), which may lead to the problem of large and long object detection such as crack. The graph represents the relationship between nodes. Transforming the image into a graph can generate the connection between every region of the image. Then, the feature maps processed by the graph branch represent the relationships between regions and describe the characteristic of cracks at large scales, such as whether the cracks span multiple regions. This design is validated in Figure 7. Note the first four images are typical cases of single long crack segmentation, including transverse cracks and longitudinal cracks. The segmentation results by GA-Unet are more continuous than the U-Net model, which means the relationship between the regions extracted by the graph branch can enhance the detailed segmentation of continue cracks.

Moreover, it is impressive that adding a graph branch can improve the robustness of the network. The fifth and last two images show the results in light crack segmentation and the multiple cracks segmentation. The result of GA-Unet is significantly outstanding than the other methods. Although there is a disparity between the results of GA-Unet and the ground truth, the potential of adding graph branches has been validated experimentally.

However, compared to the ground truth, the GA-Unet can still be improved. In the first row of Figure 8, the concrete joints are identified as cracks, due to the similar feature between joint and crack. The joints can be considered as a separate category for detection to reduce the mistake of cracks. In the second row in Figure 8, the pixels of the shallow crack are ignored by GA-Unet method, and the same situation appears in the left crack in the last row. This indicates that the feature extraction branch in the GA-Unet is not effective enough in extracting shallow cracks, and the next step can be considered to enhance the feature of shallow cracks and improve the feature extraction branch. In the last row of Figure 8, the performance of GA-Unet is worse at the junction of shallow and heavy cracks and inside the severely broken area, which may be influenced by the deeper crack, and the accuracy of the surrounding shallow crack is inhibited, so we can consider the post-processing methods to make corrections, for example, using the CRF (Conditional Random Field) method to cluster the surrounding pixel with similar feature to improve the effect.

6. Conclusion

In this work, an end-to-end concrete pavement crack segmentation network called GA-Unet is proposed, for which a graph feature extraction branch is developed. The image can be processed as a graph through the graph generation method. The graph branch extracts the detailed graph features of the concrete pavement crack. The graph feature is fused with the image feature extracted by encoder structure, which is helpful to improve the continuity of crack segmentation. After the feature fusion, the new feature map is applied to the decoder to complete the segmentation.

The crack segmentation methods based on deep learning need sufficient data to support training. Hence, a concrete pavement 3D image dataset has been built. Furthermore, we evaluate our method on the dataset. The results of experiments prove that the graph branch can significantly improve the performance of crack segmentation based on the existing network.

Data Availability

All data and program files included in this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. NSFC71871165) and the Fundamental Research Funds for the Central Universities (Grant no. TTS2021-03).

References

[1] A. V. Cesar, Queiroz and Surhid Gautam, *Road infrastructure and economic development: some diagnostic indicators*, World Bank Publications, Washington, DC, USA, 1992.

[2] L. Pellecuer, G. J. Assaf, and M. St-Jacques, "Life cycle environmental benefits of pavement surface maintenance," *Canadian Journal of Civil Engineering*, vol. 41, no. 8, pp. 695–702, 2014.

[3] G. W. Flintsch and K. McGhee, *Quality management of pavement condition data collection*, Transportation Research Board, Washington, DC, USA, 2009.

[4] A. Cubero-Fernandez, F. J. Rodriguez-Lozano, R. Villatoro, J. Olivares, and J. M. Palomares, "Efficient pavement crack detection and classification," *EURASIP Journal on Image and Video Processing*, vol. 2017, no. 1, pp. 1–11, 2017.

[5] J. Laurent, M. Talbot, and M. Doucet, "Road surface inspection using laser scanners adapted for the high precision 3d measurements of large flat surfaces," in *Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling (Cat. No. 97TB100134)*, pp. 303–310, IEEE, Ottawa, ON, Canada, May 1997.

[6] S. Mathavan, K. Kamal, and M. Rahman, "A review of three-dimensional imaging technologies for pavement distress detection and measurements," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2353–2362, 2015.

[7] K. C. P. Wang, Q. J. Li, G. Yang, Y. Zhan, and Y. Qiu, "Network level pavement evaluation with 1 mm 3d survey system," *Journal of Traffic and Transportation Engineering*, vol. 2, no. 6, pp. 391–398, 2015.

[8] J. Wang and R. X. Gao, "Pavement distress analysis based on dual-tree complex wavelet transform," *International Journal of Pavement Research and Technology*, vol. 5, no. 5, p. 283, 2012.

[9] Y. Zhang and H. Zhou, "Automatic pavement cracks detection and classification using radon transform," *Journal of Information and Computational Science*, vol. 9, no. 17, pp. 5241–5247, 2012.

[10] A. Ayenu-Prah and N. Attoh-Okine, "Evaluating pavement cracks with bidimensional empirical mode decomposition," *EURASIP Journal on Applied Signal Processing*, vol. 1–7, 2008.

[11] S. Sorncharean and S. Phiphobmongkol, "Crack detection on asphalt surface image using enhanced grid cell analysis," in *Proceedings of the 4th IEEE International Symposium on Electronic Design, Test and Applications (delta 2008)*, pp. 49–54, IEEE, Hong Kong, January 2008.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[13] A. Zhang, K. C. P. Wang, R. Ji, and Q. J. Li, "Efficient system of cracking-detection algorithms with 1-mm 3d-surface models and performance measures," *Journal of Computing in Civil Engineering*, vol. 30, no. 6, Article ID 04016020, 2016.

[14] Y. Du, N. Pan, Z. Xu, F. Deng, Y. Shen, and H. Kang, "Pavement distress detection and classification based on yolo network," *International Journal of Pavement Engineering*, vol. 22, no. 13, pp. 1659–1672, 2021.

[15] Z. Liu, Y. Cao, Y. Wang, and W. Wang, "Computer vision-based concrete crack detection using u-net fully convolutional networks," *Automation in Construction*, vol. 104, pp. 129–139, 2019.

[16] Y. Shi, L. Cui, Z. Qi, F. Meng, and Z. Chen, "Automatic road crack detection using random structured forests," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3434–3445, 2016.

[17] R. Stricker, D. Aganian, M. Sesselmann et al., "Road surface segmentation - pixel-perfect distress and object detection for road assessment," in *Proceedings of the International Conference on Automation Science and Engineering (CASE)*, Lyon, France, August 2021.

[18] J. Long, E. Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, Boston, MA, USA, June 2015.

[19] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," *Lecture Notes in Computer Science*, Springer, in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, October 2015.

[20] X. Yang, H. Li, Y. Yu, X. Luo, T. Huang, and X. Yang, "Automatic pixel-level crack detection and measurement using fully convolutional network," *Computer-Aided Civil and Infrastructure Engineering*, vol. 33, no. 12, pp. 1090–1109, 2018.

[21] Z. Qu, W. Chen, S.-Y. Wang, T.-M. Yi, and L. Liu, "A crack detection algorithm for concrete pavement based on attention mechanism and multi-features fusion," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2021.

[22] B. Sanchez-Lengeling, E. Reif, P. Adam, B. Alexander, and Wiltschko, *A Gentle Introduction to Graph Neural Networks*, Distill, 2021, https://distill.pub/2021/gnn-intro.

[23] J. Bruna, W. Zaremba, Arthur Szlam, and Y. LeCun, "Spectral Networks and Locally Connected Networks on Graphs," 2013, https://arxiv.org/pdf/1312.6203.

[24] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, https://arxiv.org/abs/1609.02907.

[25] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035, California, CA, USA, December 2017.

[26] H. Gao, Z. Wang, and S. Ji, "Large-scale learnable graph convolutional networks," in *Proceedings of the 24th ACM SIGKDD International Conference On Knowledge Discovery Amp; Data Mining, KDD '18*, New York, NY, USA, 2018.

[27] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proceedings of the International Conference on Machine Learning*, PMLR, South Korea, July 2016.

[28] G. Huo, Y. Zhang, J. Gao, B. Wang, Y. Hu, and B. Yin, "Caegcn: cross-attention fusion based enhanced graph convolutional network for clustering," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[29] J. Wang, Y. Zhang, Y. Wei, Y. Hu, X. Piao, and B. Yin, "Metro passenger flow prediction via dynamic hypergraph convolution networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 12, pp. 7891–7903, 2021.

[30] J. Laurent, D. Lefebvre, and E. Samson, "Development of a new 3d transverse laser profiling system for the automatic measurement of road cracks," in *Proceedings of the Symposium on Pavement Surface Characteristics, 6th*, Portoroz, Slovenia, October 2008.

[31] J. Laurent, J. F. Hébert, D. Lefebvre, and Y. Savard, "Using 3d laser profiling sensors for the automated measurement of road surface conditions," in *Proceedings of the 7th RILEM International Conference on Cracking in Pavements*, pp. 157–167, Springer, Netherlands, June 2012.

[32] R. Kaushik, J. Xiao, W. Morris, and Z. Zhu, "3d laser scan registration of dual-robot system using vision," in *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4148–4153, IEEE, St. Louis, MO, USA, October 2009.

[33] A. Cohen, C. Zach, S. N. Sinha, and M. Pollefeys, "Discovering and exploiting 3d symmetries in structure from motion," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1514–1521, IEEE, Rhode Island, USA, June 2012.

[34] A. Zhang, K. C. P. Wang, B. Li et al., "Automated pixel-level pavement crack detection on 3d asphalt surfaces using a deep-learning network," *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, no. 10, pp. 805–819, 2017.

[35] H. Lang, J. J. Lu, Y. Lou, and S. Chen, "Pavement cracking detection and classification based on 3d image using multi-scale clustering model," *Journal of Computing in Civil Engineering*, vol. 34, no. 5, Article ID 04020034, 2020.

[36] P. Diederik, "Kingma and Jimmy Ba. Adam: a method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015*, San Diego, CA, USA, May 2015.

[37] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2881–2890, Honolulu, HI, USA, July 2017.

[38] Jeya Maria Jose Valanarasu, V. A. Sindagi, I. Hacihaliloglu, and V. M. Patel, "Kiu-net: towards accurate segmentation of biomedical images using over-complete representations," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 363–373, Springer, Lima, Peru, October 2020.

Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 5033601, 8 pages
https://doi.org/10.1155/2022/5033601

Research Article

Research on Direct Braking Force Estimation and Control Strategy Using Tire Inverse Model

Zhiguo Zhou[1] **and Xiaoning Zhu**[2]

[1]*Zhejiang Institute of Communications, Hangzhou 311112, China*
[2]*Xi'an Aeronautics Computing Technique Research Institute, Aviation Industry Corporation of China, Xi'an 710065, China*

Correspondence should be addressed to Zhiguo Zhou; zhouzzg1107@126.com

Received 13 March 2022; Revised 4 May 2022; Accepted 13 May 2022; Published 3 June 2022

Academic Editor: Yanming Shen

With the rapid development of computer control and vehicle intelligence technology, speed and safety of vehicles have been greatly improved, and the requirements for vehicle control performance are getting higher and higher. For the direct braking force control, in the process of deceleration, a fast braking response can be obtained, which improves the braking performance and vehicle safety. This paper concentrates on direct braking force estimation and control strategy using a tire inverse model based on the antilock braking system, and to solve the problem of the existing ABS system is mainly antilock braking function, no direct braking force control function. Taking magic formula model for reference inverse model, the critical parameters under different road surfaces are obtained according to experience data. Then, the desired slip ratio corresponding to braking force can be obtained via fast tire inverse model look-up table method. The tyre friction self-adjustment decision making is obtained using the tire inverse model method. A direct braking force antilock braking system (DBF-ABS) controller is built using the nonsingular fast terminal sliding mode method. The simulation results indicated that the control strategy has adaptability and stability to the change of road conditions.

1. Introduction

On behalf of adapting to the complex working conditions and enhancing the vehicle's safety and comfort, various types of automotive active electronic control systems are presented. The research on integrated vehicle dynamics control already became an urgent problem to be solved and has attracted extensive attention [1–3].

These research studies have improved vehicle performance to a certain extent but still have some problem to be solved. Some studies focus only on the design of the main circuit [4–6]. The calculated stable side forces and total yaw moments are applied without considering targeted production and allocation manners. The influence of tire dynamics is essentially treated as nominal parameters, such as the basic angular stiffness when the problem formulates. But there is an interaction between the nonlinearity of tire characteristics and vehicle dynamics [7–10]. However, these studies based on the main loop design can provide the maximum performance margins and theoretic insight, and the vehicle motion force generation process does not fully take the special interactions between tires and the road into account. It can lead to insufficient control accuracy or overly optimistic performance results. When need more tire force, for example, if the tires have been in big slip rate, applying large braking force will only make things worse. More importantly, the realization of tire forces is still a critical problem in relation to handling property [11–14].

This paper concentrates on direct braking force estimation and control strategy using tire inverse model based on the ABS. The main content of the rest section of the paper is as follows. Section 2 discusses the development and new features of vehicle integrated control and direct torque ABS control technology in recent years. Section 3 describes tyre friction self-adjustment decision-making

method and direct braking force DBF-ABS controller design. The results of simulation analysis are presented in Section 4. The conclusions and future related work is provided in Section 5.

2. Literature Review

ABS system has been developed since the early 20th century [15, 16]. At the end of 1970s, the great progress of digital electronic technology and large-scale integrated circuit laid the technical foundation of ABS. After the mid-1980s, the development of ABS paid more attention to its own cost performance ratio [17, 18]. The work during this period has increased the popularity of ABS. The ABS system is considered as the most important safety technical achievement since the adoption of safety belt in automobiles [19, 20].

With the improvement of vehicle speed and intelligent technology level, the related vehicle control technology based on ABS has also achieved new and rapid development. The EBD and ABS are integrated to form the automobile auxiliary integrated system using CAN bus [21]. Brake-by-wire control systems for intelligent vehicles are studied [22]. In the past decades, quite a few advanced intelligence, automatic control, and computer technologies have been widely used in ABS for smart vehicles, for example, distributed and self-adaptive vehicle speed estimation and control [23]; optimal slip rate is obtained and tracked based on the multiphase method [24, 25]. Besides, a nonlinear predictive control strategy was proposed [26, 27].

Especially with the improvement of vehicle integration, collaborative or optimal control has become a new research hotspot, such as, combined emergency braking, integrated vehicle chassis control [28–30], and self-learning adaptive control [31, 32]. In addition, with the development of modern computer and communication technology, some new technologies are applied to reduce traffic congestion and vehicle driving safety, such as advanced driving assistance system and autonomous driving [33–35]. In particular, the development of intelligent networked vehicle technology has further improved vehicle safety [36, 37]. Based on these previous studies, this paper concentrates on direct braking force estimation and control strategy using the tire inverse model based on the ABS to solve the direct braking force control problem.

3. Method

In this section, the control structure of the direct braking force self-adjustment decision making and control is designed. The desired direct braking force friction is estimated and tracked. Based on the estimated values of tyre friction, the desired slip ratio can be obtained, which is corresponded with the specific desired tyre friction, using the reverse look-up table method. Then, based on the tire braking force model, the terminal sliding mode method ensures that the antilock braking system can achieve the desired slip rate to obtain direct braking.

3.1. Control Structure. The direct braking force estimation, self-adjusting decision, and control structure are shown in Figure 1. The details are as follows:

Step 1: the driver commands are received from the brake pedal system. The control inputs, namely, wheel control moments, are obtained by a servo loop to distribute force and torque to the four tire-road contact blocks.

Step 2: the direct braking force is estimated using direct braking force quick look-up table based on tire inverse model. The ideal tyre-road friction is obtained by direct braking force decision-making system.

Step 3: the control target error is calculated and direct braking target control force is obtained and assigned by direct braking force control system.

At last, simulations and results are analyzed based on the vehicle dynamic model, including tyre-road dynamic model, vehicle dynamic model, and braking force sensor model.

3.2. Vehicle Model. Tire-road friction has obvious nonlinear characteristics, which should be estimated. An attempt has been made to measure braking torque using force sensors mounted on caliper mounts. Assuming that braking torque can be obtained from sensors, then, tyre-road friction can be calculated [24].

In Figure 2, vehicle dynamics and brake model are built as follows:

$$\begin{aligned} \dot{u} &= -\frac{\sum_{i=FL,FR,RL,RR} F_{fb,i}}{M}, \\ \dot{\omega}_i &= \frac{R_{b,i}F_{xb,i} - T_{b,i}}{J_{b,i}}, \\ F_{xb,i} &= \mu_i \cdot F_{Z,i}, \\ F_{Z,i} &= \frac{1}{4}\mathrm{Mg}, \end{aligned} \tag{1}$$

where M is the mass of vehicle, $F_{xb,i}$ is the friction force, ω_i is the angular speed, $J_{b,i}$ is the wheel inertia, $R_{b,i}$ is the radius of the vehicle wheel, $F_{fb,i}$ is the brake force measured by force transducer, $T_{b,i}$ is the brake torque, g is the acceleration of gravity, $F_{Z,i}$ is the vertical load of the wheel, and i is the front, rear, left, and right positions wheel.

3.3. Tyre Model. The tire model is derived from Magic formula. Magic formula is the universal semiempirical tire model [38]. The general form is as follows:

$$\mu_l = A_l \sin\left[B_l \arctan\left\{C_l\lambda - D_l\left(C_l\lambda - \arctan\left(C_l\lambda\right)\right)\right\}\right], \tag{2}$$

where μ_l is the longitudinal friction coefficient, C_l is the stiffness factor of the tire, λ is the longitudinal slip of the

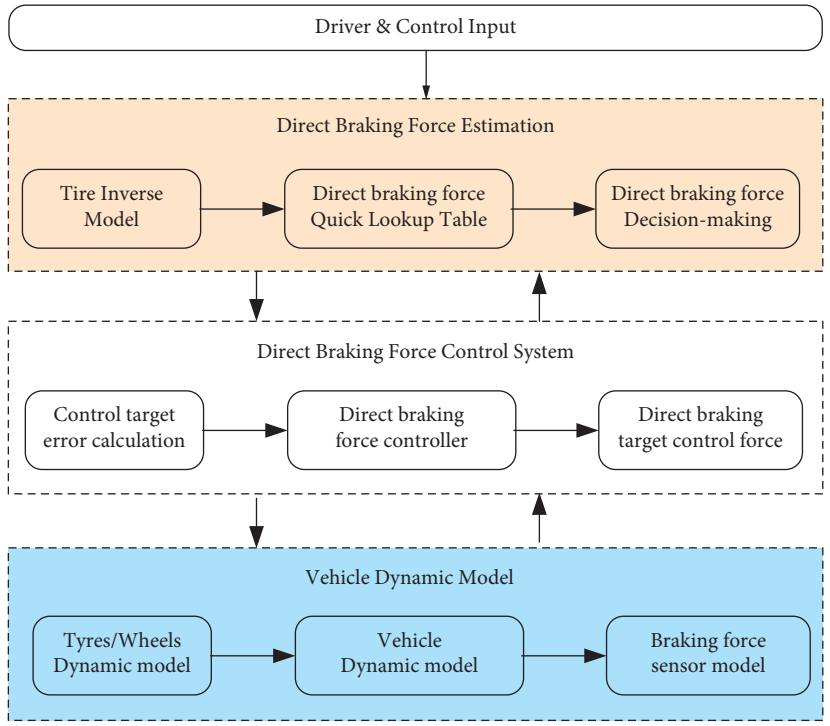


Figure 1: Schematic diagram of direct braking force estimation and control configuration.

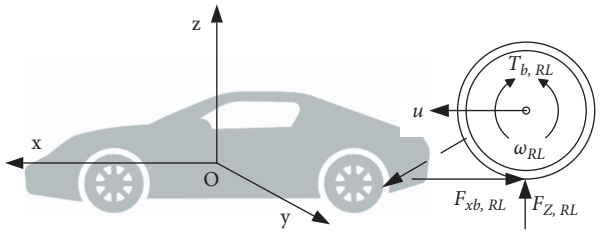


Figure 2: Vehicle braking dynamics model.

vehicle, A_l is the peak value, B_l is the shape factor, D_l is the curvature factor.

3.4. Direct Braking Force System Controller Design. In this section, a direct braking force controller based on ABS with terminal sliding mode control method is proposed, as shown in Figure 3.

After introducing the controller, the parameter uncertainty and the influence of external interference can be eliminated.

The following equation is the sliding surface designed in this paper:

$$S_{\mathrm{DBE-ABS}} = \frac{\mathrm{d}\left(F_{\mathrm{Dir\_Brak}} - F_{\mathrm{Ref\_Brak}}\right)}{\mathrm{d}t + \xi\left(F_{\mathrm{Dir\_Brak}} - F_{\mathrm{Ref\_Brak}}\right) + \zeta\left(F_{\mathrm{Dir\_Brak}} - F_{\mathrm{Ref\_Brak}}\right)^{(a/b)}}, \tag{3}$$

where $e \in R$; ξ, ζ are constants, and $\xi > 0$, $\zeta > 0$; a, b are positive odd integers. At the same time, $a < b < 2a$.

The dynamic adjustment process of sliding mode control consists of arrival stage and sliding control two stages. To make the switch manifold reachable, smooth, and fast convergence in a finite time, a "terminal attractor" is proposed to improve chatter less control while taking full advantage of nonsingular fast terminal sliding mode control. This controller sliding surface design as follows:

$$\dot{S}_{\mathrm{DBE-ABS}} = \left(-\varsigma s - \vartheta s^{(g/f)}\right)\left(F_{\mathrm{Dir\_Brak}} - F_{\mathrm{Ref\_Brak}}\right)^{(p-q/q)}, \tag{4}$$

where $\varsigma \in R^+$; $\vartheta \in R^+$; $m > 0$ are odd integers. $n > 0$ is odd integers. At the same time, $0 < g/f < 1$. And the direct braking force antilock braking control law is shown below:

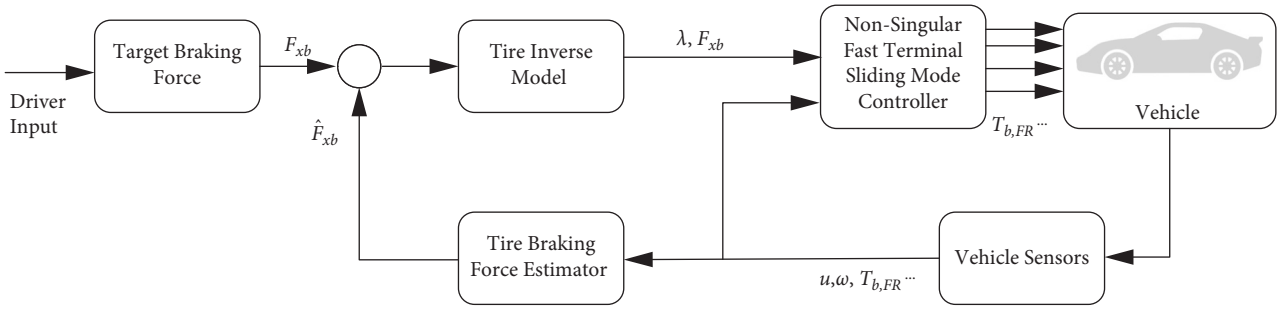


Figure 3: Schematic of direct braking force antilock braking system controller.

$$F_{\text{Con\_Brak}} = \frac{u \cdot J}{r_b^2} \frac{q}{\zeta \cdot p} \left(-\ddot{F}_{\text{Err\_Brak}} \cdot F_{\text{Err\_Brak}}^{(q-p/q)} - \xi \cdot \dot{F}_{\text{Err\_Brak}} \cdot F_{\text{Err\_Brak}}^{(q-p/q)} + \left(\varsigma s + \vartheta s^{(g/f)}\right)\right) - \frac{u}{r_b^2 \cdot J} \left(\frac{r_b^2 F_{\text{Dir\_Brak}} + J(1-\lambda)(\mathrm{d}u/\mathrm{d}t)}{u} - \dot{F}_{\text{Inv\_Ref\_Brak}}\right). \tag{5}$$

In the above equation, $F_{\text{Err\_Brak}} = F_{\text{Dir\_Brak}} - F_{\text{Ref\_Brak}}$, since $0 < q - p/q$, the system eliminates the singularity problem and can converge to system equilibrium by tracking the sliding surface.

4. Simulations and Analysis

On behalf of proving the effectiveness of the direct braking friction self-adjusting decision and controller, simulation is carried out in this section. Firstly, the characteristics of tyre friction are presented by using three sets of different test points, analyzing the impact of these test points on vehicle speed control and vehicle braking distance. Then, the superiority of the nonsingular fast terminal sliding mode method-based DBF-ABS controller is compared with fast sliding mode control and Bang-Bang-based ones. Finally, the overall performance of friction self-tuning control in μ-split condition is achieved.

4.1. Parameter Set. Parameters required to build the simulation and analysis system are listed in Table 1.

The different road surface Magic formula empirical parameters can be obtained in Table 2. Based on the above information, A_L, B_L, C_L, and D_L can be constrained in the range of corresponding different roads [24].

4.2. Simulations of Quasilinear Braking Area Characteristics. The braking force between the tire and the ground has a characteristic that are transitioned from linear to nonlinear, including two areas quasilinear braking area and emergency braking nonlinear area, as shown in equation (2). In order to better estimate and control the direct braking force, it is necessary to analyze the interaction characteristics of these two regions. In the tyre friction self-adjustment decision making, control sets in quasilinear braking area points $\lambda \in [0.01 \sim 0.10]$ have been are selected 3 points, that is A, B, and C. And based on these points, the simulations are conducted. The results can be obtained in Figures 4 and 5. The control points of $(\widehat{\lambda}_{xb}, \widehat{F}_{xb})$ are within the area that $\lambda \in [0.01 \sim 0.10]$. And as is shown in Figure 5, little change is present in control slip rate, so there is a little effect on braking distance.

4.3. Simulations of Emergency Braking Area Characteristics. The control set points in emergency braking area, in $\lambda \in [0.10 \sim 0.20]$, have already selected 3 points, that is A, B, and C. Based on these points, the simulations are conducted. The results can be obtained in Figures 6 7.

The control points of $(\widehat{\lambda}_{xb}, \widehat{F}_{xb})$ are within the area that $\lambda \in [0.10 \sim 0.20]$. From Figures 6 and 7, although the step of the slip rate is the same as that of case 4.2, it has a greater impact on the braking distance. The result is significantly improved compared with Case 4.3, because the identification point information is more applicable to the nonlinear variation of tire-road friction. In conclusion, the sampling point λ contributes to more impact on the braking distance.

4.4. Simulations under μ-Split Condition of Different Road Surface. The braking force varies with different road environment. The scenarios of the vehicle running under μ-split condition of different road surfaces are chosen to verify the self-tuning and adaptive performances of proposed estimator and controller, as shown in Figure 8.

Set the constant reference friction force *sss* in this μ-split simulation. The road conditions change between the asphalt, dry road and the asphalt, wet road at 1.5s, as shown in Figures 9 and 10. In figures, the friction and vehicle acceleration have not changed drastically. As shown in Figure 9, although the friction force is same, the figure displays that the reference value at 1.0 seconds decreases from about 0.054 to 0.036 as the road conditions change. As shown in Figure 10, although the friction force is same, the vehicle

Table 1: Parameters used in simulations.

Notation	g acceleration of gravity	J_b wheel inertia	R_b wheel radius	M vehicle mass
Unit	m/s^2	$kg \cdot m^2$	m	kg
Value	9.8	12	0.25	1530

Table 2: Magic formula parameters of different roads.

Road	Snow	Cobblestone (wet)	Asphalt-wet	Cobblestone (dry)	Concrete (dry)	Asphalt (dry)
A_L	0.20	0.40	0.80	0.85	0.37	1.10
B_L	1.45	1.45	1.60	1.40	1.64	1.55
C_L	17.43	14.02	15.63	10.09	13.42	13.42
D_L	0.65	0.60	0.45	0.64	0.53	0.53

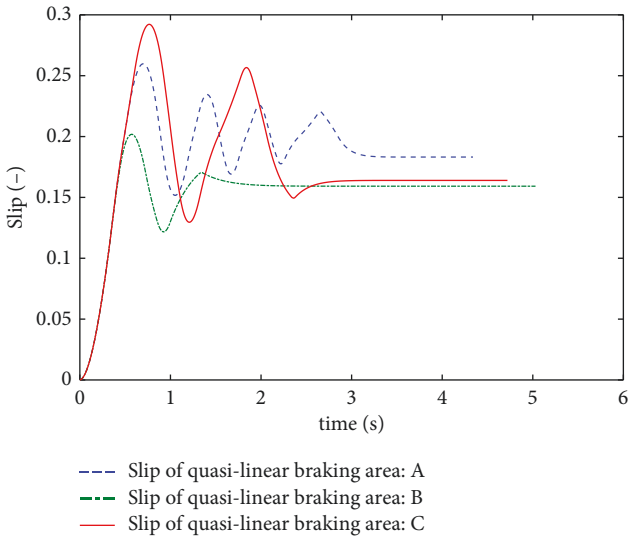


Figure 4: μ in quasilinear area.

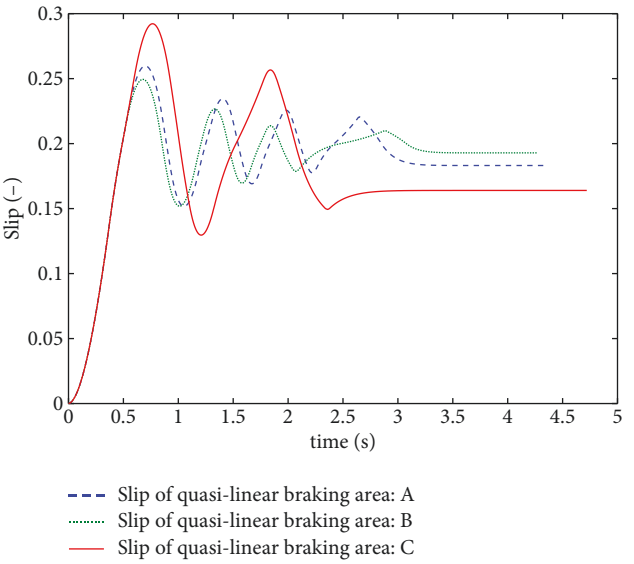


Figure 6: μ in emergency area.

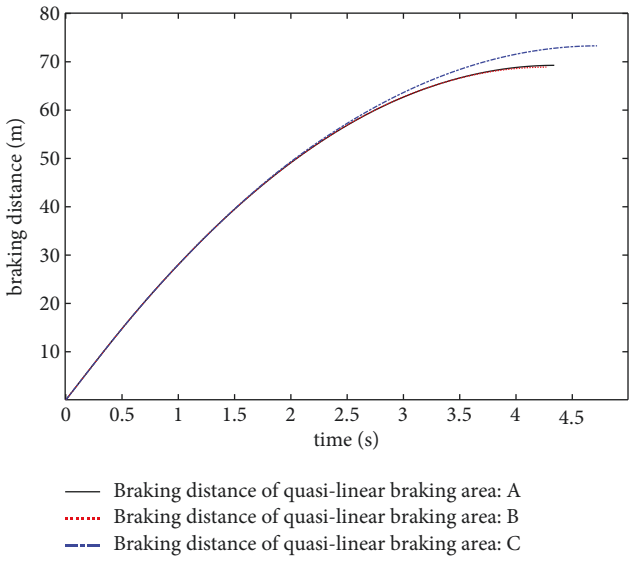


Figure 5: Braking distance in quasilinear area.

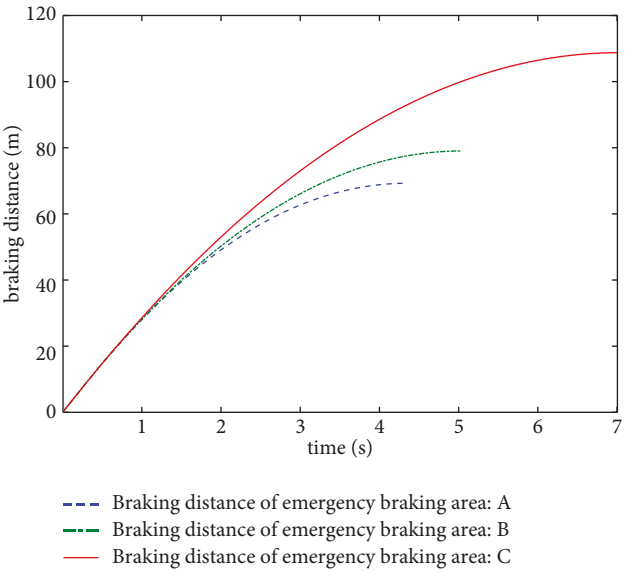


Figure 7: Braking distance in emergency area.

Figure 8: Scene of direct braking friction control under μ-split condition of different road surfaces.

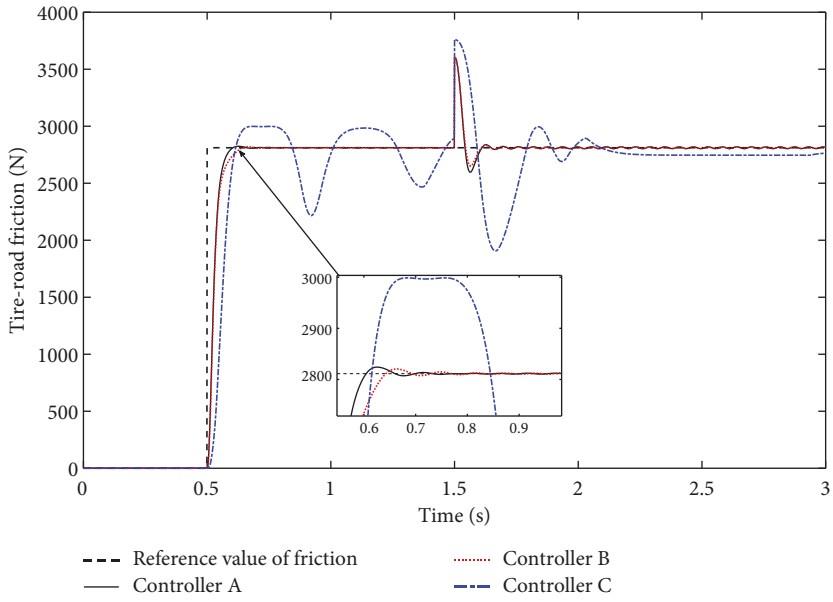


Figure 9: Reference and real value of friction.

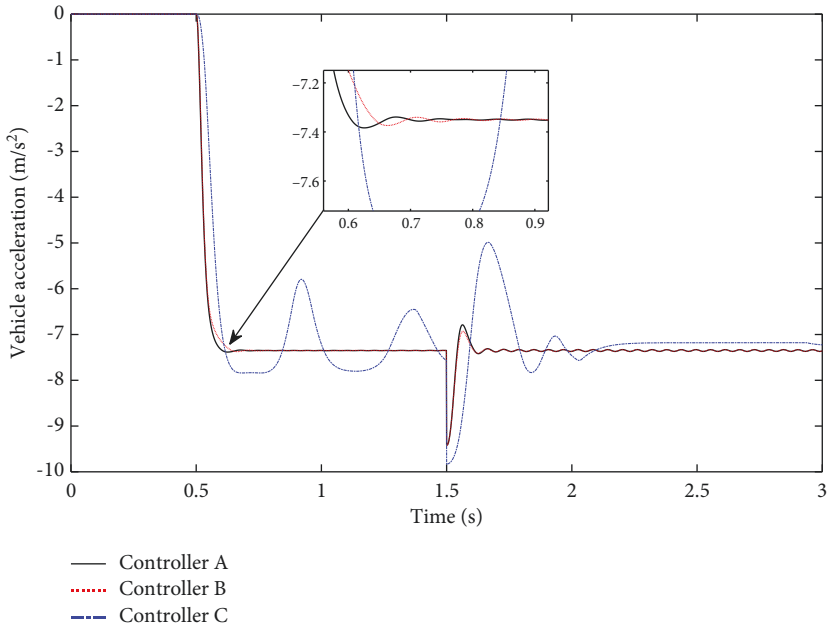


Figure 10: Vehicle acceleration.

maintains the same acceleration on different road surfaces. However, the figure displays that the acceleration value at 1.5 seconds decreases as the road conditions change. The results shows that the proposed self-tuning controller can estimate and keep the tracking value consistent with the reference value under different road surfaces.

Furthermore, the proposed direct braking force controller (controller A) is compared with fast terminal sliding

mode controller (controller B) and Bang-Bang controller (controller C). From Figure 9, the controller A converges between reference and real value of friction faster than controllers B and C. In Figure 10, the results are similar for vehicle acceleration control. From the results, although the performance of the Bang-Bang controller is not as good as the other two, it is often used in engineering due to its simple structure and low requirements for the control processing unit. The figure shows that the proposed DBF-ABS controller can keep the tracking value consistent with better performance.

5. Conclusions and Future Work

Based on the existing ABS, a kind of direct braking force estimation and control strategy based on DBF-ABS control system was proposed to solve the problem of the existing ABS system which is mainly antilock braking function, no direct braking force control function. For the direct braking force control, in the process of deceleration, a fast braking response can be obtained, which improves the braking performance and vehicle safety. Firstly, taking magic formula model for reference inverse model, the critical parameters under different road surface are obtained according to experience data. Then, the desired slip ratio corresponding to braking force can be obtained via tire inverse model look-up table method. The tyre friction self-adjustment decision making is obtained using the tire inverse model method. A direct braking force antilock braking system (DBF-ABS) controller is built using the nonsingular fast terminal sliding mode algorithm. Finally, the simulations and analysis results show that the control method has adaptability and stability under different driving conditions.

Due to limited sensing equipment for direct braking force control data acquisition, future work will be focused on advanced sensing and data estimation. In addition, a wider range of dynamic adaptive direct braking torque control and matching with AEBS is also an interesting topic.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This work was partially supported by the Public Projects of Zhejiang Province (No. LGG19E050001), and the project named as research on key technologies of monitoring and early warning of operational blind area for trucks. And this work was partially supported by the Transport Vehicle Safety Technology Key Laboratory of Transportation Industry Opening Project, and the project named as research on blind area monitoring method of Car train.

References

[1] P. Song, M. Tomizuka, and C. Zong, "A novel integrated chassis controller for full drive-by-wire vehicles," *Vehicle System Dynamics*, vol. 53, no. 2, pp. 215–236, Feb 1 2015.

[2] K. Gao, Y. Yang, and X. Qu, "Diverging effects of subjective prospect values of uncertain time and money," *Communications in Transportation Research*, vol. 1, pp. 100007–102021, 2021.

[3] R. Zhang, K. Li, F. Yu, and Z. He, "Novel electronic braking system design for evs based on constrained nonlinear hierarchical control," *International Journal of Automotive Technology*, vol. 18, no. 4, pp. 707–718, 2017.

[4] M. Nagai, M. Shino, and F. Gao, "Study on integrated control of active front steer angle and direct yaw moment," *JSAE Review*, vol. 23, no. 3, pp. 309–315, 2002.

[5] C. Li, Y. Xie, G. Wang, X. Zeng, and H. Jing, "Lateral stability regulation of intelligent electric vehicle based on model predictive control," *Journal of Intelligent and Connected Vehicles*, vol. 4, no. 3, pp. 104–114, 2021.

[6] Y. Cai, T. Luan, H. Gao, and H. Wang, "Yolov4-5D: an effective and efficient object detector for autonomous driving," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, Article ID 4503613, 2021.

[7] D. Li and F. Yu, *A Novel Integrated Vehicle Chassis Controller Coordinating Direct Yaw Moment Control and Active Steering*, SAE, Thousand Oaks, CA, USA, 2007.

[8] X. Shen and F. Yu, "Study on vehicle chassis control integration based on a main-loop-inner-loop design approach," *Proceedings of the Institution of Mechanical Engineers - Part D: Journal of Automobile Engineering*, J. Automobile Engineering, vol. 220, no. 11, pp. 1491–1502, 2006.

[9] T. J. Gordon, "A flexible hierarchical model-based control methodology for vehicle active safety systems," *Vehicle System Dynamics*, vol. 46, no. 1, pp. 63–75, 2008.

[10] M. B. Alberding, "Nonlinear Hierarchical Control Allocation for Vehicle Yaw Stabilization and Rollover Prevention," in *Proceedings of the 2009 European Control Conference (ECC)*, Diploma thesis, Budapest, Hungary, August 2008.

[11] J. Deur, D. Pavković, G. Burgio, and D. Hrovat, "A model-based traction control strategy non-reliant on wheel slip information," *Vehicle System Dynamics*, vol. 49, no. 8, pp. 1245–1265, 2011.

[12] L. Alvarez, J. Yi, R. Horowitz, and L. Olmos, "Dynamic friction model-based tire-road friction estimation and emergency braking control," *Journal of Dynamic Systems, Measurement, and Control*, vol. 127, no. 1, pp. 22–32, 2005.

[13] T. Shim and D. Margolis, "Model-based road friction estimation," *Vehicle System Dynamics*, vol. 41, no. 4, pp. 249–276, 2004.

[14] D. Pavkovi´c, J. Deur, G. Burgio, and D. Hrovat, "Estimation of tyre static curve gradient and related model-based traction control application," in *Proceedings of the 2009 IEEE Multiconference on Systems and Control*, pp. 594–599, St. Petersburg,Russia, July 2009.

[15] Y. Chamaillard, G. L. Gissinger, J. M. Perronne, and M. Renner, "An Original Braking Controller with Torque Sensor," in *Proceedings of the Third IEEE Conference on Control Applications*, pp. 619–625, Glasgow, UK, August 1994.

[16] G. L. Gissinger, C. Menard, and A. Constans, "A mechatronic conception of a new intelligent braking system," *Control Engineering Practice*, vol. 11, no. 2, pp. 163–170, 2003.

[17] T. Ishige, H. Furusho, Y. Aoki, and K. Kawagoe, *Adaptive Slip Control Using a Brake Torque Sensor*, AVEC, 2008.

[18] M. Gerard and M. Verhaegen, *Global and Local Chassis Control Based on Load Sensing*, in *Proceedings of the 2009 American Control Conference Hyatt Regency Riverfront*, pp. 677–682, St.Louis,MO,USA, June 2009.

[19] M. Gobbi, J. C. Botero, and G. Mastinu, "Improving the active safety of road vehicles by sensing forces and moments at the wheels," *Vehicle System Dynamics*, vol. 46, no. 1, pp. 957–968, 2008.

[20] M. Brusarosco, A. Cigada, and S. Manzoni, "Experimental investigation of tyre dynamics by means of MEMS accelerometers fixed on the liner," *Vehicle System Dynamics*, vol. 46, no. 11, pp. 1013–1028, 2008.

[21] Y. Qiu, J. Fang, and Z. Zhu, "ABS/EBD automobile auxiliary brake system based on CAN bus," in *Proceedings of the 2021 7th International Symposium on System and Software Reliability (ISSSR)*, Chongqing, China, September 2021.

[22] Z. Xue, C. Li, X. Wang, and Z. Zhong, "Coordinated control of steer-by," *IET Intelligent Transport Systems*, vol. 14, no. 14, pp. 2122–2132, 2020.

[23] Z.-G. Zhao, L. J. Zhou, J. T. Zhang, Q. Zhu, and J.-K. Hedrick, "Distributed and self-adaptive vehicle speed estimation in the composite braking case for four-wheel drive hybrid electric car," *Vehicle System Dynamics*, vol. 55, no. 5, pp. 750–773, 2017.

[24] R. H. Zhang, Z. C. He, H. W. Wang, F. You, and K. N. Li, "Study on Self-Tuning Tyre Friction Control for Developing Main-Servo Loop Integrated Chassis Control System," *IEEE Access*, vol. 5, pp. 6649–6660, 2017.

[25] C. Du, F. Li, C. Yang, Y. Shi, L. Liao, and W. Gui, "Multiphase-based optimal slip ratio tracking control of aircraft antiskid braking system via second-order sliding mode approach," *IEEE*, vol. 27, no. 2, pp. 823–833, 2021.

[26] G. Morrison and D. Cebon, "Combined emergency braking and turning of articulated heavy vehicles," *Vehicle System Dynamics*, vol. 55, no. 5, pp. 725–749, 2017.

[27] Q. Wang, L. Liu, and W. Chen, "Integrated control of automotive electrical power steering system and suspension system based on random sub-optimal control," *China Mechanical Engineering*, vol. 16, no. 8, pp. 743–747, 2005.

[28] D. Li and S. Du, "Integrated vehicle chassis control based on direct yaw moment, active steering and active stabiliser," *Vehicle System Dynamics*, vol. 46, no. 1, pp. 341–351, 2008.

[29] J. Zhang, W. Sun, and H. Du, "Integrated motion control scheme for four-wheel-independent vehicles considering critical conditions," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7488–7497, 2019.

[30] R. Zhang, K. Li, Z. He, and H. Wang, "Advanced Emergency Braking Control Based on a Nonlinear Model Predictive Algorithm for Intelligent Vehicles," *Applied sciences*, vol. 7, no. 5, p. 504, 2017.

[31] S. Rajendran, S. Spurgeon, G. Tsampardoukas, and R. Hampson, "Self-learning Adaptive Integrated Control of an Electric Vehicle in Emergency Braking," in *Proceedings of the 2021 European Control Conference (ECC)*, Delft, Netherlands, July 2021.

[32] S. Chen, X. Zhang, and J. Wang, "Sliding mode control of vehicle equipped with brake-by-wire system considering braking comfort," *Shock and Vibration*, vol. 2020, pp. 1–13, Article ID 5602917, 2020.

[33] Y. Xu, Z. Ye, and C. Wang, "Modeling commercial vehicle drivers' acceptance of advanced driving assistance system (ADAS)," *Journal of Intelligent and Connected Vehicles*, vol. 4, no. 3, pp. 125–135, 2021.

[34] M. Gressai, B. Varga, T. Tettamanti, and I. Varga, "Investigating the impacts of urban speed limit reduction through microscopic traffic simulation," *Communications in Transportation Research*, vol. 1, Article ID 100018, 2021.

[35] Z. Yang, J. Huang, D. Yang, and Z. Zhong, "Design and optimization of robust path tracking control for autonomous vehicles with fuzzy uncertainty," *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 6, pp. 1788–1800, 2022.

[36] J. Larsson, M. F. Keskin, B. Peng, B. Kulcsár, and H. Wymeersch, "Pro-social control of connected automated vehicles in mixed-autonomy multi-lane highway traffic," *Communications in Transportation Research*, vol. 1, Article ID 100019, 2021.

[37] H. Li, J. Zhang, Z. Zhang, and Z. Huang, "Active lane management for intelligent connected vehicles in weaving areas of urban expressway," *Journal of Intelligent and Connected Vehicles*, vol. 4, no. 2, pp. 52–67, 2021.

[38] H. Pacejka, *Tire and Vehicle Dynamics*, Elsevier, Amsterdam, Netherlands, 2006.

Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 1213221, 16 pages
https://doi.org/10.1155/2022/1213221

Research Article

JSTC: Travel Time Prediction with a Joint Spatial-Temporal Correlation Mechanism

Alfateh M. Tag Elsir, Alkilane Khaled, Pengfei Wang, and Yanming Shen

School of Computer Science and Technology, Dalian University of Technology, Dalian, China

Correspondence should be addressed to Pengfei Wang; wangpf@dlut.edu.cn

Received 13 February 2022; Revised 27 March 2022; Accepted 20 April 2022; Published 23 May 2022

Academic Editor: Lijun Sun

Accurate travel time prediction is one of the most promising intelligent transportation system (ITS) services, which can greatly support route planning, ride-sharing, navigation applications, and effective traffic management. Several factors, like spatial, temporal, and external, have big effects on traffic patterns, and therefore, it is important to develop a mechanism that can jointly capture correlations of these components. However, spatial sparsity issues make travel time prediction very challenging, especially when dealing with the origin-destination (OD) method, since the trajectory data may not be available. In this paper, we introduce a unified deep learning-based framework named joint spatial-temporal correlation (JSTC) mechanism to improve the accuracy of OD travel time prediction. First, we design a spatiotemporal correlation block that combines two modules: self-convolutional attention integrated with a temporal convolutional network (TCN) to capture the spatial correlations along with the temporal dependencies. Then, we enhance our model performance through adopting a multi-head attention module to learn the attentional weights of the spatial, temporal, and external features based on their contributions to the output and speed up the training process. Extensive experiments on three large-scale real-world traffic datasets (NYC, Chengdu, and Xi'an) show the efficiency of our model and its superiority compared to other methods.

1. Introduction

Travel time forecasting (TTF) has been considered as one of the most essential services in intelligent transportation systems (ITSs), which greatly supports route planning, ride-sharing, navigation applications, and effective traffic management. TTF is widely used throughout location-based applications and has become one of the most important services in these applications. However, producing an accurate TTF is still challenging since understanding the effects of different dynamic factors (such as urban flows, jams, peak hours, and special situations like public holidays, events, and vacations) on the travel time is a complex task [1]. The dynamic factors can be categorized into four groups as follows:

(1) Spatial dependencies: travel time is greatly affected by the traffic conditions of each region and its neighbors as well, so trips from areas with heavy traffic will take a longer time than others.

(2) Temporal dependencies: traffic conditions during different periods of the day affect the time of travel. For example, road traffic congestion in downtown cities is more severe during the morning and evening peak hours.

(3) Periodical dependencies: periodic patterns such as working hours, weekends, and public events can also affect travel time, where traffic is more congested during workdays and peak times, for example.

(4) External factors: several external factors have also a big impact on the travel time fluctuations, such as weather, holidays, and public events.

Due to the complexity of the spatiotemporal correlations, TTF is a very challenging problem, so accurately

predicting travel time has become a vital task recently [2, 3]. In general, the TTF has been treated as one of two methods (route-based and OD-based) using statistical methods, classical machine learning, and deep learning approaches. First, for route-based approaches, GPS and time series datasets of trajectories are useful in estimating travel times for both road segments and the entire path. However, some complex issues in this technique lead to inaccurate results and costly computations, such as sparsity in trajectory data and GPS devices' errors. Second, the OD-based approach is completely based on the shortest path between the origin and destination points, which reduces the heavy computations and minimizes accumulated error rates of GPS devices. Therefore, the aim of this work is to provide a solution that improves the forecasting accuracy of the OD-based travel time. Many methods have been proposed for TTF, including linear regression (LR) [4], time-varying [5], Kalman filtering (KF) [6, 7], autoregressive integrated moving average (ARIMA) [8], seasonal ARIMA (SARIMA-KF) [9, 10], and random forest (RF) with gradient boosting (GB) (RF-GB) [11]. However, the major disadvantage of these approaches is that they are inappropriate for capturing the relationships between the complicated traffic factors. Most recent researchers have proposed deep learning models that strive to enhance TTF results, such as backpropagation neural networks (BP-NNs) [12–14], long short-term memory (LSTM) [15, 16], convolutional neural networks (CNNs) combined with LSTM (CNN-LSTM) [17], and attention mechanism [18].

Unfortunately, these approaches still suffer from some difficulties, e.g., time-consuming and low speed during the training process, so these methods cannot perform concurrent processing. The sparsity of traffic data represents another concern of TTF approaches, where the historical traffic data do not cover the entire region. On the other hand, the correlations between the spatial features have been considered in many existing works, but most of these methods only focused on the local spatial correlations with the observance of the GPS coordinate points' nearby relationships [19–21]. Sometimes there may not exist similar records with the same location in the historical traffic data. Therefore, we attempt to solve this issue by considering the records of distant neighbors. Besides, nearby regions can be relevant and very similar in terms of traffic patterns during various periods. Herein, finding a mechanism capable of integrating relevant spatial and temporal features and simultaneously capturing the complicated dependencies between them can be very helpful. The supplementary critical factors play a significant role in traffic pattern fluctuations, especially within the extreme circumstances of these factors as examples (weather conditions, public holidays, events, and vacations). Thus, we model these features according to the features' correlations and dependencies between each other and also consider the features' contributions to the output. The main contributions of our work can be summarized as follows:

(i) Since data sparsity is a key challenge in real traffic scenarios, we propose a method to solve this issue and achieve better results by splitting the city into N × N grids using geo-hashing techniques and dividing the city into different clusters using the K-means algorithm. This allows us to use neighboring trips if there are no historical records or if the historical records are insufficient.

(ii) We propose a new mechanism to capture both spatial and temporal dependencies. This mechanism comprises two modules: the spatial self-attention module (SSAM) that is used to infer the spatial relationships and the residual dilated convolutional module (RDCM) to capture dynamic time dependencies.

(iii) Moreover, we adopt a multi-head attention approach to learn the attentional weights of a multimodality factor (spatial, temporal, and external) based on their contributions to the target. While many previous works use RNNs in their models, which are time-consuming in the training stage due to their recurrent nature, we use a multi-head attention mechanism that supports parallel computing in this work to dramatically reduce training time.

(iv) We conduct extensive experiments using three large-scale traffic datasets in three different cities (NYC, Chengdu, and Xi'an). The results demonstrate the efficiency of our model compared to other methods under various traffic conditions.

The rest of this paper is organized as follows. Section 2 reviews the related works about the TTF approaches. Section 3 contains the problem definition and formalization, followed by data processing and analysis. Thereafter, we describe our proposed framework (JSTC) in detail. Section 4 discusses the experimental results of our model compared to other models. Finally, a summarized conclusion of this paper is presented in Section 5.

2. Related Work

Generally, TTF methods can be classified into two categories: route-based and OD-based methods.

2.1. Route-Based Methods. Route-based methods can be divided into two approaches.

2.1.1. Segment-Based Method. This method divides the road into segments and then estimates the travel time for each segment individually. Finally, the total travel time for the entire path is the summation travel time of all segments [22, 23]. Many researchers consider the TTF as time series forecasting for a single road, such as the ARIMA model and KF [24, 25], which have been applied in short-term forecasting for road section travel time. In addition, support vector regression (SVR) was used due to its competence and generalization compared to the historical average (HA) method [26]. The gradient boosting decision tree method (GBDT) has been also used to improve prediction accuracy

on TTF problems [27]. Wang et al. [15] investigated the sequence relationship between the road segments. They treated the travel time of the segment as a sequence of time series data and then used the LSTM model to solve this sequence prediction problem. The spatiotemporal hidden Markov method (STHM) was also applied to capture the correlations among different traffic time series and then predict the travel time [28].

2.1.2. Path-Based Method. Another group of researchers combined multiple route segments as an entire path instead of using one road segment to solve the TTF problem. This considers the impact of intersections and traffic lights, which leads to more accurate predictions in the path-based method [21, 29]. A non-parametric technique for route TTF based on floating car data (FCD) is the first to use the path-based approach [30]. It accumulated the travel time of each road segment from a low frequency instead of calculating the travel time of the subpath. Rahmani in [31] also proposed a route-based method for route TTF by combining multi-traffic data sources collected by FCD and automated number plate recognition (ANPR). In [32], the K-shortest path algorithm was developed to infer the possible paths from each OD trip and then predict the link travel time. However, these techniques frequently suffer from dispersed data or the high cost [21]. Nowadays, vast amount of taxi trajectory data is collected by GPS equipment, so the TTF model for a direct path was proposed based on a three-dimensional tensor by applying two essential components; first, compute the travel time for each segment by the tensor decomposition. Then, find the most optimal elements that help to estimate the route's travel time [33, 34]. In [35], a deepIST model was proposed that takes spatial and temporal dependencies of traffic patterns into account by using map image information of the trajectory to predict travel time. In this framework, two CNN-based modules were combined to make images of the route segments and then look for spatial and temporal traffic correlations. To address the data sparsity issue that may occur in some trajectory segments, a CNN with LSTM model named DeepTTE was proposed for raw trajectory data processing [17].

2.2. OD-Based Method. Many scholars have chosen the OD-based methods to address the TTF issues, to minimize the time needed and avoid the complex computations and complicated implementation. In [20], the authors proposed a multi-task representation learning model (MURAT) based on OD data, which achieved promising results. However, this method requires a long processing time and needs a lot of data, which seems to be the main disadvantage of this model.

The estimation of the average time of the urban routes based on the candidates' paths expected between OD trip coordinates was proposed in [19, 32]. They combined the trucks' and taxis' travel datasets to predict travel time between each grid zone, followed by the same methodology in [32], while Faruk in [36] were the first scholars to develop a model for the TTF based on travel distance predicted directly through the OD coordinates' GPS data. However, they ignored delays in intersection queuing, which can reduce the TTF prediction precision. Recently, an ensemble technique with a multi-modality data source model named TTE-Ensemble was proposed in [21]. In this model, the ensemble method was adopted with GBDT and DNN models. GBDT and DNN predicted the travel time separately. Then, each models' results are fed to a decision tree algorithm as a meta-learner model to achieve the final TTF for each OD trip. However, this model basically relied on converting the trajectory data into 2D square cells instead of real OD locations which means that all trips with the same grid ID will have similar characteristics regardless of their distance. Nevertheless, the GBDT and decision tree approaches are unsuitable for big data due to the high computational cost.

Recently, the attention mechanism has been widely used for traffic forecasting. In [37], the authors proposed a pairwise self-attention mechanism for capturing the spatial and temporal dependency of traffic flow prediction. In [18], a deep learning model named FMA-ETA was proposed, which predicted travel time by combining a feed-forward network and self-attention. This model focus on spatial dependencies while temporal correlations were ignored. Besides, convolutional and graph neural networks have been used for spatial and temporal correlations in traffic speed forecasting [38]. A model called GSTGCN, which applies dilated convolutional network architectures to take the advantage of dilation rate by increasing covered spaces between the inputs, was designed.

The literature survey concluded that most of the previously discussed methods did not completely handle the TTF issues and achieve high accuracy due to the complexity of spatial-temporal correlations learning, considering the differentiation of the road network topology and extreme temporal conditions. Also, there are some techniques that could be beneficial for improving the accuracy of travel time prediction. Inspired by the aforementioned ideas, we propose a JSTC framework relying on OD-based strategy, which can achieve high accuracy with promising performance in predicting the travel time for any given OD GPS points. Herein, our work mainly addresses the sparse spatial data problem and also focuses on the multi-component correlations between spatial, temporal, and external factors, which significantly affect the travel time.

3. Methodology

The aim of the traffic forecasting task in this paper is to predict travel time between any pair of locations by means of the observed historical traffic datasets. The general overview of our methodology mainly consists of three main parts: data preparation and preprocessing, analysis of traffic pattern similarity, and introducing our proposed model in detail. To begin, data preparation and preprocessing are critical, which include data cleaning and removal of noise and outliers, feature extraction, and geo-localization (clustering and grid-partitioning). Then, we get through the spatial and temporal dependencies' similarity investigation to observe the influence of these components in traffic patterns' fluctuation.

Finally, we introduce our prediction model, which aims to predict the total travel time of the OD trips accurately. The detailed descriptions of each of these parts are given in the following sections. In advance, we formalize the traffic forecasting problem in this work as in the following key concepts and definitions.

3.1. Preliminaries. We define and formalize the TTF problem as a travel time prediction task between two given points (A) and (B).

Definition 1. OD-trip P $_i$: We define a trip from the historical records as (P), which consists of 5-tuples (o, d, t, **D**, T), where (o) is the pickup location (A), while (d) is the drop-off location (B). Also, (t) denotes the trip time-stamp, which includes the pickup and drop-off times as (t_o) and (t_d), respectively. Both the origin (A) and destination (B) are 2-tuple GPS coordinates, as $o_i = (olat_i, olong_i)$ and $d_i = (dl\ at_i, dl\ ong_i)$, where trip distance (D $_i$) can be obtained from these coordinates. To find the matched historical trips for trip P_i, we define a query (Q) as follows:

$$Q = \{P_i\}_{i=1}^{N}. \tag{1}$$

Definition 2. Spatial and temporal tensors: after splitting a city into N × N grids (G) and K-clusters (C) as a geo-region based on the OD-GPS coordinates, the GPS points have been mapped into G and K as well. We define two 3D tensors $\delta^i \in P^{H_\delta \times F_\delta \times 1}$ and $\tau^i \in P^{H_\tau \times F_\tau \times 1}$ to represent spatial features (δ^i) including pick-up locations, drop-off locations, speed, distance, cluster-ids, grid-ids, and other auxiliary features. Besides, temporal (τ^i) features include the day of the week in-between (0–6), the hour of the day in-range (0–23), and the day of the month as (0–30), where H represents the historical record ID and F denotes spatial or temporal features. Note that we consider the trip features as sequence.

Definition 3. TTF for trip P $_i$: we define the travel time T_i as the total time for the trip P $_i$ from (A) to (B) as follows:

$$T_i = \left[t_{d_i} - t_{o_i}\right]. \tag{2}$$

Hence, the main goal of our work is to estimate the total time (T_i) for an OD-trip (P $_i$) with an assist from the historical trips by a query (Q).

3.2. Data Analysis and Preprocessing. In this paper, we used three large-scale real-world traffic datasets (NYC, Chengdu, and Xi'an) to verify the efficiency of our model across various road network topologies and traffic patterns. The first dataset is the NYC taxi dataset, which is provided by the New York City Taxi and Limousine Commission (TLC) [39] with billions of trip records from 2009 until now and comprises 21 different variables, including GPS coordinates for pick-up and drop-off, pick-up and drop-off time-stamp, total trip distance in miles, and other features. Following [40], we extracted six months of the traffic data between 01/01/2016 and 30/06/2016 for analysis and experiments in our work. The data we have selected contain approximately 75 million records, with over 12 million trips per month and 416,666 trips per day. The other two datasets are Chengdu and Xi'an, which were provided by the "Didi Chuxing platform" containing 9,707,970 and 5,272,758 taxi trajectories in September and October 2018 for Chengdu and Xi'an, respectively. The average trip per day is (123,463 and 133,843) trips, respectively (Table 1).

The analysis of traffic data can greatly assist in recognizing the fluctuations in traffic patterns. Spatiotemporal data cleaning and anonymous value filtration were conducted by removing the invalid or uncharted trips' records that contain missing information in one or more parts of OD GPS location, passenger count, and pick-up/drop-off interval-time records. We consider the trips out of the city boundary as spatial outliers and clean them accordingly. Also, all trips with a distance less than 500 meters and more than 100 kilometers have been cleaned. The temporal components have been filtered by taking only the records with the travel time less than 24 hours (86,400 seconds) and over 3 minutes (180 seconds). In order to observe the traffic patterns over the whole city, 15 regions were classified according to the city's boundaries. Then, each region was grouped by temporal dependencies (day of month and day of week) to obtain the similarity of week and day rhythms. Considering the time-interval of the day as (0–23), we measured the average rate of travel time for all trips within the same spatial and temporal information, as well as traffic intensity for all trips that flow in and flow out across these regions. Figure 1(a) represents the average rate of trip density, and we can see a low-density rate in the period from midnight up to 6 AM. In contrast, we can notice that the maximum density rate happens during two peak periods, from 7 AM to 9 AM as morning rush hours and from 6 PM to 8 PM as the evening rush period. For example, during the early morning and evening rush hours, there is heavy traffic congestion that means the movement will be slow. Therefore, through the non-peak hours, traffic patterns seem to be normal. Note that the average rate of travel time in Figure 1(b) is quite similar to the density rhythm in terms of increase and decrease rate, except for trips with a long duration. So, each trip was considered as one counted trip in the density rate computation, whereas the trip's duration was taken into account while calculating the average rate of travel time, which affects the total average time in this case. Moreover, to determine peak and non-peak periods for Chengdu and Xi'an cities, we did some statistical analysis over various given regions within the same conditions. We randomly chose regions to illustrate the influence of traffic patterns. Table 1 shows that the average traffic volume measured (historical records which enter or leave the cluster or grid) is probably relatively low or high, especially in areas with heavy activity. The results show that the average travel time varies from one region to another according to the traffic rhythms during the hours of the day. On the other hand, traffic density during morning and evening hours is much higher than night and afternoon hours, which explains that traffic overcrowding influences traffic speed and travel time.

Table 1: Statistical analysis of the traffic patterns and fluctuations in particular locations in Xi'an and Chengdu cities.

Day	10 Oct #Xi'an⟍#Chengdu	13 Oct #Xi'an⟍#Chengdu	15 Oct #Xi'an⟍#Chengdu
Pick_Hour	9, 11, 20	8, 10, 16	7, 14, 19
Pick_location_Grid	136⟍20	136⟍20	136⟍20
Drop_location_Grid	39⟍38	39⟍38	39⟍38
Avg_traffic_volume/grid	2593, 2301, 2311⟍1610, 1381, 1370	1470, 1633, 2177⟍1551, 1338, 996	2368, 1405, 2563⟍1457, 1238, 706
Trip_distance (km)	~ 8.5⟍ ~ 5.3	~ 8.5⟍ ~ 5.3	~ 8.5⟍ ~ 5.3
Trip_speed (kms)	30.7, 33.3, 57.06⟍19.1, 12.1, 17.5	39.1, 33.8, 23.9⟍10.3, 19, 26.5	24.5, 41.5, 52.6⟍11.3, 19.1, 23.9
Trip_duration (sec)	1372, 1368, 744⟍1178, 1986, 1324	1076, 1242, 1761⟍2187, 1178, 951	1719, 1014, 798⟍2394, 1263, 923

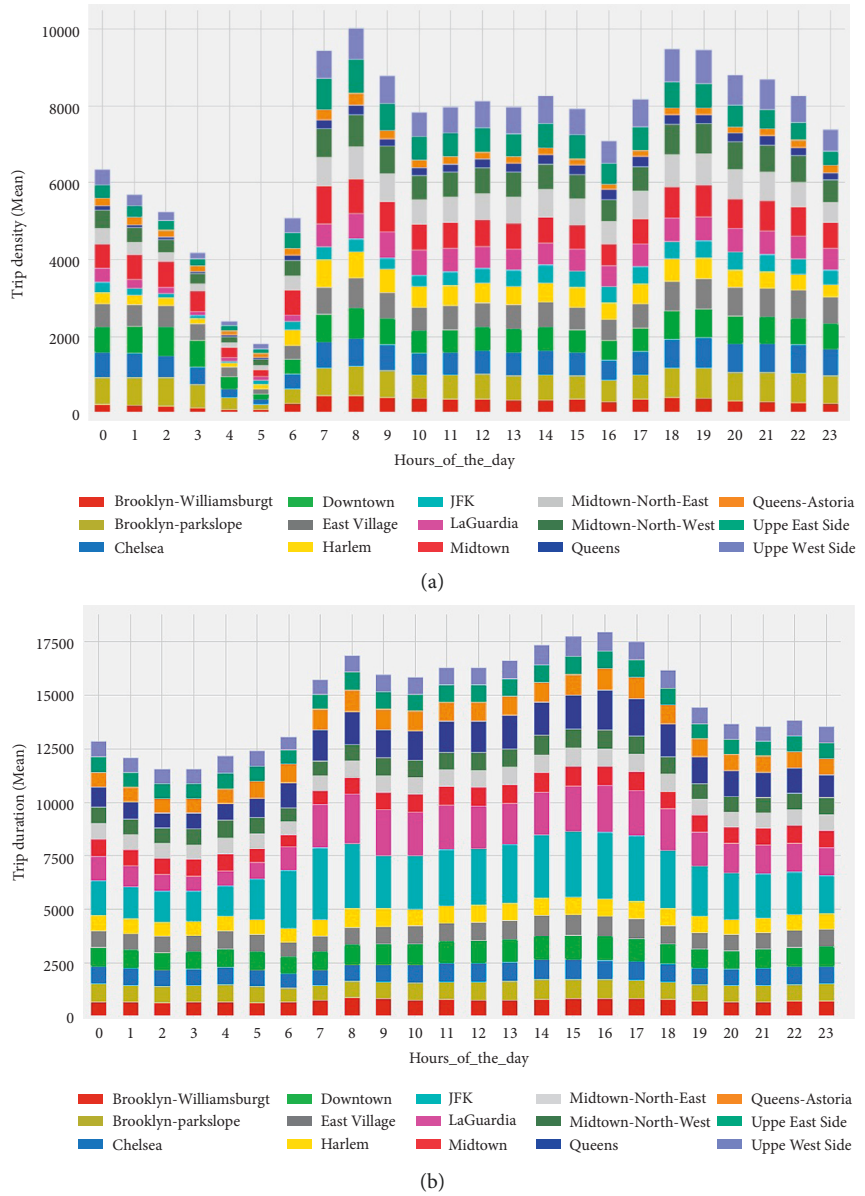


Figure 1: Traffic pattern visualization for city boroughs. (a) Average rate of trip density. (b) Average rate of trip duration.

Eventually, to ensure that our proposed model is capable of producing effective results, after investigating the traffic patterns' similarities, two peak periods have been adopted for NYC, Chengdu, and Xi'an as the morning and evening peak periods, which include (7 ~ 10 AM and 5 ~ 8 PM), respectively.

3.3. Feature Extraction and Data Preparation. Similar to [21], we apply data preprocessing based on the perspective of multi-modality. Thus, accurate prediction of TTF is greatly influenced by numerous dynamic components, including complicated spatial and temporal dependencies, and the influence of external factors such as weather status, social events, or public holidays [41, 42]. Hence, to improve the prediction accuracy, we adopted three components in our proposed method: spatial, temporal, and external. We adopt two 3D tensors δ^i and τ^i for spatial and temporal components' representation, while the external components were divided into two subvectors: weather data and public holiday data.

3.4. Spatial Components. The original dataset provides the trips' pick-up and drop-off GPS locations only, so we further extracted additional spatial features from these two points such as distance and speed, which are essential spatial features. We applied two different methods to calculate the distance between two GPS locations. The two methods are the Manhattan and haversine distance approaches [40]. Manhattan distance is formulated as follows:

$$\left|\triangle \mathrm{lat}_{P_i}\right| + \left|\triangle \mathrm{lon}_{P_i}\right|, \tag{3}$$

where $\triangle \mathrm{lat}_{P_i}$ and $\triangle \mathrm{lon}_{P_i}$ denote the total distance difference between the ordered pairs of OD coordinates computed by the following equations:

$$\begin{aligned} \triangle \mathrm{lat}_{P_i} &= \left|o_i\mathrm{lat} - d_i\mathrm{lat}\right|, \\ \triangle \mathrm{lon}_{P_i} &= \left|o_i\mathrm{lon} - d_i\mathrm{lon}\right|. \end{aligned} \tag{4}$$

The haversine distance is also formulated as follows:

$$2r\arcsin\sqrt{\sin^2(\triangle\phi/2) + \cos(o_i\mathrm{lat})\cos(d_i\mathrm{lat})\sin^2(\triangle\lambda/2)}, \tag{5}$$

where $(\triangle\phi)$ is $(\triangle \mathrm{lat}_{P_i})$ and $(\triangle\lambda)$ is $(\triangle \mathrm{lon}_{P_i})$.

Furthermore, the average speed was calculated regarding the trip distance and trip duration. In addition, we extracted other supplementary spatial features from the GPS coordinates, for example, cluster and grid density, which are explained in Definitions 4 and 5, respectively. In the real-world road network, traffic patterns' variation is highly related to time (e.g., traffic tidal phenomena during the weekdays) and space, including neighboring regions. Thus, the traffic patterns in neighboring regions are more relevant. Generally, traffic in neighboring regions exhibits similar flows over the day-time intervals.

To improve the proposed model's performance, we applied the K-means clustering method in the spatial component preprocessing phases. Since K-means attempts to group places based solely on their Euclidean distance, it returns clusters of places that are close to each other and geo-positioning trips within nearby regions into the same cluster. In order to determine whether we are using the right number of clusters, we applied the elbow curve method [43] based on calculating the sum of squared errors (SSE) for a range of values of k (60, 80, 100, 120, and 150) and then picking the elbow of the curve as the optimal number of clusters to use by choosing a small value of k that still has a low SSE. From Figure 2, we can observe that the optimal value of K is 100.

Similarly, we mapped each OD-trip into 2D**D** grid cells with an area of approximately 0.5 km × 0.5 km. Thus, we can represent each trip with two grid-ID features, one for pick-up and the other for drop-off. Finally, after the clustering and geo-location mapping processing, the degree of crowding for each part (cluster and grid) throughout the city is computed depending on the following definitions.

Definition 4. Density score for cluster:

$$\mathrm{Cluster}_{\mathrm{density}}(d_C) = \sum_{i=1}^{N} o_{C_i} + \sum_{i=1}^{M} d_{C_i}. \tag{6}$$

Definition 5. Density score for grid cell:

$$\mathrm{Grid}_{\mathrm{density}}(d_G) = \sum_{i=1}^{N} o_{G_i} + \sum_{i=1}^{M} d_{G_i}, \tag{7}$$

where N and M represent the total number of origin (o) and destination (d) trips' locations recorded within the same cluster (C) and grid (G) at time interval of the day. These two spatial features are essential to reflect the traffic flow of the region through different periods.

3.4.1. Temporal Components. The temporal features are significant factors to understand travel time changes through time variation. Therefore, trip duration is affected by several temporal factors, which may occur daily, weekly, or seasonally [44]. The rhythm of commuters' flow over workplaces, schools, and even public places is an example of activities that cause traffic jams at various times. To this end, the following temporal features were extracted from the traffic datasets, using the one-hot encoding (OHE) and label-encoding techniques as follows:

(i) We represent the day of the month as a label value from 0 to 30.

(ii) We represent weekdays as a categorical value from 0 to 6.

(iii) We represent hours of the day as a label value from 0 to 23.

(iv) Working days and weekends take 0 or 1.

3.4.2. External Components. The external factors were divided into two parts: weather conditions and public holiday. Generally, the trip is affected by one or more of the following weather conditions (heavy rain, snow, storms, and so on). Different weather conditions can also result in varying travel times with similar spatial patterns and different interval times. Hence, the weather is considered as an important external factor in this work. Table 2 shows the weather data categories, which are classified into 10 types (sunny, cloudy, rainy, windy, and so on). Also, three more features are used

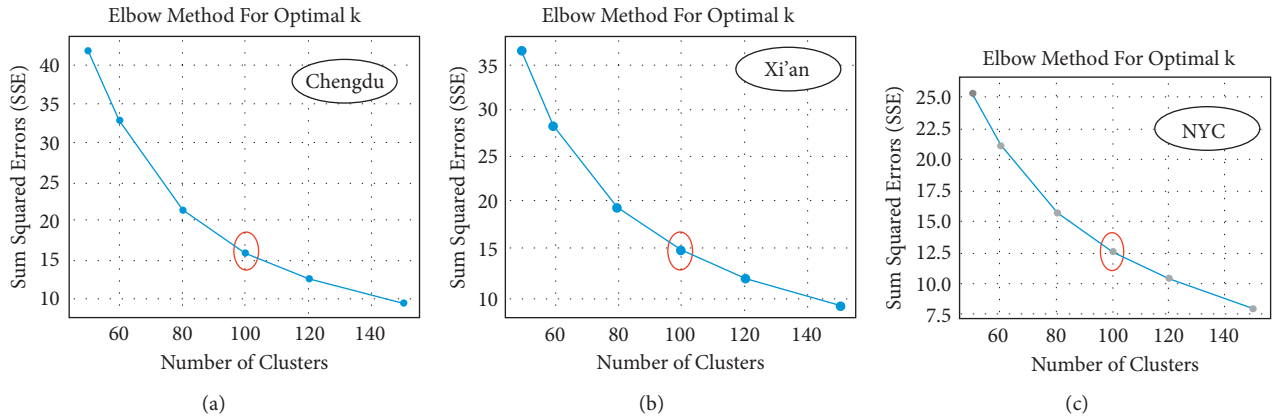


Figure 2: Investigation of selecting K-means clustering value using elbow curve method.

Table 2: Weather data formalization and labeling.

Weather condition	Label
Clear	1
Overcast, partly cloudy, mostly cloudy, scattered clouds	2
Haze, fog	3
Light freezing fog, light freezing rain,	4
Light snow	5
Rain	6
Snow	7
Heavy rain	8
Heavy snow	9
Light rain, sleet	10

to describe the weather situation of trips circumstances in terms of extreme weather conditions (snowing, raining, or foggy). There are 16 different types of weather conditions, according to the historical weather data provided in [45]. Thus, this classification process makes similar weather conditions much closer and helps to reduce the data dimensions. Because of variable weather conditions, the same spatial locations in terms of OD-grids may not have the same trip times, as shown in Figure 3. This figure shows that when the weather is regular, travel time between the same origin and destination grids takes less time than hours characterized by extreme weather conditions when comparing two different days.

Besides the factors mentioned above, the traffic patterns during public holidays and events can differ from those of the daily routine, due to increased outdoor activities or variation in daily traffic patterns, leading to extreme traffic jams. As a result, two subcategorical features are concluded from the NYC and China public holiday datasets to represent whether the day is a holiday or not. Eventually, externals are classified into two types: categorical features by using the OHE technique and discrete features. Furthermore, data standardization and scaling techniques for features have been utilized.

3.5. JSTC Model Architecture. Our proposed framework mainly comprises three modules, as shown in Figure 4. The first block is designed to learn the dependencies between spatial and temporal components and capture their complicated relations. This block also helps to capture the correlation between grids and clusters for OD-trips during different time patterns, especially when observing adjacent locations' properties and dealing with the sparse data. After processing the external features, we combine all feature representations and pass them to the last block, which is the multi-head attention module to learn the attentional weights of all features based on their contribution to the output. Next, we describe each part in detail.

3.5.1. Spatial Self-Attention Module. In this section, we develop a self-convolutional attention mechanism that captures the correlations across different spatial features and learn their attentional weights. To this end, we adopt a 1D convolutional layer followed by self-attention heads. Figure 5 shows our proposed spatial self-attention module, and the spatial feature's tensor includes a pair of GPS coordinates, a pickup cluster, a drop-off cluster, a pickup grid, a drop-off grid, distance, and speed {D and S}. First, we reshape the input into three dimension as an input for the 1D convolutional layer. To do so, we used a reshape function to reshape the 2D features vector into 3D tensor δ^i. Then, we used the convolution filter and kernel size as shown in Figure 5 to handle the spatial input tensor. Thus, we can get *Query* $\{Q^{\delta}\}$, *Key* $\{K^{\delta}\}$, and *Value* $\{V^{\delta}\}$ as an output from each 1D-Conv layer followed by the ReLU activation function as follows:

$$Q^{\delta} = \omega_q^f \cdot \chi^{\delta}, \quad K^{\delta} = \omega_k^f \cdot \chi^{\delta}, \nu^{\delta} = \omega_V^f \cdot \chi^{\delta},$$
$$\text{Conv1}d_{(K,Q,V)^{\delta}} = \chi_{i,j}^{\delta} = \sum_{j=1}^{J} \omega_f^{(j)} \odot \chi_{(i+\kappa)} + \beta^j, \tag{8}$$
$$\text{ReLU}(\chi) = \text{Max}(0, \chi),$$

where χ denotes the tensor input, i is the convolution processed index, j refers to the filter (f) position, and κ is the kernel size. (ω_f^j) represents the filter (f^j) weight matrix, and (β^j) is the learnable parameter (bias). We set

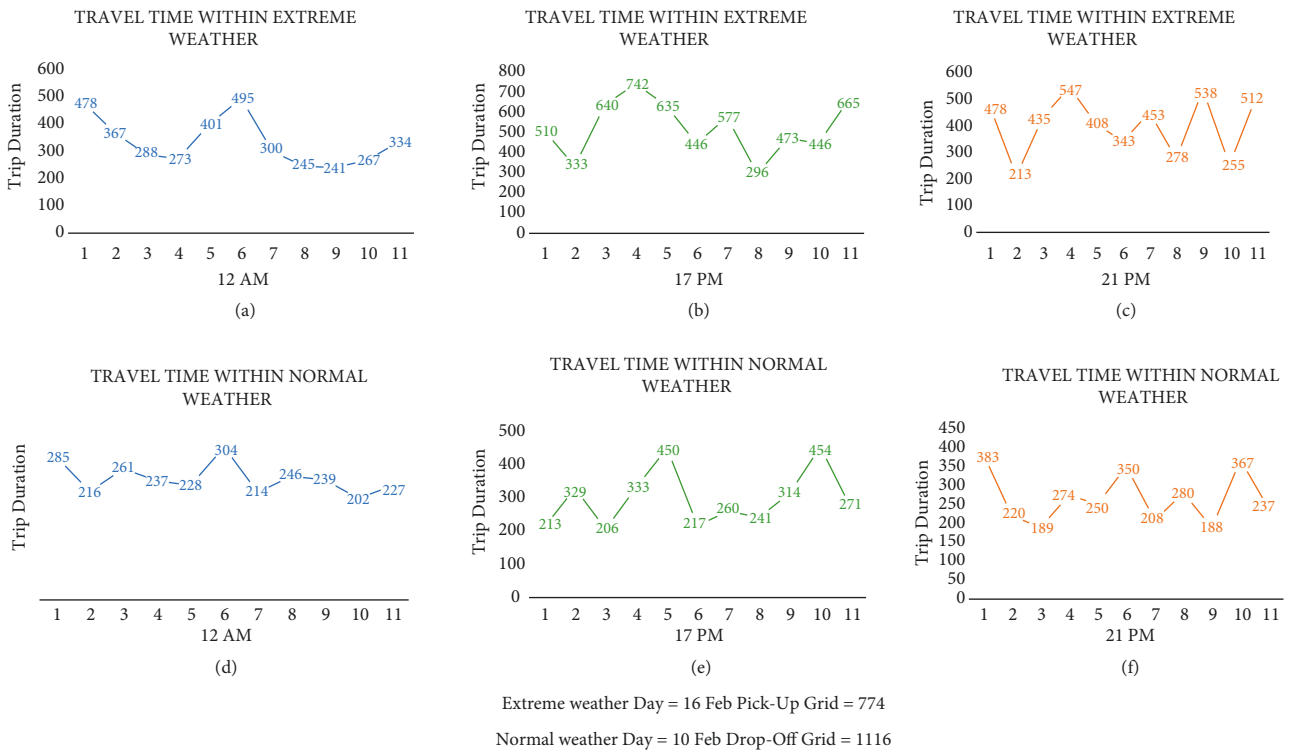


FIGURE 3: Visualization of travel time changes when the weather is different, for example, from pick-up grid-id = 774 to drop-off grid-id = 1116 on the NYC dataset on two different days over the same time interval.

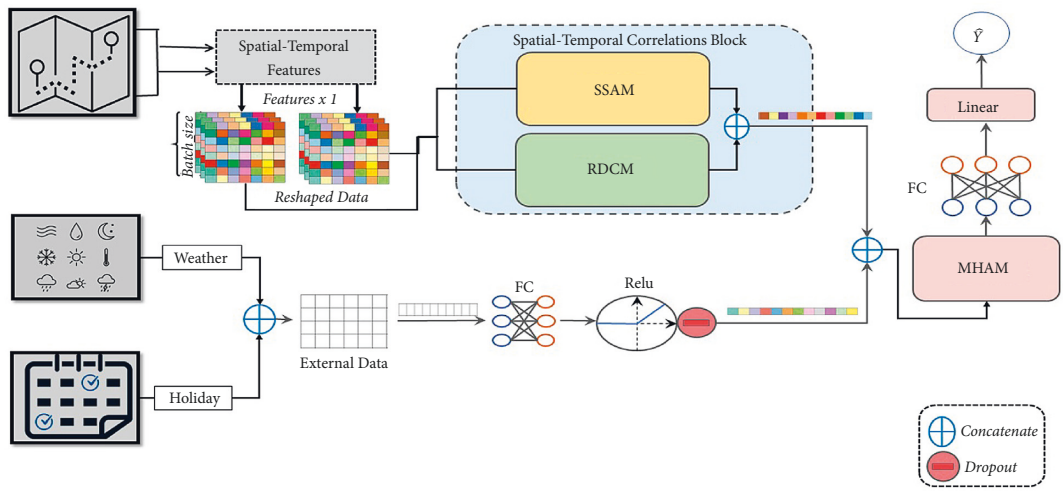


FIGURE 4: Joint spatial and temporal correlation (JSTC) mechanism architecture combines spatiotemporal correlation block, which includes the spatial self-attention module (SSAM) and residual dilated convolutional module (RDCM). Then, we used a multi-head attention module (MHAM).

the filter and kernel size to 1 and 3, respectively. We set the padding to "same" to avoid dropping some information and verify that all inputs are completely represented. Therefore, the weight matrix $(\breve{U})$ between K^{δ} and Q^{δ} is computed by using the scaled dot attention function, and then the final attention score $(\breve{W}^{\delta})$ is computed as in the following equation:

$$\begin{aligned} \breve{U} &= K^{\delta} \odot Q^{\delta}, \\ \breve{W}^{\delta} &= \breve{U} \odot V^{\delta}. \end{aligned} \tag{9}$$

Afterward, the final attention output is obtained over the multiple self (attention) layers, and then we flatten the output of the spatial self-attention block and concatenate it with the temporal correlation output.

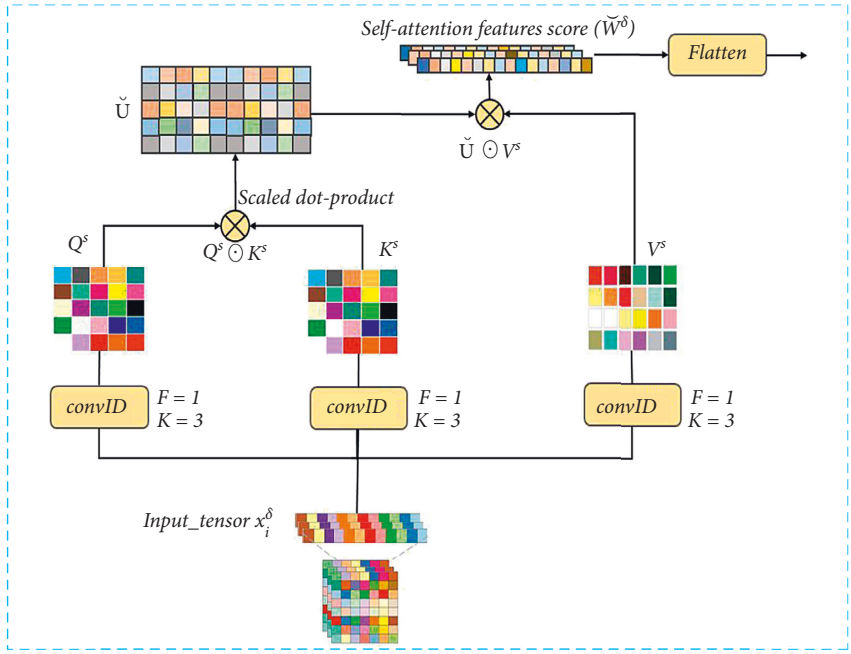


Figure 5: The structure of spatial self-attention module (SSAM).

3.5.2. Residual Dilated Convolutional Module. The temporal convolutional module aims to capture the temporal patterns. Several previous studies have considered the temporal dependencies of traffic forecasting tasks. In [46, 47], the RNN architecture was applied to capture temporal relations, while references [48, 49] utilized the gated recurrent units (GRUs) and long-short memory (LSTM) networks to model the temporal components on traffic pattern fluctuations. Although these approaches have shown good performance, they still suffer from many problems (e.g., exploding/vanishing gradients, time-consuming in the training phase, and some other limitations in modelling long sequences).

Inspired by the recent success of the temporal convolutional network (TCN), we propose a residual temporal correlation module (RDCM), which comprises multiple dilated 1D-Conv layers stacked together as shown in Figure 6. We employed the TCNs advantages in the convolutional operations expanding domain by adjusting the dilation rate parameter on each layer. Empirically, same as the preprocessing we have used for the spatial components, we construct 3D tensor (τ^i) for the temporal features. Since the traffic patterns during the different periods of the day are highly affected by the traffic flow in each region. Accordingly, while investigating the dependencies of temporal factors, some spatial features should be considered due to their significant impact on the output. In our case, the density score grid and cluster for both pick-up and drop-off, which are measured hourly, have been adopted as supplementary features for the temporal correlation modelling. By now, the temporal component of each trip record is represented by the (χ_i^{τ}) tensor, which includes the temporal features and the supplementary features.

In order to capture the interactions and patterns of temporal features in terms of long-short dependencies between the input features, we built three dilated convolutional layers with different "dilation −rates" as = {1, 2, 4} to address the following two key points: avoiding the backpropagation issue (gradient vanishing or exploding) and receptive field expansion to cover the entire input's representation through the shallow hierarchical layers. Thus, to achieve the normal convolution operation, we set the dilation "$d^r = 1$" and the kernel-size "$K = 3$" in the first layer followed by ReLU and drop layers, and then the output is used as an input for the next dilated convolution layer with "$d^r = 2$" and "$K = 3$." Then, "$d^r = 4$" and "$K = 5$" for the last layer. Figure 6(b) illustrates the dilated convolution steps. As a result, we make sure that the different space (long-short) of the relationship between the temporal factors has been considered. Also, an efficient representation of the features without missing any important information is also considered. The dilated convolutional layers were combined into a residual block, and an element-wise concatenation layer was used to add the last output to the input (χ_i^{τ}), which can improve training and maintain an optimal feature correlation distribution. In this paper, we formulated the DRCM block operations as follows:

$$f\left(\tau_i^{dr}\right) = f\left(\chi_i^{\tau}\right) + \chi_i^{\tau},$$
$$f\left(\chi_i^{\tau}\right) = \sum_{s=1}^{S} \left[\chi_i^{\tau} + d^r \circledast s\right] \omega[s], \tag{10}$$

where d^r denotes the "dilation −rate" and s denotes the "filter− size." Eventually, the temporal correlation output is concatenated with the previous spatial correlation outputs and passed to a multi-head attention mechanism.

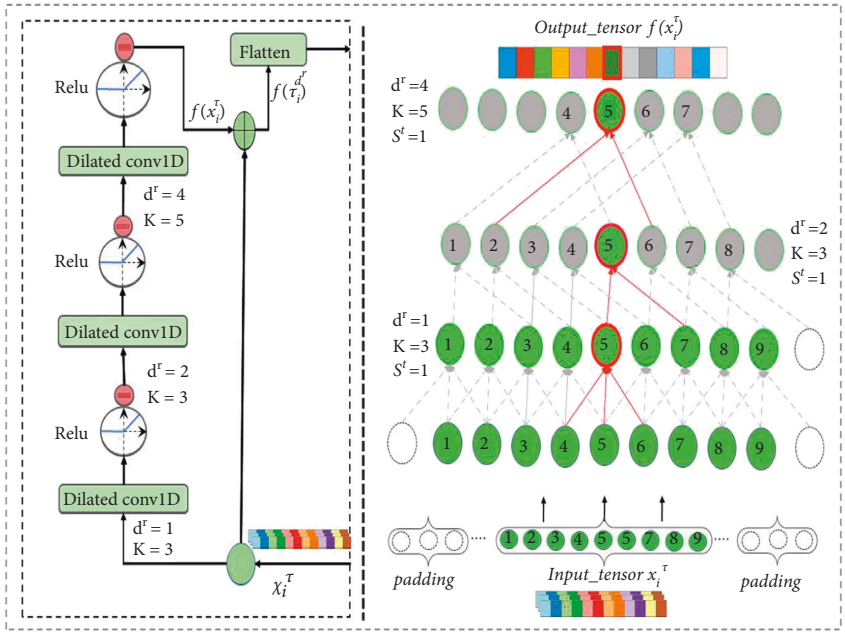


FIGURE 6: An illustration of our proposed residual dilated convolutional module (RDCM). On the left side of the figure, we show the RDCM module's architecture. The right side represents the dilation convolutional operations by expansion of the receptive field (dilation parameter) and different sizes of kernels (K) to obtain optimal feature representations.

3.5.3. Multi-Head Attention Module. The multi-head attention mechanism is illustrated on the right side of Figure 4 as reported in [42], which has been adopted in our model in charge of getting accurate prediction results. First, due to the impact of the external features on the travel time as mentioned before, we apply a fully connected layer followed by ReLU and dropout layers as subblock to represent the external factors (weather details and public holidays), and then we combine the external features' representation vector with the vector that represents the spatial and temporal correlations outputs (for more details, see Sections 3.4.1 and 3.4.2). By implementing this mechanism, we can enhance our model's ability to learn the attentional weights of various features using multiple attention layers. Besides, it makes the training process robust and fast where it guarantees processing strategies across multiple (H_{Att_h}) heads. Thus, from the concept of learning the attentional weights of all features based on their contribution to the output. In this study, the attention scores represent the intercorrelations of the input features to the target (travel time). Therefore, we applied a "scaled-dot" function to compute the attention score based on the contribution of each feature to the output target. To do so, we constructed (query (Q), key (K), and value (V)) vectors, which include the feature representations. Firstly, we can get the features' scores (weights) between each feature in (Q) and the set of keys, and then the second round of dot-product function takes these scores' (weights) vector and set of keys (K) to get the values' (V) vector, for calculating the final attention score. We formally defined this process as follows:

$$\begin{aligned} MH_{\text{Att}}(Q,K,V) &= \text{Concat}\left(H_{\text{Att}_1},\ldots,H_{Att_h}\right)W^{\mathbb{O}}, \\ H_{\text{Att}_i} &= \text{Attention}\left(QW_i^{\mathbb{Q}},KW_i^{\mathbb{K}},VW_i^{\mathbb{V}}\right), \end{aligned} \tag{11}$$

where $QW_i^{\mathbb{Q}}$, $KW_i^{\mathbb{K}}$, and $VW_i^{\mathbb{V}}$ represent the (K, Q, and V) weights for each head and $W^{\mathbb{O}}$ is a combination of scores'/weights' matrix. h is the number of head parameter; after several trials with the h values $\{4, 6, 8, 10\}$, we adopted 6 as the number of attention heads, which leads to fast performance and achieves optimal results.

Eventually, we use a dense layer followed by a linear operation to get the final prediction results $(\hat{y}^i_{OD_\tau})$ ideally as follows:

$$\hat{y}^i_{OD_\tau} = \varphi\left(W_f\chi^i + b_f\right), \tag{12}$$

where (φ) is the linear activation function and (W_f) and (b_f) are learnable parameters.

4. Experimental Results and Analysis

We used three large-scale traffic datasets (NYC, Chengdu, and Xi'an) in our experiment. Section 3.2 describes in detail the data analysis and preprocessing. We randomly split the datasets into 80% for training and 20% for testing. The training set was then divided into two subsets: 70% for model training and 30% for validation. The learning rate values range (0.01, 0.001, and 0.0001), batch size as (128, 256, and 512), dropout values range (0.1, 0.2, and 0.3), and multi-head (h) as (4, 6, 8, and 10). The optimal values for parameters are as follows: the learning rate is 0.001, the number of training epochs and attention heads is (60 and 6), respectively, and batch size is 512. Besides, to reduce overfitting, we applied both the kernel regularizer (L2 norm) and dropout (0.2). Also, we adopted the Adam optimizer as an optimizing function with a linear activation function.

4.1. Evaluation Metrics. To evaluate our model, we use two common prediction metrics.

Mean absolute percentage error (MAPE) is calculated as

$$\text{MAPE} = \frac{100\%}{N}\sum_{i=1}^{N}\left|y^i - \hat{y}^i/y^i\right|. \tag{13}$$

Mean absolute error (MAE) is calculated as

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|y^i - \hat{y}^i\right|, \tag{14}$$

where y^i and $\hat{y}^i$ are the actual and predicted OD-trip durations in seconds, respectively. N indicates the total number of records in the test dataset.

4.2. Results

4.2.1. Comparison of Various Models' Results with JSTC Model. To show the performance efficiency of our model, we compared it with the following models:

(i) LRM: we applied the LR model in [20] with almost all features except the grid and cluster, which have a high dimension and cause overflow.

(ii) XGBoost: a machine learning model widely used for both classification and regression problems. However, XGBoost with a deep tree may lead to better predictions. Following [50], we set the max-depth parameter between 4 and 6 to avoid overfitting.

(iii) LightGBM: the LightGBM model is based on decision tree algorithm with leaf-wise and level-wise. This model is more appropriate for large datasets with large dimension of features [51]. Accordingly, we set the LightGBM parameters same as in [21].

(iv) ST-NN: spatiotemporal-based model was proposed in [19], which combined two DNN modules to predict the trip distance and then used this distance to predict the travel time.

(v) TTE-Ensemble: the collaborative model proposed in [21] combines machine learning and neural network (GBDT and DNN) modules for modelling multi-modality data to predict the OD-trip travel time.

(vi) FMA-ETA [18]: a deep learning model based on a multi-self-attention technique integrated with a feed-forward structure (FFN) for capturing spatial and temporal dependencies and obtaining TTF.

(vii) STTNs [37]: two spatial-temporal blocks are integrated into an approach based on graph neural network and transformer (STTNs), which jointly investigates the dynamic spatial and temporal dependencies to enhance the traffic flow prediction result's accuracy.

Table 3 illustrates our model results compared with other models in terms of MAPE and MAE for the NYC, Chengdu, and Xi'an datasets. The results show that our model outperforms other approaches. As previously mentioned, we divided the comparative models into two parts (ML and DL models). The results of ML (LR, XGBoost, and LightGBM) models show worse accuracy compared with the DL models because these simple statistical ML algorithms have difficulty in modelling the non-linearity relations of complex traffic patterns. We notice that the LR model gives the worst results compared to others (26.12, 24.37, and 25.85) in MAPE and (168.34, 176.33, and 197.14 sec) in MAE for NYC, Chengdu, and Xi'an, respectively. The error rate (MAE) was reduced by (14.4, 14.14, and 9.11 sec) and (18.62, 20.94, and 20.88) with the XGBoost and LightGBM models, respectively. In contrast, our model shows better performance where it reduces the errors by approximately (71.22, 104.4, and 108.26 sec) compared with LR and (56.82, 90.26, and 94.15) on XGBoost, while (52.6, 83.46, and 82.38) for LightGBM on NYC, Chengdu, and Xi'an, respectively.

On the other hand, the ST-NN achieved the lowest results of all the DL models because it only utilizes two MLP blocks. In comparison, our model reduced the errors (MAPE) by at least (~ 7%) on NYC and Chengdu, while 6.31% on Xi'an. Furthermore, our model has also shown remarkable superiority over the TTE-Ensemble model by reducing the errors by (5.19%, 5.5%, and 4.73%) on NYC, Chengdu, and Xi'an, respectively. Thus, we can observe that ST-NN and TTE-Ensemble models achieved better results than ML algorithms (LRM, XGBoost, and LightGBM). This is because deep learning approaches consider the non-linear relations between the variables. Although, the ST-NN applied two DNN modules for estimating the trip distance first, then using this distance to predict the time, which means they also adopted the spatial component (distance) only, while the temporal patterns was ignored. The TTE-Ensemble model was built based on combining the DNN module with the ML (GBDT) model. These models are not sufficient to capture the complicated correlations.

Eventually, as it can be seen from the table, FMA-ETA and STTN models give results which are more closer to our proposed model because these models have also adopted attention mechanisms to capture the non-linear correlations between the spatial and temporal features. The auxiliary spatial features that influence traffic patterns play a significant role when considering the dynamic scales of inner spatial and temporal correlations.

Therefore, compared to FMA-ETA, our proposed model components (SSAM, RDCM, and MHAM) play a significant role in reducing the MAPE and MAE error rates by (2.67%, 3.74%, and 1.91%) and (15.09, 35.42, and 27.25 sec). Also, our model achieves better performance than the STTN model through reducing the errors by at least (1.24%, 2.17%, and 1.61%) and (8.11, 22.68, and 16.78 sec) on MAPE and MAE for NYC, Chengdu, and Xi'an, respectively.

Moreover, to validate our model, two different datasets at morning peak (7 to 10 AM) and evening peak (5 to 8 PM) have been used to test all models during these two periods in terms of MAPE and MAE, as shown in Tables 4 and 5 for NYC, Chengdu, and Xi'an, respectively. Prediction errors are typically higher during these two peak periods than

Table 3: Comparison of all models' results on the NYC, Chengdu, and Xi'an datasets.

Model	NYC		Chengdu		Xi'an	
	MAPE	MAE (sec)	MAPE	MAE (sec)	MAPE	MAE (sec)
LRM	26.12	168.34	24.37	176.33	25.85	197.14
XGBoost	25.39	153.94	22.59	162.19	23.37	188.03
LightGBM	22.19	149.72	21.98	155.39	21.51	176.26
ST-NN	20.04	136.34	19.02	131.26	20.44	154.07
TTE-Ensemble	18.33	122.71	17.58	114.08	18.86	136.35
FMA-ETA	15.81	112.21	15.74	107.17	16.04	121.13
STTNs	14.38	105.23	14.25	94.61	15.74	110.66
JSTC	**13.14**	**97.12**	**12.08**	**71.93**	**14.13**	**93.88**

We denote our model's results in bold font as the best scores for each metric.

Table 4: An illustration of all models' performances with morning and evening peak periods (MAPE) for NYC, Chengdu, and Xi'an.

Model	NYC MAPE		Chengdu MAPE		Xi'an MAPE	
	Morning	Evening	Morning	Evening	Morning	Evening
ST-NN	25.42	26.33	24.12	25.74	26.04	27.16
TTE-Ensemble	22.36	23.65	21.01	23.66	23.75	24.52
FMA-ETA	20.52	21.33	18.77	21.82	20.16	22.93
STTNs	17.78	19.42	16.34	18.25	17.66	19.38
JSTC	**15.82**	**17.13**	**14.52**	**16.04**	**16.27**	**18.45**

We denote our model's results in bold font as the best scores for each metric.

Table 5: An illustration of all models' performances with morning and evening peak periods (MAE) for NYC, Chengdu, and Xi'an.

Model	NYC MAE		Chengdu MAE		Xi'an MAE	
	Morning	Evening	Morning	Evening	Morning	Evening
ST-NN	159.02	164.34	166.62	168.87	163.46	170.19
TTE-Ensemble	145.28	154.11	150.84	155.43	150.22	161.82
FMA-ETA	128.61	133.72	126.96	140.48	137.63	144.57
STTNs	119.83	125.61	117.13	128.72	121.75	134.44
JSTC	**108.74**	**115.93**	**98.31**	**118.49**	**106.59**	**125.16**

We denote our model's results in bold font as the best scores for each metric.

during non-peak periods. From the results shown above, we can demonstrate that our model provides more accurate results compared to other models, even during the morning and evening peak hours. Also, based on the random selection of trips used for testing our proposed model, Figure 7 shows a comparison between actual values and predictions of 50 random trips for all models on NYC, Chengdu, and Xi'an, respectively. Each point on the X-axis represents a trip from the test set, while the y-axis indicates the trip duration in seconds.

4.2.2. Ablation Analysis. We built our model based on three main components (SSAM, RDCM, and MHAM). Besides, we consider external factors that influence travel time by improving the accuracy of our results. Therefore, additional experiments were conducted to verify the contribution of each component in our prediction task. The ablation models we use in this analysis are as follows:

(1) Without SSAM: in this model, we removed the spatial self-attention module (SSAM) and applied RDCN and MHAM modules only with a fully connected and output layer.

(2) Without RDCM: in this model, we removed the DRCM module and applied SSAM and MHAM modules only with a fully connected and output layer.

(3) Without externals: to verify the effect of external factors (weather and public holidays), we remove the block responsible for representing these factors' dependencies.

(4) Without MHAM: we removed a multi-head module (MHAM). So, after getting the spatial and temporal components' correlations, we concatenate these blocks' outputs with external features' representations and then apply fully connected and output layers directly.

We should mention that MLP layers were adopted as an alternative to each module that was removed during the ablation investigation phases 1, 2, and 4, as shown in Table 6. However, the impact of externals was just measured by removing the external factors' representation block in ablation 3. The results in Table 6 demonstrate that the performance of all modules combined together in one model leads to better results. In contrast, removing some parts affects the process of capturing traffic pattern fluctuations.

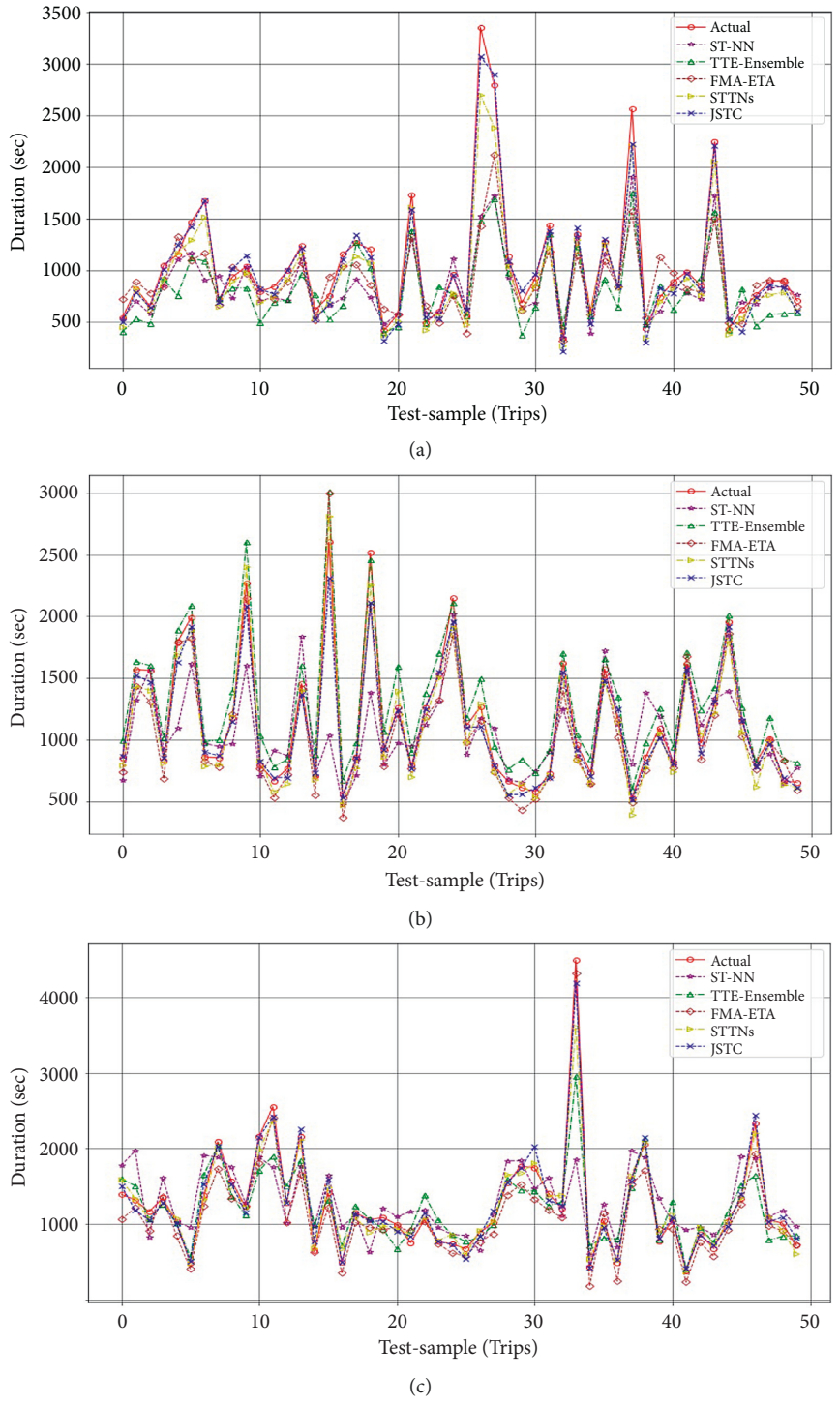


Figure 7: Comparison of prediction vs. actual values for all models on (a) NYC, (b) Chengdu, and (c) Xi'an.

We can notice that the MAPE increases by (8.22, 8.41, and 8.63) and the MAE increases by (55.15, 67.23, and 70.29 sec) in NYC, Chengdu, and Xi'an, respectively, when removing the SSAM block, while removing the RDCM block increases the MAPE by approximately (6.95, 6.25, and 6.28) and MAE by (47.66, 53.98, and 55.5 sec). By removing external factors, the MAPE increases by at least (4.29, 4.85, and 4.92) and the MAE increases by (93.68, 46.44, and 41.66 sec). That means applying external factors improves our model's results by a significant margin. Whereas, applying the SSAM and RDCM modules combined with external factors representation without the MHAM block, we achieve results with small errors rates (MAPE%) about (3.67, 2.87, and 3.45) and MAE at least by (21.42, 38.39, and 27.38 sec) for NYC, Chengdu, and Xi'an, respectively, compared to combining all the JSTC model's components. We can observe that disabling joint

Table 6: Impact of the JSTC model's components: SSAM, RDCM, MHAM, and external factors.

Feature	NYC		Chengdu		Xi'an	
	MAPE	MAE (sec)	MAPE	MAE (sec)	MAPE	MAE (sec)
Without SSAM	21.36	152.27	20.49	139.16	22.76	164.17
Without RDCM	20.09	144.78	18.33	125.91	20.41	149.38
Without externals	17.43	136.82	16.93	118.37	19.05	138.54
Without MHAM	16.81	118.54	14.95	110.32	17.58	121.26
JSTC	**13.14**	**97.12**	**12.08**	**71.93**	**14.13**	**93.88**

We denote our model's results in bold font as the best scores for each metric.

Table 7: Comparison of time consumption of different models on all datasets.

Model	NYC		Chengdu		Xi'an	
	Train (1 epoch) sec	Test (1M trip) sec	Train (1 epoch) sec	Test (1M trip) sec	Train (1 epoch) sec	Test (1M trip) sec
ST-NN	157	122	110	102	114	107
TTE-Ensemble	214	155	119	117	122	115
FMA-ETA	221	218	124	120	137	126
STTNs	244	223	143	127	151	133
JSTC	253	229	168	129	170	136

correlation mechanisms (SSAM and RDCM) increases the error rates more than removing a multi-head block, which means these two modules have a higher impact on our model since they are responsible for capturing correlations of traffic spatial and temporal factors. On the other hand, external factors play an important role in improving our prediction results. Conclusively, these results emphasize the importance of each proposed block through their contributions to improving travel time prediction results.

4.2.3. Computational Cost Measurement. Measuring the computational complexity has been considered in this paper. We compute the time consumption of our model compared with deep learning-based models (ST-NN, TTE-Ensemble, FMA-ETA, and STTNs). Table 7 reports the average time of training and predicting functions for one million trips (1M) with only one epoch on NYC, Chengdu, and Xi'an datasets. Note that we performed our experiments on the same NVIDIA GPU (GeForce GTX 1050 Ti) with 4 GB. Also, we set the batch size to 512 for all models' training phase. Thus, we could observe that the complicated model's structure took more training time than the simple ones. Actually, one logical reason is that this model's complexity represents an improvement to give more accurate prediction results. In comparison, we can notice that the computation time of our model is much closer to that of the STTN model due to the fact that both models have a relevant structure.

5. Conclusion

In this paper, we first discussed the various characteristics of traffic patterns that affect travel time. Then, we presented a mechanism for capturing interactions between spatial and temporal factors based on self-convolutional attention and dilated convolutional techniques. In addition, we adopted spatial auxiliary features and integrated them with the temporal features, which play a significant role in capturing the dynamic traffic patterns and their correlations. Furthermore, we applied a multi-head attention mechanism to learn the attentional weights of the spatial, temporal, and external features based on their contribution to the output and speed up the training process. Extensive experiments using three large-scale real-world traffic datasets (NYC, Chengdu, and Xi'an) have shown that our JSTC model outperforms prior methods.

Data Availability

The terms of use of the data used in this study do not allow the authors to distribute or publish these datasets directly. However, data can be obtained directly from the following webpages: NYC Taxi and Limousine Commission (TLC)—https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page; DiDi Chengdu taxi dataset—https://outreach.didichuxing.com/app-vue/dataList.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This study was supported by the National Key Research and Development Program of China (no. 2021ZD0112400) and the National Natural Science Foundation of China (no. U1811463).

References

[1] F. H. Tseng, J. H. Hsueh, C. W. Tseng, Y. T. Yang, H. C. Chao, and L. D. Chou, "Congestion prediction with big data for real-time highway traffic," *IEEE Access*, vol. 6, Article ID 57311, 2019.

[2] F. Fan, S. Li, W. Dou, and S. Yu, "An evolutionary approach for short-term traffic flow forecasting service in intelligent transportation system," in *Smart Computing and Communication*, M. Qiu, Ed., vol. 10135, pp. 477–486, SmartCom, Springer, Berlin, Germany, 2017.

[3] S. Xu, R. Zhang, W. Cheng, and J. Xu, "MTLM: A Multi-Task Learning Model for Travel Time Estimation," *GeoInformatica*, Springer, Berlin, Germany, 2020.

[4] J. Kwon, B. Coifman, and P. Bickel, "Day-to-day travel-time trends and travel-time prediction from loop-detector data," *Transportation Research Record*, vol. 1717, pp. 120–129, 2000.

[5] X. Zhang and A. John, "Short-term travel time prediction," *Transportation Research Part C: Emerging Technologies*, vol. 11, no. 3-4, pp. 187–210, 2003.

[6] M. Chen and S. I. J. Chien, "Dynamic freeway travel-time prediction with probe vehicle data," *Transportation Research Record*, vol. 1768, no. 01, pp. 157–161, 2001, https://trrjournalonline.trb.org/doi/pdf/10.3141/1768-19.

[7] H. Ji, A. Xu, X. Sui, and L. Li, "The applied research of kalman in the dynamic travel time prediction," in *Proceedings of the 2010 18th International Conference on Geoinformatics*, Beijing, China, June 2010.

[8] T. Oda, "An algorithm for prediction of travel time using vehicle sensor data," *Computer Science*, vol. 40–44, 1990.

[9] D. J. Sun and Z.-R. Peng, "Route travel time estimation based on seasonal model and Kalman filtering algorithm," *Journal of Chang'an University (Natural Science Edition)*, vol. 441, 2014.

[10] J. Xia, M. Chen, and W. Huang, "A multistep corridor travel-time prediction method using presence-type vehicle detector data," *Journal of Intelligent Transportation Systems*, vol. 15, no. 2, pp. 104–113, 2011.

[11] B. Gupta, S. Awasthi, and R. Gupta, "Taxi travel time prediction using ensemble-based random forest and gradient boosting model," *Advances in Intelligent Systems and Computing*, vol. 63–78, 2018.

[12] J. W. C. Van Lint, S. P. Hoogendoorn, and H. J. van Zuylen, "Accurate freeway travel time prediction with state-space neural networks under missing data," *Transportation Research Part C: Emerging Technologies*, vol. 13, no. 5-6, pp. 347–369, 2005.

[13] D. Park and L. R. Rilett, "Forecasting freeway link travel times with a multilayer feedforward neural network," *Computer-Aided Civil and Infrastructure Engineering*, vol. 14, no. 5, pp. 357–367, 1999.

[14] N. Wisitpongphan, W. Jitsakul, and D. Jieamumporn, "Travel time prediction using multi-layer feed forward artificial neural network," in *Proceedings of the - 2012 4th International Conference on Computational Intelligence, Communication Systems and Networks*, pp. 326–330, Phuket, Thailand, July 2012.

[15] Y. Duan, Y. Lv, and F. Y. Wang, "Travel time prediction with LSTM neural network," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 1053–1058, 2016.

[16] Y. Liu, Y. Wang, X. Yang, and L. Zhang, "Short-term travel time prediction by deep learning: a comparison of different LSTM-DNN models," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 1–8, Yokohama, Japan, October 2017.

[17] D. Wang, J. Zhang, W. Cao, J. Li, and Y. Zheng, "When will you arrive? Estimating travel time based on deep neural networks," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pp. 2500–2507, New York, NY, USA, Febraury 2018.

[18] Y. Sun, Y. Wang, and K. Fu, "FMA-ETA: estimating travel time entirely based on FFN with attention," 2020, https://arxiv.org/abs/2006.04077, Article ID 04077.

[19] J. Ishan and Q. Tony, "A unified neural network approach for estimating travel time and distance for a taxi trip," in ", http://arxiv.org/abs/1710.04350, 2017.

[20] Y. Li, C. Shahabi, and K. Fu, "Multi-task representation learning for travel time estimation," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1695–1704, London UK, August 2018.

[21] Z. Zou, H. Yang, and A. Xing Zhu, "Estimation of travel time based on ensemble method with multi-modality perspective urban big data," *IEEE Access*, vol. 8, no. 2, Article ID 24819, 2020.

[22] Z. Jia, C. Chen, B. Coifman, and P. Varaiya, "The PeMS algorithms for accurate, real-time estimates of g-factors and speeds from single-loop detectors," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 536–541, Oakland, CA, USA, August 2001.

[23] W. H. Lee, S. S. Tseng, and S. H. Tsai, "A knowledge based real-time travel time prediction system for urban network," *Expert Systems with Applications*, vol. 36, no. 3, pp. 4239–4247, 2009.

[24] D. Billings and J. S. Yang, "Application of the ARIMA models to urban roadway travel time prediction - a case study," in *Proceedings of the Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, pp. 2529–2534, Taipei, Taiwan, October 2006.

[25] B. Daniel, T. Olli Pekka, S. Blandin, A. M. Bayen, T. Iwuchukwu, and K. Tracton, "An Ensemble Kalman Filtering approach to highway traffic estimation using GPS enabled mobile devices," in *Proceedings of the IEEE Conference on Decision and Control*, pp. 5062–5068, Cancun, Mexico, December 2008.

[26] C.-H. Wu, J.-M. Ho, and D. T. Lee, "Travel-time prediction with support vector regression," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 4, pp. 276–281, 2004.

[27] Y. Zhang and A. Haghani, "A gradient boosting method to improve travel time prediction," *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 308–324, 2015.

[28] B. Yang, C. Guo, and C. S. Jensen, "Travel cost inference from sparse, spatio temporally correlated time series using Markov models," *Proceedings of the VLDB Endowment*, vol. 6, no. 9, pp. 769–780, 2013.

[29] Y. Jing, "T-drive: driving directions based on taxi trajectories," in *Proceedings of the GIS '10: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 99–108, California, USA, November 2010.

[30] M. Rahmani, E. Jenelius, N. Haris, and Koutsopoulos, "Route travel time estimation using low-frequency floating car data," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, The Hague, Netherlands, October 2013.

[31] M. Rahmani, E. Jenelius, and H. N. Koutsopoulos, "Floating car and camera data fusion for non-parametric route travel time estimation," in *Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 1286–1291, Qingdao, China, October 2014.

[32] X. Zhan, S. Hasan, and V. Satish, "Ukkusuri and Camille Kamga. "Urban link travel time estimation using large-scale

taxi data with partial information"," *Transportation Research Part C: Emerging Technologies*, vol. 33, no. 37–49, 2013.

[33] Y. Wang, Y. Zheng, and Y. Xue, "Travel time estimation of a path using sparse trajectories," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 25–34, Newyork, USA, Auguest 2014.

[34] T. Xu, X. Li, and C. Claramunt, "Trip-oriented travel time prediction (TOTTP) with historical vehicle trajectories," *Frontiers of Earth Science*, vol. 12, no. 2, pp. 253–263, 2018.

[35] T.-Y. Fu and W.-C. Lee, "Deepist: deep image-based spatio-temporal network for travel time estimation," pp. 69–78, 2019.

[36] E. Faruk, "Commercial vehicle travel time estimation in urban networks using GPS data from multiple sources," *Computer Science*, vol. 12, pp. 141–151, 2013.

[37] M. Xu, W. Dai, C. Liu et al., "Spatial-temporal transformer networks for traffic flow forecasting," 2020, https://arxiv.org/abs/2001.02908.

[38] G. Liang, S. Li, Y. Wang, F. Chang, and K. Wu, "Global spatial-temporal graph convolutional network for urban traffic speed prediction," *Applied Sciences*, vol. 10, no. 4, 2020.

[39] Tcl, "NYC taxi and limousine commission (TLC)," https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page, 2021.

[40] M. Abdollahi, T. Khaleghi, and K. Yang, "An integrated feature learning approach using deep learning for travel time prediction," *Expert Systems with Applications*, vol. 139, 2020.

[41] H. Peng, H. Wang, B. Du et al., "Spatial temporal incidence dynamic graph neural networks for traffic flow forecasting," *Information Sciences*, vol. 521, pp. 277–290, 2020.

[42] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," 2017, https://arxiv.org/abs/1706.03762, Article ID 03762.

[43] P. Bholowalia and A. Kumar, "Ebk-means: a clustering technique based on elbow method and k-means in wsn," *International Journal of Computer Application*, vol. 105, no. 9, 2014.

[44] L. Nahar, Z. Sultana, and Z. Sultana, "A new travel time prediction method for intelligent transportation system," *IOSR Journal of Computer Engineering*, vol. 16, no. 3, pp. 24–30, 2014.

[45] Kaggle, "Weather data in New York city - 2016 | kaggle," 2016, https://www.kaggle.com/mathijs/weather-data-in-new-york-city-2016.

[46] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, https://arxiv.org/abs/1412.3555.

[47] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," 2016, https://arxiv.org/abs/1612.01022.

[48] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "Urban traffic prediction from spatio-temporal data using deep meta learning," in *Proceedings of the KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1720–1730, Anchorage AK, USA, July 2019.

[49] N. Ranjan, S. Bhandari, H. P. Zhao, H. Kim, and P. Khan, "City-wide traffic congestion prediction based on cnn, lstm and transpose cnn," *IEEE Access*, vol. 8, Article ID 81606, 2020.

[50] P.-Y. Ting, T. Wada, Y.-L. Chiu, M. T. Sun, K. Sakai, and W. S. Ku, "Freeway travel time prediction using deep hybrid model-taking sun yat-sen freeway as an example," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 8257–8266, 2020.

[51] K. Guolin, M. Qi, T. Finley et al., "LightGBM: a highly efficient gradient boosting decision tree," in *Proceedings of the NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, December 2017.

Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 8030690, 9 pages
https://doi.org/10.1155/2022/8030690

Research Article

Knowledge Graph-Based Enhanced Transformer for Metro Individual Travel Destination Prediction

Hainan Chi,[1,2] **Boyue Wang**,[1,2] **Qibin Ge**,[3] **and Guangyu Huo**[1,2]

[1]*Beijing Key Laboratory of Multimedia and Intelligent Software Technology, Beijing 100124, China*
[2]*Beijing Artificial Intelligence Institute, Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China*
[3]*Beijing Intelligent Transportation Development Center, Beijing 102208, China*

Correspondence should be addressed to Boyue Wang; wby@bjut.edu.cn

Received 4 January 2022; Accepted 28 February 2022; Published 4 April 2022

Academic Editor: Yanming Shen

Accurate and timely destination prediction of subway passengers is of great significance in improving urban residents' travel efficiency, alleviating urban traffic pressure, and recommending the proper location-based service. Although some individual travel destination prediction methods have been proposed, the prediction performance is poor due to the large difference in travel locations of different individuals, the difficulty of evaluating the individual travel intention, the sparsity of individual travel trajectory data, and other problems. To solve these problems, this paper proposes a knowledge graph-based enhanced Transformer method (KG-Trans) for the metro individual travel destination prediction task (MITD-Pre), which contains three main modules: (1) the knowledge graph (KG) module constructs a multilayer individual travel KG from top to bottom, which accurately describes the travel individuals and their travel intentions. By analyzing the association relationship between nodes in the KG, the relationship between travel individuals can be naturally established. The learned similar travel regularity can solve the problem of sparse travel trajectories of some individuals. (2) The enhanced Transformer module extracts the dynamic and hierarchical features from the long-term sequential travel trajectory data. (3) The classifier module introduces the cross-entropy loss to constrain the uniqueness of the predicted subway travel station. The experimental results show that the proposed method obtains a higher destination prediction accuracy than the previous individual travel destination prediction methods.

1. Introduction

As an important part of public transport, the urban rail transit produces a large amount of spatiotemporal trajectory data in real time, which contains rich spatiotemporal location information and reflects the travel mode of passengers. This gives us an opportunity to deeply explore the individual travel patterns and regularity. Traffic prediction is a very basic and important problem in the field of transportation. Most existing traffic prediction methods focus on traffic flow, speed, and so on [1–3]. However, with the development of information technology and various intelligent devices, strong data support and technical support are created for individual travel destination prediction. The real-time prediction of the travel destination of each individual who stays in the subway station is of great significance for the tracking of individuals, service recommendations, and the construction of the smart city. And, it is bound to become an important social demand in the era of big data.

At present, a few methods have been proposed in the field of travel destination prediction, such as the Markov model [4], Bayesian model [5], and Gaussian mixture model [6]. These methods predict the travel destination according to the general mobility characteristics of individuals. However, they ignore the differences in individual behaviors between users [7, 8] and the problems of individual travel data, such as the high spatial complexity and sparse historical travel trajectory. Therefore, their prediction results are unsatisfactory. In addition, considering that there is still a huge challenge in the task of individual travel destination

prediction, that is, how to accurately grasp the individual travel intention, travel intention may be affected by time, location, and other factors. For example, when individuals travel to the same place, their travel intentions may be different on weekdays and weekends. These challenges are beyond the previous methods.

As we all know, KG is a very advanced carrier containing a lot of common-sense knowledge and plays an important role in many practical applications. The emergence of KG provides a new perspective to comprehensively describe individual travel patterns. It carries out much application research; e.g., the entity portrait and the law prediction tasks are achieved by utilizing relationship reasoning and knowledge aggregation. So, KG provides a new method of support for accurately quantifying individual travel patterns in the public transport. It effectively breaks through the traditional expression limitations based on the traffic big data.

Inspired by the KG, this paper integrates deep learning prediction and the KG to achieve accurate travel destination prediction tasks. Specifically, we construct an individual travel KG based on the historical travel data of individuals and then conduct the portrait analysis based on the KG to accurately grasp the individual travel intention. At the same time, for the historical travel trajectory data with the long-term time series and the long-term time dependence, Transformer is used to learn the dynamic and hierarchical characteristics in the sequence data to achieve the final prediction tasks.

This study mainly aims to integrate the KG into the individual travel destination prediction model. The main contributions of this study are summarized as follows:

(i) An individual travel KG is constructed, and we propose a novel individual travel destination prediction method based on such KG, which aims to accurately analyze the individual travel patterns and intentions

(ii) We analyze the travel groups having similar travel trajectories to handle the sparse historical trajectories of some individuals, which is obviously different from the traditional individual travel destination prediction methods

(iii) An enhanced Transformer module is proposed to extract the dynamic and hierarchical features in the history of travel trajectory data

(iv) Experimental results show that the proposed method effectively exploits KG to analyze the subway card data and obtains satisfactory performance

2. Related Work

In this section, we review several related types of research about the KGs and the individual travel destination predictions.

2.1. Knowledge Graph. KG is a very advanced carrier having a lot of common-sense knowledge. And, it acquires success in many practical applications, such as knowledge questions and answers and medical fields. Several general KGs established by Baidu, Google, or other organizations also play an important role in our daily lives. At present, there are many semantic knowledge bases based on Wikipedia in the construction of general domain KG abroad, such as Freebase [9], DBpedia [10], and Yago. The general KGs in China are the bilingual encyclopedic KG XLore [11] developed by Tsinghua University, the Chinese general encyclopedia KG CN-DBpedia developed by Fudan University, and Zhishi.me [12] developed by Shanghai Jiaotong University. The main domain KGs abroad include the film and television domain graph IMDB [13], the music domain graph MusicBrainz [14], and the geographic domain graph GeoNames [15].

With the development of smart transportation, KG research in the transportation field has gradually become more and more popular. Zhou and Chen [16] combine the urban KG with the deep spatiotemporal convolution neural network to solve the problem of traffic congestion. Zeng et al. [17] use the KG to extract the mine relationships between objects, which models the causal relationships of the equipment failures of the railway trains to ensure the operational safety of the high-speed railway. Muppalla et al. [18] use the KG as an abstraction layer to annotate the traffic incidents collected through various methods. Liu et al. [19] learn the urban traffic characteristics extracted from the urban multisource heterogeneous data and construct a KG to mine the urban mobility patterns. Sun et al. [20] semimanually construct a microblog traffic event KG by integrating multiple types of open-source data and use such traffic KG and target detection methods to realize the identification of traffic events in microblogs and solve the traffic problems. Liang et al. [21] use the multilevel planning theory to construct an individual travel KG to accurately identify different types of public transport passengers so as to obtain refined public transport travel characteristics and meet the travel needs of different passengers. Zhang et al. [22] integrate the knowledge of interregional flow, events, and weather to enhance the prediction effects of population inflow and outflow in each region of the city.

As an advanced knowledge carrier, the KG has an extremely important position in the portrayal of individual travel. Therefore, we construct an individual travel KG and use it to solve individual travel destination prediction problems.

2.2. Individual Travel Destination Prediction. With the development of urbanization, people's travel patterns are gradually diversified. Understanding human behaviors and modeling individual travel behaviors are helpful to explain some complex socioeconomic phenomena, which is of great value in location-based services, traffic planning, public safety, and so on. Traditional trajectory prediction methods mostly use machine learning methods, such as the hidden Markov model [4], mixed hidden Markov model [23], Bayesian inference [5], and Gaussian mixture model [6]. Based on the research of public transport smart card data, Zhao et al. [24] predict the individual daily travel capacity, and its travel chain is defined as a set of travel start time,

starting point, and destination. Wang et al. [7] extract a variety of features from the subway card swiping data set to predict the travel destination of passengers entering the subway station but not leaving the station. Li et al. [25] improve the prediction effect of the individual travel through clustering the group travel pattern. Wang et al. [26] design a new movement feature, i.e., a time shift tensor, to consider the user's transformation pattern in the time dimension and propose the attention Markov model. Mo et al. [27] analyze the passenger activity pattern based on the public transport card swiping data in Hong Kong and propose an input-output hidden Markov model to predict the time and location of an individual's next trip at the same time.

In recent years, recurrent neural network (RNN) has obtained the excellent performance in modeling sequence data. Wu et al. [28] propose a new robust location prediction model to consider individual preference and social interaction, which alleviates the impact of randomness of location movement and improves the prediction performance. De Brebisson et al. [8] predict the taxi destination by using a multilayer perceptron and a two-way cyclic neural network. Lv et al. [29] regard the trajectory as a two-dimensional image to model the trajectory from different perspectives and apply Convolutional Neural Networks (CNNs) to extract multiscale two-dimensional trajectory features for the accurate destination prediction. Zhang et al. [30] apply Surprisal-Driven Zoneout (SDZ) to RNN, which improves the robustness of the destination prediction model and reduces the training time. Based on the Long Short-Term Memory (LSTM) model, Li et al. [31] combine the extracted depth spatiotemporal features with the original features to predict the taxi destination. Xu et al. [32] use an adaptive attention network to model different extraction features of locations and implement the time gate and the distance gate into LSTM to capture the spatiotemporal relationship between continuous locations.

Although some individual travel destination prediction methods have been proposed, some common problems still seriously affect the prediction effect; for example, it is difficult to grasp the travel intentions of different individuals and handle the sparseness of historical trajectory data of some individuals.

3. Construction of Individual Travel KG

In this section, firstly, we preprocess the subway card swiping data set to clean out the dirty data, the duplicate redundant data, and so on. Then, the individual travel KG is constructed and displayed visually.

3.1. Data Preprocessing. The original data set adopts the passenger card swiping records collected by the Beijing Metro automatic toll collection system in July, August, September, November, and December 2015. Each record contains 11 items, including card ID, subway route, subway station, date, card type, and transaction type. The card ID is the unique identifier of the intelligent transportation card, which is used to identify a unique passenger.

Due to the large volume, the dirty data interference, the missing key items, the coupling of travel records, and other problems in the data set, the card swiping records of each travel individual are scattered in the large data set, which makes it difficult to form a complete travel chain and brings great obstacles in mining the passenger travel patterns. Therefore, the original data must be preprocessed to extract the complete travel chain, so as to build a good data foundation for the construction of individual travel KGs, travel law mining, and travel destination prediction.

First, we remove the repeated card swiping records in the original subway data set and the records with the same card swiping time and the same card swiping station. Moreover, because the traffic card type and other information contribute less to the subsequent destination prediction task, we also filter this information to avoid the interference of redundant information. So, we retain 7 necessary items, including card ID, boarding line, boarding station, alighting line, alighting station, boarding time, and alighting time. In addition, since the research objects are passengers taking the subway as their normal transportation tool, we also remove the passengers whose records are less than the average of 30 card swiping records per month and whose data volume is abnormal. After these operations, 150–400 travel records in five months for each passenger are retained.

3.2. KG Construction. Accurately grasping the individual's travel intention is the major challenge in the task of the individual travel destination prediction, and such travel intention is affected by many factors, for example, time and station. To achieve this purpose, we intend to construct the individual travel KG to accurately analyze the travel individuals and grasp the corresponding travel intention. Therefore, we construct the individual travel KG using the passengers' travel location, time, date, and so on.

In this knowledge, there are five types of entities: card ID, date, travel date attribute (whether working day or not), route, and subway station and times. The corresponding relationships are divided into five categories as shown in Table 1.

We analyzed the travel data of 8000 individuals in five months. Specifically, after data preprocessing, we chose all the historical travel records of 8000 individuals from tens of millions of people. Then, we extracted the travel record knowledge, and the steps are shown in Algorithm 1.

3.3. Visualization of KG. Through the above knowledge extraction steps, we get the structured data and then visually display the graph by neo4j[1]. Taking the travel record of Card No. 787931 in July as an example, our constructed graph is shown in Figure 1.

4. Methodology

In this section, we propose a knowledge graph-based enhanced Transformer for the MITD-Pre, namely, KG-Trans. The model framework is shown in Figure 2. Firstly, we identify and extract the travel individuals with similar routes

Table 1: Entities and relationships.

Entities	Relationships
Card ID-date	Travel date
Card ID-frequency	Frequency of travel
Date-attributes of travel date	Belong to
Attributes of travel date-route	Travel route
Route-subway station	Boarding/alighting station

Step 1: get the frequency of travel. The frequency of travel of an individual (ID) is the total number of travel records of the ID within a specified time frame in the data set.
Step 2: get the travel date. In the individual travel card swiping record, the travel time is expressed as "year/month/day: hour: minute." We only take the date information and ignore the hour information.
Step 3: get the date attribute. For the travel date obtained in Step 2, we check the calendar to determine whether it is a working day. The working day is marked as 1, and the nonworking day is marked as 0.
Step 4: get the origin-destination (OD) records. The storage format of one complete travel record is "ID, boarding time, boarding route, boarding station, boarding time, alighting route, and alighting station." Then, we extract the travel OD as "boarding line * boarding station - alighting line * alighting station."
Step 5: get to the subway station. As shown in Step 4, the subway station is expressed in the form of "line * station."

Algorithm 1: Knowledge extraction.

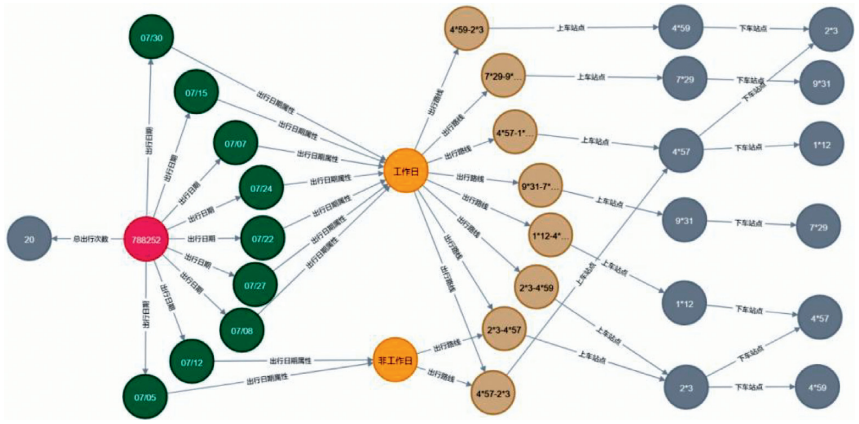


Figure 1: Graph display of July travel records of passenger 788252. The front gray node represents the total number of trips per month; the pink nodes represent the individual card ID; the green nodes represent the travel dates; the ginger yellow nodes represent the travel routes; and the behind gray nodes represent the boarding stations and the get-off stations, respectively.

based on the constructed individual travel KG. Then, the same class of data is sent to the Transformer to train the model. Finally, the prediction results are compared to the real values stored in the KG to construct the loss function.

4.1. Relationship Analysis between Travel Individuals Based on KG. Due to the large behavioral differences between individuals and the sparseness of the travel records of some individuals, the accuracy of individual travel destination prediction is seriously affected. In view of this phenomenon, we improve the effectiveness of the individual travel destination prediction through the "group effect." As for the group here, we define it as "people with similar routes." For example, for one group of commuters that live in Xierqi, Beijing, and work in Zhongguancun, Beijing, the subway routes on weekdays are very similar. This paper discovers the individuals having similar routes through analyzing the KG.

KG has many nodes and edges which contain complex information and rich semantics. We aim to find the correlation between nodes in the graph so that we can infer the correlation between the individuals. For example, there are many shared nodes between some passenger routes. As shown in Figure 3(a), we can judge that the two passenger routes are very similar. Some passenger travel routes have few or no shared nodes. As shown in Figure 3(b), it is considered that the similarity between the two passenger routes is relatively weak. In this paper, after analyzing the five-month travel data of 8000 individuals in detail and considering the impacts of classification accuracy on the prediction effect, we constrain the members in the same category satisfy the following rules:

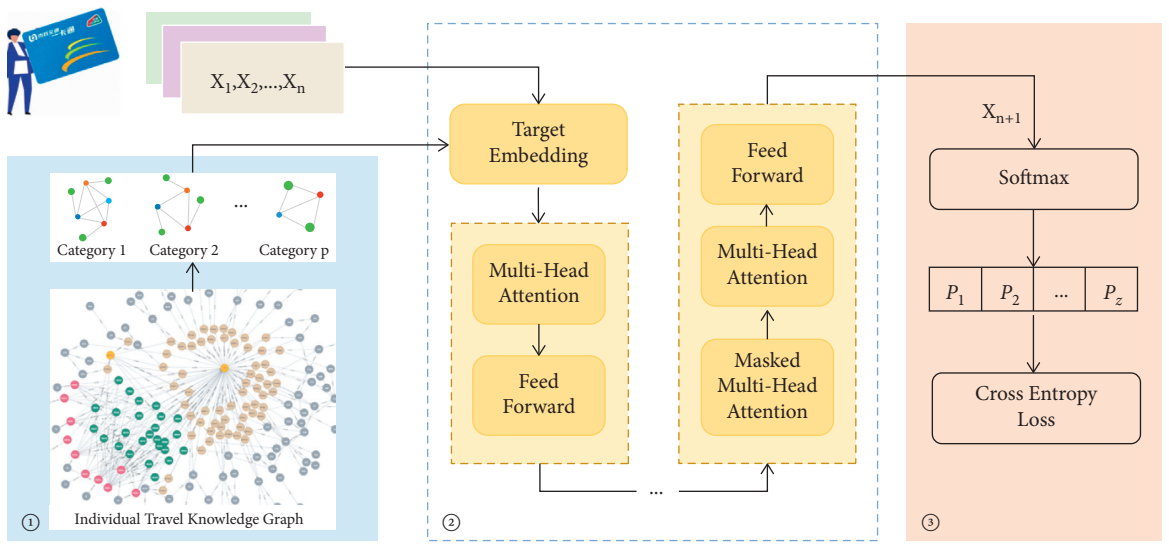


Figure 2: KG-Trans model frame diagram. ① The knowledge graph module constructs a multilayer individual travel KG from top to bottom. ②The enhanced Transformer module extracts the dynamic and hierarchical features from the long-term sequential travel trajectory data. ③The classifier module introduces the cross-entropy loss to constrain the uniqueness of the predicted subway travel station.

(i) For members in the same category, the difference in travel frequency recorded by card swiping should be less than 50

(ii) For members in the same category, they should have more than 70% similar routes between the subway stations

Then, on the constructed graph, we first divide the travel records of 8000 people into five length intervals, 150–200, 200–250, 250–300, 300–350, and 350–400 (named group 1 to group 5), according to the travel chain length interval. The node similarity between the members of each length interval is calculated according to the definition of route similarity, and finally, we obtain p classes of travel groups ($p \geq 5$). In these p travel groups, the members of each group are "route similar members" to each other. In this way, by putting each class of members, we classified them into the model for training and we can solve the problem of sparse historical travel routes of some travel individuals according to their similarity.

4.2. Transformer-Based Individual Travel Destination Prediction. The Transformer is an effective method to process the sequence data. Its multihead attention mechanism and stacking layer learn the dynamic and hierarchical characteristics of the sequence data. Therefore, Transformer can predict the traffic flow with the long-term time series and long-term time dependence very well. Considering that the card swiping data of subway passengers have the properties of long-term time series and long-term time dependence, Transformer is naturally selected as an important module in the individual travel prediction model in this paper.

The basic structure of the Transformer used in our model is shown in Figure 2. Its core module is the multihead attention layer. Firstly, the input of the model is the card swiping records of passengers with similar routes, i.e., $\mathscr{R} = \{\mathscr{X}_1, \mathscr{X}_2, ..., \mathscr{X}_m\}$, where m denotes the number of passengers and $\mathscr{X}_i = \{x_0, x_1, \cdots, x_n\}$ represents the historical trajectory sequence of the i-th passenger.

The multihead attention (MH) layer adopts different linear mappings to project the input sequence elements to the query, key, and value, i.e., a tuple (Q, K, and V). The output is the corresponding weighted sum of values. The weight assigned to each value is calculated through the compatible functions of the query and the corresponding key. The application of attention can be expressed as

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{1}$$

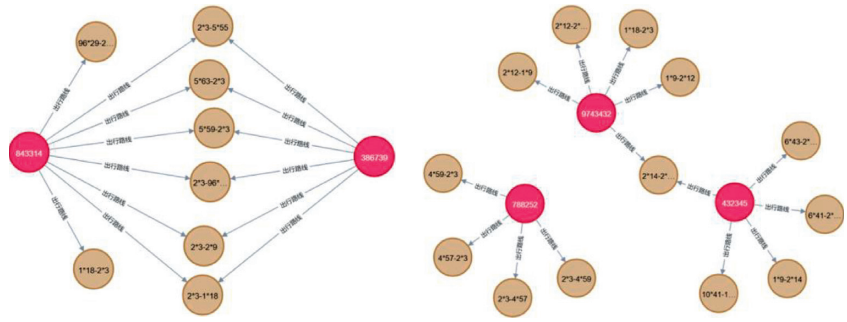
where Attention () calculates the attention of the input data and softmax () is an activation function. To establish a single-head attention module, each node can have three subspaces, namely, the queries subspace $Q \in \mathbb{R}^{N \times d_k}$, key subspace $K \in \mathbb{R}^{N \times d_k}$, and value subspace $V \in \mathbb{R}^{N \times d_k}$, where d_k is the dimension of queries, keys, and values.

In the global encoder, the input features are projected into the high-dimensional subspace and the learnable mapping is realized through the feedforward neural network, which can be expressed as

$$\begin{cases} Q = XW_q \\ K = XW_k \\ V = XW_v \end{cases}, \tag{2}$$

where X is the input feature. W_q, W_k, and W_v are the learnable parameters.

The multihead attention network uses h feedforward neural networks to linearly project Q, K, and V, which achieves a multihead mechanism. In this case, the model can pay much attention to the information of different representation subspaces from different stations, which can be expressed as



(a) (b)

Figure 3: Route similarity among different travel individuals: (a) the route similarity is strong; (b) the route similarity is weak. The ginger yellow nodes represent the travel routes, and the pink nodes represent the individual card IDs.

$$\mathrm{MH}(\mathrm{Q},\mathrm{K},\mathrm{V}) = \mathrm{Concat}(\mathrm{head}_1,\ldots,\mathrm{head}_\mathrm{h})\mathrm{W}^\mathrm{o}, \tag{3}$$

where MH () calculates the multihead attention of the input data, Concat () concatenates the input data, and the i-th head $head_i$ can be expressed as

$$head_i = Attention\left(QW_i^Q, KW_i^K, VW_i^v\right), \tag{4}$$

where the mapping matrices are $\mathrm{W}_i^\mathrm{Q} \in \mathbb{R}^{\mathrm{d}_{\mathrm{model}}\times\mathrm{d}_\mathrm{k}}$, $\mathrm{W}_i^\mathrm{K} \in \mathbb{R}^{\mathrm{d}_{\mathrm{model}}\times\mathrm{d}_\mathrm{k}}$, $\mathrm{W}_i^\mathrm{Q} \in \mathbb{R}^{\mathrm{d}_{\mathrm{model}}\times\mathrm{d}_\mathrm{v}}$, and $\mathrm{W}_i^\mathrm{o} \in \mathbb{R}^{\mathrm{hd}_\mathrm{v}\times\mathrm{d}_{\mathrm{model}}}$, respectively. And, $\mathrm{d}_{\mathrm{model}}$ is the dimension of keys, values, and queries of single-head attention, $\mathrm{d}_\mathrm{k} = \mathrm{d}_\mathrm{v} = \mathrm{d}_{\mathrm{model}}/\mathrm{h}$.

The final prediction result of our model is

$$\mathrm{x}_{\mathrm{n}+1} = \mathrm{MH}(\mathrm{Q},\mathrm{K},\mathrm{V}), \tag{5}$$

where $\mathrm{x}_{\mathrm{n}+1}$ is the travel destination of an individual.

Finally, we use the softmax layer to give each station a probabilityP:

$$\mathrm{P}(\mathrm{x}_i) = \frac{\mathrm{e}^{\mathrm{x}_i}}{\sum_\mathrm{z}\mathrm{e}^{\mathrm{x}_\mathrm{z}}}, \tag{6}$$

where x_i is the feature representation of i-th station and z is the total number of stations. The station with the largest probability is the final prediction result of the proposed model.

4.3. Loss Function. The individual travel destination prediction problem is different from the previous OD prediction and taxi destination prediction problems. The performance of OD or taxi destination prediction tasks is evaluated by measuring the error of the predicted longitude and latitude. The individual destination prediction results are estimated by only correct (1) or incorrect (0). Therefore, the cross-entropy loss function was selected as the loss function in this paper.

Through the current card swiping records of a travel individual, we want to predict the travel destination of the individual. This problem can be regarded as a probability problem in which we give a probability to each station and the station with the largest probability is the destination prediction result. Cross-entropy mainly reflects the distance between the actual output (probability) and the expected output (probability). Furthermore, the smaller the value of cross-entropy, the closer the two probability distributions are. Assuming that the probability distribution p is the expected output, the probability distribution q is the actual output, and the cross-entropy H (p, q) measures the distance between p and q, we have

$$\mathrm{H}(\mathrm{p},\mathrm{q}) = -\sum_{\mathrm{i}=1}^{\mathrm{n}}\left(\mathrm{p}(\mathrm{x}_\mathrm{i})\log\mathrm{q}(\mathrm{x}_\mathrm{i}) + (1-\mathrm{p}(\mathrm{x}_\mathrm{i}))\log(1-\mathrm{q}(\mathrm{x}_\mathrm{i}))\right). \tag{7}$$

During training the deep learning network, given the input data and labels, the real probability distribution p (x) is determined. Therefore, the formula of the cross-entropy commonly used in deep learning is formulated as follows:

$$\mathrm{H}(\mathrm{p},\mathrm{q}) = -\sum_{\mathrm{i}=1}^{\mathrm{n}}\mathrm{p}(\mathrm{x}_\mathrm{i})\log(\mathrm{q}(\mathrm{x}_\mathrm{i})). \tag{8}$$

5. Experimental Results

5.1. Experiment Settings. In this paper, the proposed destination prediction model is compared with the five traditional prediction methods including Markov, LSTM, GRU, CNN, and FNN.

Markov [4]: the Markov model is a statistical analysis model, which is widely used in speech recognition, automatic part of speech tagging, sequence classification, sequence prediction, and other applications.

LSTM [33]: Long Short-Term Memory (LSTM) can perform better in longer sequences, which is a special version of RNN. The problems of gradient disappearance and gradient explosion in the process of long sequence training are solved.

GRU [34]: the Gated Recurrent Unit (GRU) is an effective variant of the above LSTM network. It has a simpler structure and better effects than the above LSTM network and still solves the long-term dependency problem in RNN. Therefore, it is an important manifold network at present.

CNN [35]: The Convolutional Neural Network (CNN) is a feedforward neural network. For some sequence

processing problems, the effect of the one-dimensional Convolutional Neural Network is comparable to that of the RNN, while the computational cost is usually much lower.

FNN [36]: the feedforward neural network (FNN), also known as a multilayer perceptron (MLP), contains multiple fully connected hidden layers. The complex mapping from input space to output space is realized by aggregating multiple simple nonlinear functions.

We conducted the whole experiments on a GPU workstation, containing a 2080TI GPU with 11G memory. During the training process, the number of epochs is set to 200; the batch size is set to 32; the learning rate is set to 0.001; the number of self-attention heads in the Transformer is set to 8; the dimension of the input vector is set to 6; and the dropout rate is set to 0.3. All experiments use 3 historical observations to predict the next point in this paper.

5.2. Evaluating Indicator. To fairly estimate the experiment performance, we refer to the evaluation indicator Hit@n that is exploited in the KG representation learning TransE [37]. During the testing procedure, the prediction results are arranged from large to small. It determines whether the first n results contain the correct options. If it contains the correct options, we increase the hit value by 1; otherwise, it processes to the next cycle. In other words, we do not require the first one value to be right (Hit@1 except), as long as there is the correct result in the first n results. The final accuracy ACC is calculated as follows:

$$\mathrm{ACC} = \frac{\mathrm{hit}}{\mathrm{hit} + \mathrm{FP}}, \tag{9}$$

where hit is the number of travel stations that are predicted correctly and FP is the number of incorrectly predicted results.

5.3. Prediction Performance of KG-Trans. Table 2 shows the destination prediction results of above five compared methods on the card swiping data set of Beijing Metro. The data set is described in Section 3.1 in detail. It can be seen that our KG-Trans is obviously superior to other methods in all indicators. Group 1–Group 5 in the experimental data set are distributed from less to more according to the number of travel records of one individual (the detailed description is in Section 4.1). The experimental results show that the prediction effect of KG-Trans is more accurate as the number of travel records increases.

For the destination prediction problem solved in this paper, our data set has a long travel sequence of passengers. Although LSTM can mine the long-term sequence characteristics hidden in the data, the LSTM gradient will disappear and affect the prediction results when the sequence length exceeds a certain limit. Moreover, the passenger card swiping data are the sequence data, but the time intervals between records in the sequence do not have the obvious regularity. For the feature learning of such data, LSTM still has difficulty to capture the time correlation.

In Table 2, we can see that the prediction effect of GRU is the worst. We believe the reason is that GRU ignores the middle layer compared to LSTM. Although this operation reduces parameters and prevents over-fitting risk, the middle layer plays a key role in the long-term sequence feature extraction. Therefore, the GRU effect receives the worst performance in our passenger travel prediction scenario.

Table 2: Experimental results.

		Group 1	Group 2	Group 3	Group 4	Group 5
Hit@3	LSTM	0.383	0.386	0.403	0.490	0.404
	GRU	0.159	0.210	0.128	0.169	0.128
	CNN	0.328	0.578	0.317	0.481	0.469
	FNN	0.547	0.328	0.640	0.688	0.779
	Markov	0.386	0.466	0.489	0.550	0.598
	KG-trans	**0.572**	**0.583**	**0.689**	**0.731**	**0.790**
Hit@5	LSTM	0.585	0.538	0.416	0.750	0.689
	GRU	0.309	0.286	0.203	0.258	0.148
	CNN	0.372	0.588	0.426	0.492	0.476
	FNN	0.602	0.464	0.758	0.762	0.794
	Markov	0.427	0.522	0.527	0.597	0.640
	KG-trans	**0.611**	**0.760**	**0.767**	**0.809**	**0.810**
Hit@10	LSTM	0.633	0.605	0.470	0.755	0.796
	GRU	0.618	0.433	0.316	0.456	0.160
	CNN	0.450	0.608	0.433	0.875	0.487
	FNN	0.672	0.516	0.773	0.883	0.805
	Markov	0.486	0.589	0.600	0.659	0.690
	KG-trans	**0.733**	**0.775**	**0.792**	**0.947**	**0.929**

The prediction effect of CNN is similar to that of LSTM. The receptive field of CNN has a fixed size, and the processing effect of the local information is very prominent. It can capture some local specific features, while the capture ability of global features is worse than LSTM.

In addition to our model, FNN has the best experimental effects. FNN is a fully connected network, and the computational complexity is very large, which results in low training efficiency. In addition, FNN can only learn the high-order combined features while the low-order features are not modeled in the model.

Since the prediction effect of Markov heavily depends on the data of the previous time, the performance of the characteristic learning of long-term time series is poor, so the prediction effect of individual travel destinations is not good.

Compared with the above methods, with the help of the accurate portrait analysis of passengers in our individual travel KG, we can identify and extract the route similar passengers and put them into different in-depth learning models for training. Such KG solves the problem of the sparse individual travel trajectory by exploiting the characteristics of similar passenger routes. In addition, the Transformer has the multihead self-attention mechanism and the stacking layer, which has stronger structural flexibility and captures a wider range of time correlation. The Transformer achieves a very good prediction effect for the individual travel data series with the long-term time series and the long-term time dependence.

6. Conclusion

In this paper, a knowledge graph-based enhanced Transformer for the metro individual travel destination prediction method is proposed. The method of constructing individual travel KG is used to accurately analyze the travel individuals. And then, the Transformer's outstanding sequential learning ability is used to capture the sequence information in the individual travel chain. The test results on the AFC card swiping data set of Beijing Metro show that our method can well learn the regularity and characteristics of passenger travel records, which is greatly improved compared with previous studies. In future work, our model should also consider the influence of more external factors (the weather, the road network, and traffic events) and build more levels of traffic KG to improve the prediction accuracy. Moreover, in terms of individual travel destination prediction, in addition to the next travel location, the next travel time will also be the focus of our next research.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The research project was partially supported by the National Natural Science Foundation of China under Grant nos. 62072015, U19B2039, and U1811463 and National Key R&D Program of China under Grant no. 2018YFB1600903.

References

[1] X. L. Wei, Y. Zhang, Y. Wei et al., "Metro passenger-flow representation via dynamic mode decomposition and its application," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[2] J. Wang, Y. Zhang, Y. Wei, Y. Hu, X. Piao, and B. Yin, "Metro passenger flow prediction via dynamic hypergraph convolution networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 12, pp. 7891–7903, 2021.

[3] Y. Wang, Y. Zhang, X. Piao, H. Liu, and K. Zhang, "Traffic data reconstruction via adaptive spatial-temporal correlations," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1531–1543, 2018.

[4] D. Lian, X. Xie, V. W. Zheng, N. J. Yuan, F. Zhang, and E. Chen, "Cepr: a collaborative exploration and periodically returning model for location prediction," *ACM Transactions on Intelligent Systems and Technology*, vol. 6, no. 1, pp. 1–27, 2015.

[5] J. Krumm and E. Horvitz, "Predestination: inferring destinations from partial trajectories," in *Ubiquitous Computing, 8th International Conference*, Berlin, Germany, 2006.

[6] J. Wiest, M. Hoffken, U. Kreßel, and K. Dietmayer, "Probabilistic trajectory prediction with Gaussian mixture models," in *Intelligent Vehicles Symposium (IV)*, pp. 141–146, Madrid, Spain, 3-7 June 2012.

[7] H. Wang, J. Zhao, K. Ye et al., "A destination prediction model for individual passengers in urban rail transit," *International Conference on High Performance Big Data and Intelligent Systems*, pp. 1–6, 2020.

[8] A. De Brébisson, É. Simon, A. Auvolat, P. Vincent, and Y. Bengio, "Artificial neural networks applied to taxi destination prediction," *Computer Science*, 2015.

[9] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pp. 1247–1250, New York, NY, USA, 2018.

[10] A. Sören, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "DBpedia: A nucleus for a web of open data," *Semantic Web*, vol. 4825, pp. 722–735, 2007.

[11] Y. Wang, Y. Jia, D. Liu, X. Jin, and X. Cheng, "Open web knowledge aided information search and data mining," *Journal of Computer Research and Development*, vol. 52, no. 02, pp. 456–474, 2015.

[12] Y. Ma and G. Qi, "An analysis of data quality in DBpedia and Zhishi.me," *Linked Data and Knowledge Graph*, vol. 406, pp. 106–117, 2013.

[13] W. Peng, S. Li, G. Sun et al., "RC-NVM: Enabling symmetric row and column memory accesses for in-memory databases," in *IEEE International Symposium on High Performance Computer Architecture*, pp. 518–530, Vienna, Austria, 24-28 Feb. 2018.

[14] A. Swartz, "MusicBrainz: A semantic web service," *IEEE Intelligent Systems*, vol. 17, no. 1, pp. 76-77, 2002.

[15] B. Regalia, K. Janowicz, G. Mai, D. Varanka, and E. L. Usery, "GNIS-LD: Serving and visualizing the geographic names information system gazetteer as linked data," in *European Semantic Web Conference*, Cham, Switzerland, 2018.

[16] G. L. Zhou and F. Chen, "Urban congestion areas prediction by combining knowledge graph and deep spatio-temporal convolutional neural network," in *2019 4th International Conference on Electromechanical Control Technology and Transportation*, pp. 105–108, Guilin, China, 26-28 April 2019.

[17] Y. Zeng, Y. Qin, D. Liu, Y. Fu, M. Gong, and X. Zhang, "Railway train device fault causality model based on knowledge graph," in *2020 International Conference on Sensing, Diagnostics, Prognostics, and Control*, pp. 385–390, Beijing, China, 5-7 Aug. 2020.

[18] R. T. Muppalla, S. Lalithsena, T. Banerjee, and A. Sheth, "A knowledge graph framework for detecting traffic events using stationary Cameras.WebSci '17," in *ACM Web Science Conference*, pp. 431–436, New York, NY, USA, 2017.

[19] J. Liu, T. Li, S. Ji et al., "Urban flow pattern mining based on multi-source heterogeneous data fusion and knowledge graph embedding," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[20] X. R. Sun, Y. Meng, and W. L. Wang, "Identifying traffic events from weibo with knowledge graph and target detection," *Data Analysis and Knowledge Discovery*, vol. 48, no. 12, pp. 140–151, 2020.

[21] Q. Liang, J. C. Weng, and P. F. Lin, "Public transport commuter identification based on individual travel graph," *Journal of Transportation Systems Engineering and Information Technology*, vol. 18, no. 002, pp. 100–107, 2018.

[22] J. Zhang, Z. Yu, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," *AAAI*, vol. 31, 2016.

[23] J. Ye, Z. Zhu, and H. Cheng, "What is your next move: User activity prediction in location-based social networks," in

Proceedings of the SIAM International Conference on Data Mining, Austin, TX, USA, 2013.

[24] Z. Zhao, H. N. Koutsopoulos, and J. Zhao, "Individual mobility prediction using transit smart card data," *Transportation Research Part C: Emerging Technologies*, vol. 89, no. APR, pp. 19–34, 2018.

[25] F. Li, Q. Li, Z. Li, Z. Huang, X. Chang, and J. Xia, "A personal location prediction method based on individual trajectory and group trajectory," *IEEE Access*, vol. 7, pp. 92850–92860, 2019.

[26] H. Wang, Y. Li, D. Jin, and Z. Han, "Attentional Markov model for human mobility prediction," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, pp. 2213–2225, 2021.

[27] B. Mo, Z. Zhao, H. N. Koutsopoulos, and J. Zhao, "Individual mobility prediction in mass transit systems using smart card data: an interpretable Activity-based hidden Markov approach," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[28] R. Wu, G. Luo, Q. Yang, and J. Shao, "Learning individual moving preference and social interaction for location prediction," *IEEE Access*, vol. 6, pp. 10675–10687, 2018.

[29] J. M. Lv, Q. Li, Q. Sun, and X. Wang, "T-CONV: A convolutional neural network for multi-scale taxi trajectory prediction," *Computer Vision and Pattern Recognition*, pp. 82–89, 2016.

[30] G. Zhang, Y. Li, L. Zhang, Q. Fan, and X. Li, "Taxi travel destination prediction based on SDZ-RNN," *Computer Engineering and Applications*, vol. 54, no. 6, pp. 143–149, 2018.

[31] Y. Li, S. Cui, L. Zhang, B. Liu, and D. Song, "Taxi destination prediction with deep spatial-temporal features," in *International Conference on Communications, Information System and Computer Engineering*, pp. 562–565, Beijing, China, 14-16 May 2021.

[32] J. Xu, J. Zhao, R. Zhou, C. Liu, P. Zhao, and L. Zhao, "Predicting destinations by a deep learning based approach," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 2, pp. 651–666, 2021.

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

[34] K. Cho, B. V. Merrienboer, C. Gulcehre et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *Computer Science*, 2014.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[36] G. Bebis and M. Georgiopoulos, "Feed-forward neural networks," *IEEE Potentials*, vol. 13, no. 4, pp. 27–31, 1994.

[37] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, *Translating Embeddings for Modeling Multi-Relational Data*, Curran Associates Inc, NY, United States, 2013.

Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 7580267, 12 pages
https://doi.org/10.1155/2022/7580267

Research Article

Prediction of Train Station Delay Based on Multiattention Graph Convolution Network

Dalin Zhang,[1] **Yi Xu**,[1] **Yunjuan Peng**,[1] **Yumei Zhang**,[2] **Daohua Wu**,[2] **Hongwei Wang**,[2] **Jintao Liu**,[2] **Sabah Mohammed**,[3] **and Alessandro Calvi**[4]

[1]*School of Software Engineering, Beijing Jiaotong University, Beijing 100044, China*
[2]*National Research Center of Railway Safety Assessment, Beijing Jiaotong University, Beijing 100044, China*
[3]*Department of Computer Science, Lakehead University, Thunder Bay P7A0A2, Canada*
[4]*Department of Engineering, Roma Tre University, Rome 00118, Italy*

Correspondence should be addressed to Dalin Zhang; dalin@bjtu.edu.cn and Yi Xu; 21126357@bjtu.edu.cn

Received 27 October 2021; Revised 8 January 2022; Accepted 26 January 2022; Published 21 February 2022

Academic Editor: Yong Zhang

Train station delay prediction is always one of the core research issues in high-speed railway dispatching. Reliable prediction of station delay can help dispatchers to accurately estimate the train operation status and make reasonable dispatching decisions to improve the operation and service quality of rail transit. The delay of one station is affected by many factors, such as spatiotemporal factor, speed limitation or suspension caused by strong wind or bad weather, and high passenger flow caused by major holiday. But previous studies have not fully combined the spatiotemporal characteristics of station delay and the impact of external factors. This paper makes good use of the train operation data, proposes the multiattention mechanism to capture the spatiotemporal characteristics of train operation data and process the external factors, and establishes a Multiattention Train Station Delay Graph Convolution Network (MATGCN) model to predict the train delay at high-speed railway stations, so as to provide references for train dispatching and emergency plan. This paper uses real train operation data coming from China high-speed railway network to prove that our model is superior to ANN, SVR, LSTM, RF, and TSTGCN models in the prediction effect of MAE, RMSE, and MAPE.

1. Introduction

High-speed rail transit will be affected by many factors such as stations, lines, and equipment [1]. Train delay will cause long time of passenger detention and bring inconvenience. In addition, with the increase of lines and the decrease of train tracking interval, the delay of one train may affect the other trains and form a knock-on effect. Train delay has always been one of the core research problems in high-speed railway dispatching [2]. Reliable prediction of station delay can help dispatchers to accurately estimate the train operation status and make reasonable dispatching decisions to improve the operation and service quality of rail transit.

Out of the above consideration, this paper aims to dig out the hidden train operation law in the actual operation data based on the previous research, that is, on the basis of the actual operation data, comprehensively consider the dual propagation characteristics of time and space of train operation delays and external factors such as weather, wind level, and major holiday to predict train station delays. This paper uses statistical analysis to observe whether the weather, wind speed, and major holiday have an impact on train delay and comprehensively considers the impact of spatiotemporal characteristics and external factors on train delay to predict the delay of some stations in a certain period of time.

The train delay prediction of high-speed railway stations is a typical spatiotemporal network prediction problem [3, 4]. In the analysis of train delay, it is necessary to comprehensively consider the spatiotemporal dependence between multiple trains and multiple lines [5]. The adjacent stations are spatially related and the timestamps are related

in time [6]. So, the train delay data has the characteristics of spatial dependence, temporal relevance, and spatiotemporal correlation.

In addition to spatiotemporal factors, the operation of one train is also affected by many external factors [7]. For example, in rainy, snowy, and foggy weather, the operation speed of one train is limited, which may lead to delay, and, in extremely bad weather, trains may even be suspended. In addition, passenger flow is also a major influencing factor. During major holiday, a substantial increase in passenger flow will affect the trains' stop time. Through the above analysis, we find that the train operation analysis needs to consider not only spatiotemporal factors but also relevant external factors. In this paper, the factors we choose are wind level, temperature, weather conditions, and whether it is a major holiday.

The single-train delay refers to the delay of a specific train at each station; this paper does not predict the delays of one specific train, because if one train is delayed, the specific dispatching decision is issued by the railway dispatching department, which depends on the experience and knowledge of the dispatchers. On the contrary, we vaguely predict the number of train delays in each time period for each station. The main difference between the single-train delay and station delay is whether to pay attention to the delay of a train or the total number of delayed trains in a station over a period of time.

At present, there are many SOAT models in the field of traffic prediction, but most of the predictions of flow and speed are concentrated on the highway network, such as DCRNN [8] and its derived models.

It is difficult for us to directly apply these models to the prediction of train station delay; the reasons are as follows:

(1) At present, we cannot obtain such close train operation data in time and space similar to the highway network.

(2) All kinds of vehicles running on the highway have no fixed speed and direction, and the train needs to travel in strict accordance with the minimum and maximum speed limit and line on the train diagram. Many traffic prediction SOAT models are based on random walk, so they cannot be directly applied to train delay prediction.

(3) Traffic predictions are often concentrated on several roads or within a city. But this paper uses a large dataset, including most high-speed rail stations and lines in China. Its research scope runs through China, and almost no highway prediction work is established in such a large range. This problem brings us more difficulties, such as the extraction range of node features, the capture of spatiotemporal characteristics, the different train operation laws between different regions and different lines, and the test of the robustness of the model.

There are many works on the analysis and prediction of train delay in high-speed railway. For example, Liu et al. [9] used statistical methods to study the actual operation data of the two stations of Beijing-Shanghai railway lines and calculated the delay rate of the station; Milinković and Marković [10] proposed a fuzzy Petri net (FPN) model to simulate the traffic process and train operation in the railway system to estimate train delays; Marković and Milinković [11] analyzed the relationship between passengers and various characteristics of the railway system in train arrival delays and applied the support vector machine model to make train delay analysis; Lessan et al. [12] built a train delay prediction model based on Bayesian network. Our work is an improvement of the paper of Zhang et al. [13]; compared with that paper, we proposed the multiattention mechanism to achieve more accurate prediction, and we will introduce the differences in Section 3.3. Most of these works have some similar characteristics: (1) The research on train operation data mostly stays in the stage of statistical analysis but fails to tap the hidden train operation law in it. (2) It is rare to consider the spatiotemporal attributes of trains. The temporal impact caused by delay is obvious, but the spatial impact of different lines in some hub stations is often ignored. (3) Almost no research considers the comprehensive impact of spatiotemporal characteristics and external factors.

Compared with existing works, the contributions of this paper can be summarized as follows:

(1) We define the train operation network as a graph and the stations on the network as nodes and add node features. We define the lines connecting stations as edges and the reciprocal of the distance between adjacent stations as the weight of edges, indicating the mutual influence between adjacent stations.

(2) We propose a MATGCN model based on multiattention mechanism to predict the total number of train delays at one certain station in a certain period of time; this mechanism makes MATGCN able to adjust the parameters during training according to the importance of different attributes, so as to have better robustness.

(3) We spent a lot of time building a high-speed rail delay dataset and published it on Figshare [14]; this dataset contains the train operation data from October 8, 2019, to January 27, 2020, and the train delay data of the railway stations passing by these trains. Weather, temperature, wind power, and major holidays are considered as factors affecting train operation. As we know, this is the first public large-scale high-speed rail delay dataset.

(4) In the contrast experiment, we use real-world data and make predictions for 1 to 6 hours. The result shows that our MATGCN model can well capture the periodic law of train operation and maintain good accuracy in long-term prediction.

The following parts of this paper are organized as follows: Section 2 systematically investigates the existing train delay prediction and spatiotemporal data mining methods.

Section 3 shows the materials and methods. Section 4 shows the results of the experiment and Section 5 summarizes the work of this paper.

2. Literature Review

Some achievements have been made in the prediction of train delay previously. Generally, it can be divided into the following categories: (1) works based on scenario calculation and simulation data; (2) works based on actual data without considering the spatiotemporal characteristics of train operation; (3) works based on actual data and considering external factors but ignoring the spatiotemporal characteristics of train operation; (4) works based on the actual performance data, considering the spatiotemporal characteristics of train operation but ignoring the external factors.

Some studies are not based on actual train operation data. For example, Wang et al. [15] analyzed the four aspects of people, equipment, environment, and management and further selected 14 main influencing factors of train delay; the interpretive structure model is used to analyze the train delay. Based on scenario calculation, Ma [16] analyzed the influencing factors of train delay degree and calculated the corresponding weight through expert scoring method and analytic hierarchy process, solved the models of different scenarios by introducing genetic factor and information entropy, and solved the train operation adjustment model by example simulation, so as to adjust and optimize the train delay model.

Some studies are based on actual performance data but do not consider the spatiotemporal characteristics of the train. For example, Huang et al. [17] put the delay time of the train at the initial late station, the total delay time of train passing through each station, and the total interval buffer time for each stop, as well as the 0-1 variable that identifies whether the train is delayed through the Zhuzhou West-Changsha South interval as independent variables, and used random forest regression to predict train delays. Oneto et al. [18] proposed a fast learning algorithm for shallow and deep extreme learning machines based on the useful and actionable information in a large amount of historical train operation data of the Italian railway network and made full use of the recent memory scale data processing technology to predict train delays.

Some studies consider external factors but do not consider the spatiotemporal characteristics. For example, the research of Oneto et al. [19] does not use the historical data of train operation but uses the static rules established by railway infrastructure experts based on classical univariate statistics and uses the weather information provided by the national meteorological service to further improve the model. The train operation data changes with time and space. The model that only depends on the rules defined by experts has poor flexibility and portability, and it is hard to grasp the train operation law in the data.

More studies consider the spatiotemporal characteristics on the basis of actual operation data but ignore the impact of external factors. For example, Huang et al. [5] used the dynamic system of moving objects to generate multiattribute data, including static, time series, and spatiotemporal format, and used a three-dimensional convolutional neural network. The long-term and short-term memory cycle neural network and fully connected neural network were used to predict train delay. Zhang et al. [20] comprehensively considered the relationship between the delay propagation of current train and its adjacent trains, constructed a hierarchical prediction model of train associated delay based on wavelet neural network for delay prediction, and divided it into four categories: serious delay, dissipated delay, potential delay, and general delay. Lessan et al. [12] proposed a train delay prediction model based on Bayesian network, which used the real train operation data from high-speed railway line and adopted three different Bayesian network schemes to capture the superposition and interaction of train delays. Zeng et al. [21] designed the classification method of initial delay and associated delay on the basis of delay propagation analysis and performance data statistics. Based on the data provided by the classification method, they proposed a delay prediction model and used back-propagation neural network to predict the delay time. Hu et al. [22] established the prediction model of train delay recovery time by using multilayer perceptron and cyclic neural network with initial delay time, station stop redundancy time, and interval redundancy time. Corman and Kecman [23] used Bayesian network to predict train delay propagation based on a set of historical traffic actual data of busy sections in Sweden and fully considered the dynamic changes of train delay with time and space. Hou et al. [24] used the train operation records from the scheduled and actual train schedules to sort the modeling data, used the stepwise regression method to determine the importance of the influencing factors corresponding to the train delay time, and applied the gradient boosting regression tree to construct the delay recovery model.

It can be observed that the above research methods mainly have one or more of the following problems:

(1) The spatiotemporal correlation of train delay is not comprehensively considered.

(2) The impact of external factors such as weather and major holiday on train operation is not considered.

(3) There is too much focus on the delay prediction of one specific train but the importance of dispatchers is ignored.

(4) Some works do not use actual train operation data, and there will be problems in the actual application.

The change of weather plays an important role in train operation. Ludvigsen and Klæboe [7] evaluated how the 2010 winter weather affected rail freight operations in Norway, Sweden, Switzerland, and Poland, as well as the response behavior mobilized by railway managers to reduce adverse consequences. The results show that railway operators are not prepared to deal with the three kinds of bad conditions: low temperature, heavy snow, and strong wind. Moreover, studies have shown that 60% of the delays of freight trains are related to winter weather. For

example, with a snowfall of 5 millimeters and a temperature below −20°C, there will be a 79% change in arrival delay.

In fact, some works consider the external factors, but a common way like Huang et al. [25] did is to treat these as the nonoperational data and use the simple fully connected layers to process, but our paper thinks that these data can be better processed by treating as the feature of the nodes in graph and should be added in the model to do convolution duo to the spatiotemporal characteristics as mentioned above.

In the graph convolution, we propose a multiattention mechanism; it consists of three parts: a spatial attention mechanism for different nodes in network, a temporal attention mechanism for the correlation of traffic conditions in different time slices, and a multifeature attention mechanism for different external factors fed into MATGCN.

During the experiment, we conducted experiments without considering the spatiotemporal attention mechanism, only considering the spatiotemporal attention mechanism, and considering the above three attention mechanisms. The results show that the three attention mechanisms proposed in this paper play a positive role in improving the performance of the model.

3. The Method

Before this section, as shown in Table 1, we first give a table of notation definitions to help find the meanings of notations used in the model and method descriptions.

3.1. Train Delay Prediction. Train delay can be roughly divided into station delay, interval delay, line delay, single-train delay, boundary delay, and so on. The work of this paper focuses on the prediction of station delay which refers to the delay of trains passing through one station in a certain period of time.

The train operation network can be regarded as an undirected graph [16]. The nodes in the graph represent a series of interconnected stations, and the connection between stations is determined by the running lines of one or more trains. Any train running on the train network has an itinerary consisting of station $S = S_1, S_2, \ldots, S_N$. This itinerary is composed of a departure station, a target station, and one or more intermediate stations. These stations are distributed in different locations. For one station, the scheduled arrival time in station S is T^S_{SA} and the scheduled departure time in station S is T^S_{SD}. According to the railway operating plan, these data should be accurate and strictly implemented. It should be noted that the initial station S_1 has no scheduled arrival time, and the target station S_N has no scheduled departure time.

In this way, through the analysis of the trains at all stations, we convert the existing train operation data into spatiotemporal data and then add historical weather data from China Weather Network (https://www.tianqi.com), as well as the information of major holiday.

3.2. Data Preparation

3.2.1. Data Collection. The train operation data used in this paper comes from the train delay data of the China Railway Ticket System (https://www.12 306.cn) and the historical weather data from the China Weather website (https://www.tianqi.com) [14]. It is spliced according to date and station ID, including the train operation records of 727 stations from October 8, 2019, to January 27, 2020. The attributes include arrival delay, departure delay, wind level, weather condition, temperature, and major holiday. The train operation data is recorded in whole minute. The running data of some passed trains can be seen in Table 2.

Table 2 shows the actual operation data from the China Railway Passenger Ticket System. As shown in the table, there are three delayed trains entering Beijing South Railway Station on October 19, 2019; Table 3 shows the historical weather data published by China Weather Network with major holiday including Spring Festival and Public Sacrifice Day.

3.2.2. Data Analysis. Train operation data is typical spatiotemporal network data [5]. In the real high-speed railway network, the operation of trains has a strong spatial dependence, temporal relevance, and spatiotemporal correlation. Spatial dependence is the direct influence between adjacent stations. The number of train delays at the next station will be affected by the delays at the previous station. Temporal relevance refers to the fact that the delay of a certain time period at a certain station has the same trend as that in the past few days and weeks. Spatiotemporal correlation refers to the fact that, in the spatial dimension, the mutual influence between different stations is different. Even the same station has different effects on its adjacent stations over time, and, in the time dimension, the historical observation data of different stations have different effects on the delay status of the station and its adjacent stations at different times in the future; therefore, the train operation data of high-speed railway shows strong dynamic correlation in spatiotemporal dimension.

This paper uses three ways to sample data: the latest time series (by hour) and the time series of one day and one week. Weather conditions and major holidays also have dual attributes in time and space. From the perspective of temporal dimension, for a special station, the change of weather in a week will be greater than that in a day, and the change in a day will be greater than that in each hour. From the perspective of spatial dimension, in the same time period, different stations have different weather. For example, the weather conditions between closer stations will be more same, while the weather conditions of stations farther away will be more different. Therefore, we believe that weather factors have spatiotemporal characteristics. For major holidays, we believe that the major holiday factors have the temporal characteristics.

This paper makes statistics on the external data. Among the 1,954,176 pieces of data, about 89.59% of the day it is weak wind, about 10.02% it is middle wind, and 0.37% is

Table 1: Some notation definitions.

T^S_{SA}	The scheduled arrival time in station S
T^S_{SD}	The scheduled departure time in station S
T^S_{AA}	The actual arrival time of the train in station S
$T^S_{A\ D}$	The actual departure time in station S
$T^S_{AA} - T^S_{SA}$	The arrival delay
$T^S_{A\ D} - T^S_{\mathrm{SD}}$	The departure delay
$X^\tau_i \in \mathrm{R}$	All the features of station i in τ
$X^\tau = (X^\tau_1, X^\tau_2, \ldots, X^\tau_N)^T \in R^{N\times F}$	All features of all stations in τ
$\chi = (\mathrm{X}^1, X^2, \ldots, X^{\mathrm{t}})^T \in R^{N\times F\times t}$	All the features of all stations in t time periods
$y^\tau_i \in \mathrm{R}$	The number of arrival delays of station i in the future time period τ
$Y = (y_1, y_2, \ldots, y_N)^T \in R^{N\times T_p}$	The arrival delay sequence of all stations
$y_i = (y^{\tau+1}_i, y^{\tau+2}_i, \ldots, y^{\tau+T_p}_i)$	The arrival delay sequence of station i in the future T_P time period

Table 2: China railway ticketing system train operation data.

Train date	Train number	Station name	Expected arrival time	Expected departure time	Actual arrival time	Actual departure time	Stopover time (minutes)	Arrival delay	Departure delay
October 19, 2019	G17	Beijingnan	19:00	19:00	19:00	19:00	—	False	False
October 19, 2019	G39	Beijingnan	19:04	19:04	19:03	19:03	—	False	False
October 19, 2019	G21	Beijingnan	19:06	19:08	19:08	19:10	2	True	True
October 19, 2019	G269	Beijingnan	19:14	19:18	19:15	19:17	4	True	False
October 19, 2019	G207	Beijingnan	19:28	19:30	19:36	19:37	2	True	True
October 19, 2019	G4961	Beijingnan	19:36	19:37	19:36	19:38	1	False	True
October 19, 2019	G333	Beijingnan	19:55	19:57	19:54	19:56	2	False	False

Table 3: Historical weather data and holiday data (before classification).

Station name	Train date	Wind	Weather	Temperature	Holiday
YiMianPoBei	October 8, 2019	Westerly 4-5	Shower	11	No
YiMianPoBei	October 9, 2019	Southwest wind 4-5	Fine	17	No
YiMianPoBei	October 10, 2019	Northwest wind 4-5	lightRain	16	No
YiMianPoBei	October 11, 2019	Westerly 3-4	Fine	12	No
YiMianPoBei	October 12, 2019	North wind 3-4	Fine	10	No
YiMianPoBei	October 13, 2019	Northwest wind 3-4	Cloudy	9	No
YiMianPoBei	October 14, 2019	Westerly 3-4	Fine	8	No
YiMianPoBei	October 15, 2019	Southwest wind 4-5	Fine	12	No

strong wind; 96.63% of the trains are in good weather, 2.11% in normal weather, and 1.24% in bad weather. At the same time, about 7.14% of the days are major holiday and 92.85% are not major holiday. Table 4 shows the departure delay and arrival delay rate of train operation under various external factors. For example, in good weather, the departure delay rate of train operation is 16.38%; in normal weather, the rate is 17.78%; and, in bad weather, the rate is 19.56%.

In order to more directly observe the influence of different external factors on the change of departure and arrival rate, this paper uses a heat map to describe it. As shown in Table 5, the departure and arrival rates under different weather conditions and wind levels and in whether it is a major holiday are changing. External factors are the statistics of the proportion of the total data of each factor. For example, 7.14% of the days are major holiday. As the color gradually deepens from left to right, with the increase of wind level, the worse of weather conditions, and the influence of major holiday, the departure and arrival rates increase, that is, the external factors used in this paper have impacts on the departure and arrival rate.

3.2.3. Data Processing. However, there are nearly 80 types in different weather, wind direction, wind level, and holiday. Although many of them are different, the impact on train operation is roughly the same; for example, southwest wind levels 1-2 and northeasterly wind levels 1-2 are relatively low wind levels and have roughly the same impact on train

Table 4: Changes of departure rate and arrival rate under the influence of external factors.

External factors	Total num	Rate	Arrival delay num	Arrive delay rate	Depart delay num	Departure delay rate
Weak wind	1750872	0.8959	287903	0.1644	187942	0.1073
Middle wind	195984	0.1002	32246	0.1645	22199	0.1133
Strong wind	7320	0.0037	1272	0.1738	961	0.1313
Good weather	1888512	0.9663	309313	0.1638	203038	0.1075
Normal weather	41304	0.0211	7343	0.1778	4881	0.1182
Bad weather	24360	0.0124	4765	0.1956	3183	0.1307
Holiday	139584	0.0714	21704	0.1652	12915	0.1092
Nonholiday	1814592	0.9285	299717	0.1554	198187	0.0925

Table 5: Different external factors on the change of departure delay rate and arrival delay rate.

	nice weather	normal weather	bad weather	weak wind	middle wind	strong wind	holiday	non holiday
external factors	0.9663	0.0221	0.0124	0.8959	0.1002	0.0037	0.0714	0.9285
arrival delays	0.1638	0.1778	0.1956	0.1644	0.1645	0.1738	0.1652	0.1554
departure delays	0.1075	0.1182	0.1307	0.1073	0.1133	0.1313	0.1092	0.0925

operation. Therefore, these two types of wind direction and wind level can be classified as weak wind levels. Similarly, the wind levels are classified in this paper. The wind below level 4 is weak, the wind from level 4 to level 6 is middle, and the wind above level 6 is strong. The weather conditions are classified. Nine kinds of weather such as sunny and cloudy are classified as good weather, six kinds of weather such as moderate snow and moderate rain are classified as normal weather, and nine kinds of weather such as sleet and blizzard are classified as bad weather, as shown in Table 6.

But we find that the weather conditions, wind level, and holiday data are not numerical and cannot be fed into the MATGCN model for calculation and training. Therefore, we use one-hot encoding to transcode these data. This process is implemented by using Python machine learning third-party library scikit-learn.

As shown in Algorithm 1, the input data are spatiotemporal and external factors data and columns that need to be encoded. The program reads the original data, uses the OneHotEncoder class provided by scikit-learn to convert nonnumerical columns into one-hot encoding and combines and splices the converted data with the original data to obtain numerical data that can be applied to model calculations. The conversion result is shown in Table 7. Take the data in the first row as an example, during the period from 2:00 to 3:00 on October 8, 2019 (not a major holiday), at WanZhou Station, the temperature is 22°C, the wind level is weak, the weather is good, and there are no delayed trains.

We need to reprocess and modify the original data of the train as in Table 8, assuming that the actual arrival time of the train in station S is T^S_{AA}, the actual departure time in station S is $T^S_{A\,D}$, $T^S_{AA} - T^S_{SA}$ is defined as the arrival delay, and, similarly, $T^S_{A\,D} - T^S_{\mathrm{SD}}$ is defined as the departure delay. If $T^S_{AA} - T^S_{SA} > 0$, it will be counted as an arrival delay; if $T^S_{A\,D} - T^S_{\mathrm{SD}} > 0$, it will be counted as a departure delay.

3.3. MATGCN. The train network is defined as an undirected graph $G = (\mathrm{S}, \mathrm{E}, \mathrm{A}, \mathrm{M})$, where S is the set of all stations; $|S| = N$, and N represents the number of stations, E is the set of all edges, which represents the train line between the stations, $A \in R$, representing the connectivity between the stations, is the adjacency matrix of G, and M representing the distance between the stations is the distance weight matrix of G. Because the greater the distance between the two stations, the less the influence, the weight is also smaller. In G, each station has a number of statistical values in the time period τ, including the total number of departure delays and arrival delays. We use F to represent the number of station features, and $\mathrm{X}^\tau_i \in \mathrm{R}$ represents all features of station i in τ. $\mathrm{X}^\tau = (X^\tau_1, X^\tau_2, \ldots, X^\tau_{\mathrm{N}})^T \in R^{N\times F}$ represents all features of all stations in τ. $\chi = (\mathrm{X}^1, X^2, \ldots, X^{\mathrm{t}})^T \in R^{N\times F\times t}$ represents all the features of all stations in t time periods; that is, $\chi \in \mathrm{R}^{N\times F\times T}$. In addition, we set $\mathrm{y}^\tau_i \in \mathrm{R}$ to represent the number of arrival delays of station i in the future time period τ. Given a fixed time period τ, the various eigenmatrices of all stations on the train network generated by the train dataset in the past τ time period are used to predict the arrival delay sequence of all stations on the entire train network $Y = (y_1, y_2, \ldots, y_N)^T \in R^{N\times T_P}$ in the future T_P time period. Among them, $y_i = (y_i^{\tau+1}, y_i^{\tau+2}, \ldots, y_i^{\tau+T_P})$ represents the arrival delay sequence of station i in the future T_P time period.

The MATGCN model (as shown in Figure 1) is a significant improvement of TSTGCN [13]. TSTGCN is a train station delay prediction deep learning model we proposed before, which uses train operation data on the original high-speed railway network and effectively captures dynamic spatiotemporal characteristics to predict the delay of high-speed train stations. Our MATGCN model does some significant change based on TSTGCN. Like TSTGCN, we divide the input data into three

Table 6: Historical weather data and holiday data (after classification).

Station name	Train date	Temperature	Holiday	Wind class	Weather class
YiMianPoBei	October 8, 2019	11	No	Middle	Normal
YiMianPoBei	October 9, 2019	17	No	Middle	Good
YiMianPoBei	October 10, 2019	16	No	Strong	Good
YiMianPoBei	October 11, 2019	12	No	Weak	Good
YiMianPoBei	October 12, 2019	10	No	Weak	Good
YiMianPoBei	October 13, 2019	9	No	Weak	Good
YiMianPoBei	October 14, 2019	8	No	Weak	Good

Table 7: Coding results of model input data.

Station name	Start time	End time	Holiday	Nonholiday	Weak wind	Middle wind	Strong wind	Good weather	Normal weather	Bad weather
Wanzhou	October 8, 2019, 2:00	October 8, 2019, 3:00	0	1	1	0	0	1	0	0
Sanming	October 8, 2019, 6:00	October 8, 2019, 7:00	0	1	1	0	0	1	0	0
Linhai	October 8, 2019, 6:00	October 8, 2019, 7:00	0	1	0	1	0	0	1	0
Fenglin	October 8, 2019, 15:00	October 8, 2019, 16:00	0	1	1	0	0	1	0	0
Nanjing	October 8, 2019, 17:00	October 8, 2019, 18:00	0	1	1	0	0	1	0	0
Nanping	October 13, 2019, 15:00	October 13, 2019, 16:00	0	1	1	0	0	1	0	0

Input:
data, encoded row list;
Output:
encodeddata;
(1) Data = read(data);
(2) ec = OneHotEncoder();
(3) one hot data = ec.fit transform(encoded row list).to array();
(4) new dataFrame = DataFrame(one hot data);
(5) concat result = concat([data, new dataFrame], axis = 1);
(6) **return** concat result;

Algorithm 1: Encoding nonnumerical data.

Table 8: The total number of delayed trains at a station in a certain period of time.

Station name	Start time	End time	Departure delay	Arrival delay
WanZhouBei	October 8, 2019, 2:00	October 8, 2019, 3:00	0	0
SanMing	October 8, 2019, 6:00	October 8, 2019, 7:00	0	0
LinHai	October 8, 2019, 6:00	October 8, 2019, 7:00	1	0
FengLin	October 8, 2019, 15:00	October 8, 2019, 16:00	0	0
NanJing	October 8, 2019, 18:00	October 8, 2019, 19:00	0	3
NanPing	October 13, 2019, 16:00	October 13, 2019, 17:00	0	0

categories, the recent, daily period, and weekly period, but we add more external features into the graph nodes and redivide the input data as follows: recent-external, daily-period-external, and weekly-period-external, and further the multiattention attention mechanism we proposed is a combination of spatial attention module, temporal attention module, and multifeature attention module; it can solve the spatiotemporal data and process

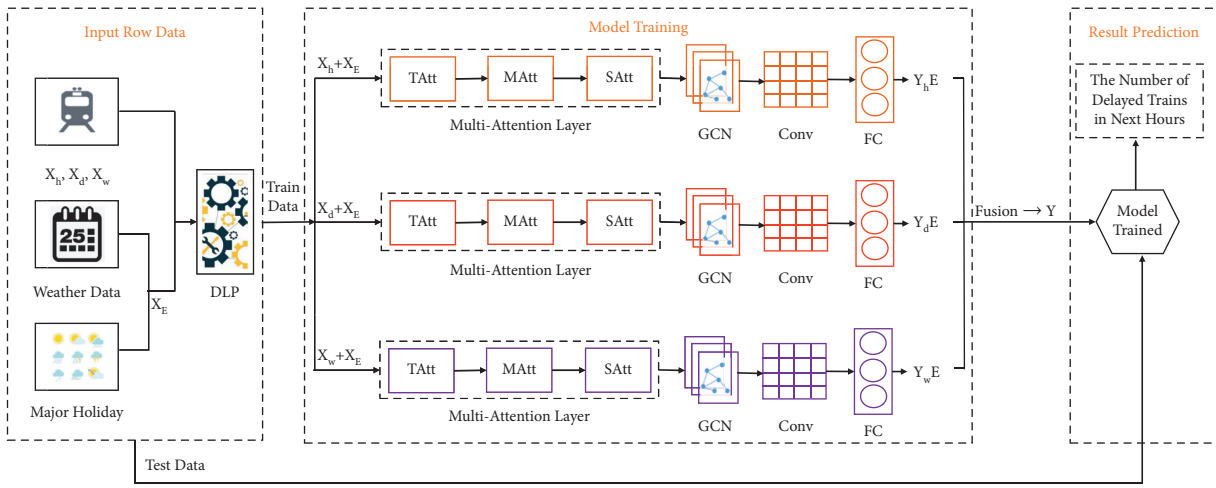


Figure 1: Structural diagram of MATGCN train delay model. TAtt: temporal attention mechanism, MAtt: multifeature attention mechanism, SAtt: spatial attention mechanism, GCN: graph convolution neural network, Conv: convolution layer, FC: fully connected layer.

the input data in every layer according to its importance to the model. So it is much better than the TSTGCN. We use the similar ways to combine the results from three components to get the final result. Then we will introduce the MATGCN in detail.

As shown in Figure 1, the input data is the integration of three time series (X_h, X_d, X_w) with external factors data X_E. When these data pass TAtt (temporal attention block) and SAtt (spatial attention block), the MATGCN model can capture the spatiotemporal correlation; when they pass the MAtt (multifeature attention block), MATGCN can add attention matrix to external factors and then model the spatial characteristics of the nodes on the train operation network through the GCN and make full use of the correlation of the graph node signals in the train operation network. Finally, the result is obtained by fusing the output of the three components through the full connection layer according to the influence weight.

3.3.1. Input Row Data. The input data are divided into three categories:

(1) Recent time series data with external factors. The arrival delay of the previous one or more stations in the past will affect the arrival delay of multiple stations in the future; among them, external factors will have an effect on it. The mathematical representation is as follows:

$$X_h + X_E = \left[\left(X_{t_0-T_h+1}, X_{t_0-T_h+2}, \ldots, X_{t_0}\right) + X_E\right] \in R^{N\times F\times T_h}. \tag{1}$$

(2) Daily-period series data with external factors. People's daily travel is regular; station delays may occur in a relatively fixed time period, such as five to six o'clock in the afternoon every day, and external factors will have an effect on it; the purpose of the daily-period component is to simulate the daily-periodity of the train arrival delay data. The mathematical representation is as follows:

$$X_d + X_E = \left[\left(X_{t_0-(T_d/T_P)\times q+1}, \ldots, X_{t_0-(T_d/T_P)\times q+T_P}, \ldots, X_{t_0-(T_d/T_P-1)\times q+1}, \ldots, X_{t_0-q+T_P}\right) + X_E\right] \in R^{N\times F\times T_d}. \tag{2}$$

(3) Weekly-period series data with external factors. The weekly attributes and time intervals of these fragments are the same as the predicted period. Normally, the traffic pattern on Wednesday is similar to the traffic pattern on Wednesday in history, but it may be very different from that on Thursday and Friday, and external factors will have an effect on it. For example, even if there are similar train delay rules every week, this rule will change under continuous blizzards. Therefore, external factors also

play a key role in exploring the rules of train delays. The mathematical representation is as follows:

$$X_W + X_E = \left[\left(X_{t_0-7\times(T_W/T_P)+1}, \ldots, X_{t_0-7\times(T_W/T_P)\times q+1}, \ldots, X_{t_0-7\times q+T_P}\right) + X_E\right] \in R^{N\times F\times T_W}. \tag{3}$$

3.3.2. GCN. In this paper, GCN is used to model the spatial characteristics of nodes on the train operation network. In the spatial dimension, train operation data is a kind of graph structure data. Different from grid data, it exists in non-Euclidean space, which makes it difficult for the traditional neural network to process. However, graph convolution neural network can directly model the original graph structure data and obtain the representation of nodes in graph structure data. In this paper, the spectral method is used to define the graph convolution. The spectral method uses the convolution theorem and Fourier transform to transfer the graph from the node domain to the spectral domain and then defines the convolution kernel in the spectral domain.

3.3.3. 2D-CNN. CNN is a type of feedforward neural network that contains convolution calculations and has a deep structure. It is specially used to process data with a similar grid structure. This paper uses 2D-CNN to model the time correlation characteristics of nodes on the train operation network. After collecting the adjacency information of each node on the train operation network in the spatial dimension, the graph convolution operation updates the node signal by merging the information of adjacent time slices along the temporal dimension to capture the dependence between adjacent time slices. Taking the r-th layer in the daily-period component as an example, its convolution operation is shown as follows:

$$X_{\mathrm{d}}^{r} = \mathrm{ReLU}\left(\phi * \left(\mathrm{ReLU}\left(g_\theta *_G \widehat{X}_d^{(r-1)}\right)\right)\right), \tag{4}$$

where ReLU is the activation function and ϕ is the temporal dimensional convolution kernel parameter.

3.3.4. Attention Mechanism. MATGCN model uses a multiattention mechanism including a spatial attention mechanism, a temporal attention mechanism, and a multifeature attention mechanism. This multiattention model can well capture the spatiotemporal correlation and process the input data in every layer according to its importance to the model.

In the temporal dimension, there is a correlation between the arrival delays of stations in different periods. The correlation of each station is also changing in different time. The arrival delays in the previous periods will affect the future arrival delays of the stations on the line.

We calculate the time weight matrix Z of the input data. The element Z_{ij} in Z represents the degree of dependence between times i and j. The calculation formula is as follows:

$$Z = V_t.sigmoi\ d\left((XU_1)U_2\right) \odot \left(U_3X\right) + b_t\big), \tag{5}$$

where, $\cdot$ means inner product, $\cdot$ means Hadamard product, $X = (X_1, X_1, \ldots, X_{T_{r-1}}) \in R^{N\times F_{r-1}\times T_{r-1}}$ represents the input data of the r-th layer of multiattention module, F_{r-1} represents the number of features of the r-th layer, T_{r-1} represents the length of the time series of the r-th layer, the activation function is sigmoid, and V_t, $b_t \in R^{T_{r-1}\times T_{r-1}}$, $U_1 \in R^N$, $U_2 \in R^{F_{r-1}\times N}$, $U_3 \in R^{F_{r-1}}$ are characteristic transformation matrices, which are learnable parameters. After that, we use the softmax function to normalize Z to ensure that the sum of attention weights is 1 and get the final time attention matrix:

$$Z' = \mathrm{softmax}_j\left(Z_{ij}\right) = \frac{\exp\left(Z_{ij}\right)}{\sum_{j=1}^{T_{r-1}} \exp\left(Z_{ij}\right)}. \tag{6}$$

The obtained time attention matrix will be directly applied to the input of the r-th layer of spatiotemporal module to obtain the input data X integrating temporal attention $X_{Z'} = X \odot Z'$; then $X_{Z'}$ will be used as input to the spatial attention module.

Different features have different effects on train delay, so, in this paper, we propose a multifeature attention mechanism to capture this difference:

$$P = V_P * \mathrm{sigmoid}\left(\left(x_h^{(r-1)}U\right)^T + b_P\right). \tag{7}$$

In the above equation, $X_h^{(r-1)} = (X_1, X_2, \ldots, X_{r-1}) \in R^{N\times F_{r-1}\times T_{r-1}}$ represents the input data of the r-th layer of multifeature module, $U \in R^{F_{r-1}\times F_{r-1}}$, $b_p \in R^{N\times F_{r-1}\times T_{r-1}}$, and $V_p \in R^{T_{r-1}\times N\times N}$ are learnable parameters, $*$ represents the matrix batch dot, the activation function is sigmoid, attention matrix P is dynamically calculated according to the current input of this layer, and $S_{i,j}$ in S semantically represents the importance of different features of different nodes to the model; after that, we use the softmax function to normalize P to ensure that the sum of attention weights is 1 and get the final time attention matrix:

$$P'_{i,j} = \frac{\exp\left(P_{i,j}\right)}{\sum_{j=1}^{N} \exp\left(P_{i,j}\right)}. \tag{8}$$

In the spatial dimension, there is a certain correlation between the arrival delays of trains at different stations; in particular, the influence between adjacent stations is highly correlated, and the interaction between adjacent stations with different distances is also different. The greater the distance between the two stations, the greater the possibility of adjusting from the delayed state to normal; then the delay impact of the current station on the next is smaller. Assuming that the distance between station i and station j is $d_{S_iS_j}$, the weight of the corresponding position of the distance matrix is

$$M_{ij} = \frac{1}{d_{s_i s_j}}. \tag{9}$$

Consider the static characteristics of high-speed railways network. We calculate the correlation weight matrix C of the input data. Element C_{ij} in C represents the correlation between stations i and j. The calculation formula is as follows:

$$C = V_S.\text{sigmoid}\left(\left(\left(X_{Z'}.W_1\right)W_2\right) \odot \left(W_3 X_{Z'}\right) + b_S\right). \tag{10}$$

In the above equation, $X_{Z'} \in R^{N\times F_{r-1}\times T_{r-1}}$ represents the input data processed by the multifeature attention module of the r-th layer; $W_1 \in R^{T_{r-1}}$, $W_2 \in R^{F_{r-1}\times T_{r-1}}$, $W_3 \in R^{F_{r-1}}$, and $V_S, b_S \in R^{N\times N}$ are the feature conversion matrices, which are learnable parameters.

By fusing the correlation weight matrix C and the distance weight matrix M', we obtain the spatial attention matrix Q. Similarly, we use the softmax function to normalize Q to obtain the final spatial attention matrix Q'. The calculation formula is as follows:

$$\begin{aligned} Q &= C \odot M', \\ Q' &= \text{softmax}_j\left(Q_{ij}\right) = \frac{\exp\left(Q_{ij}\right)}{\sum_{j=1}^{N}\exp\left(Q_{ij}\right)}. \end{aligned} \tag{11}$$

The spatial attention matrix can capture the correlation and distance influence between nodes on the train operation network. When performing graph convolution, we will dynamically adjust the influence weight between nodes with adjacency matrix and spatial attention matrix.

3.3.5. Multicomponent Fusion. In central cities such as Beijing, the passenger flow has obvious peak periods in the morning or evening, and trains may also be delayed. Therefore, the output of daily-period and weekly-period components is more critical. In some remote areas, due to the lack of strong periodic passenger flow, the possible prediction results of daily-period and weekly-period components are less accurate. Therefore, when the outputs of these three components are fused, the weight of the influence of the three components on each node is different, which needs to be determined according to the historical data of train operation. So the final fusion result of the three components is

$$Y = W_h \odot Y_h E + W_d \odot Y_d E + W_w \odot Y_w E. \tag{12}$$

In the above equation, $W_h, W_d, W_w \in R^{N\times P}$ can be obtained through e-learning; it reflects the impact of the three time dimensions on the prediction objectives, $\odot$ stands for Hadamard product, P stands for predicted time step, and $Y_h E, Y_d E, Y_w E$, respectively, represent the final output results obtained after the output of the recent, daily-period, and weekly-period components passing through the fully connected layer.

3.3.6. DLP. DLP (Data Link Processing) is built on the basis of NumPy, Pandas, and other third-party Python libraries and combines the external factors data (X_E) with three time series data (X_h, X_d, X_w) according to station ID and train date to obtain the time series-external factors data.

4. Results and Discussion

In this paper, we use the three following common evaluation indexes to evaluate the prediction performances of ANN, SVR, LSTM, RF, TSTGCN, and MATGCN models. They are mean absolute error (MAE), root mean square error (RMSE), and mean absolute percentage error (MAPE). The calculation formulas are as follows:

$$\begin{aligned} \text{MAE} &= \frac{1}{n}\sum_{i=1}^{n}\left|x_i - \widehat{x}_i\right|, \\ \text{RMSE} &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_i - \widehat{x}_i\right)^2}, \\ \text{MAPE} &= \frac{1}{\text{n}}\sum_{i=1}^{n}\left|\frac{x_i - \widehat{x}_i}{x_i}\right| \times 100\%. \end{aligned} \tag{13}$$

In the above equation, x_i is the actual value, $\widehat{x}_i$ is the predicted value, and n is the number of test samples.

We implement the MATGCN model on the MXNet framework. In our model, the term of the Chebyshev polynomial is set to 3, and all graph convolution layers use 64 convolution kernels. All time convolutional layers also use 64 convolution kernels and adjust the time span of data by controlling the step size of time convolution. We set $\text{T}_\text{h} = 3$, $\text{T}_\text{d} = 1$, and $\text{T}_\text{w} = 1$. The size of the prediction window is $\text{T}_\text{p} = 1$; that is, our goal is to predict the number of delays in the arrival of the station in the next hours. In the training phase, the batch size is 4 and the learning rate is 0.000 01.

We implement ANN, SVR, RF, LSTM, TSTGCN, and MATGCN models on Windows 10 system. Among them, ANN uses a single hidden layer network structure with a learning rate of 0.01; the kernel function of SVR selects poly, and the learning rate is 0.001; the learning rate of RF is 0.001, and the batch size is 128; LSTM contains two hidden layers, and the activation function of the hidden layer is ReLU, the gate activation function is sigmoid, the number of outputs per layer is 100, the activation function of the output layer is softmax, the loss function is L2Loss, and the learning rate is 0.001. TSTGCN is based on MXNet, the batch size is 4, and the learning rate is 0.000 01. Except for RF and TSTGCN, the training batch sizes of other models are all 64, and the other parameters remain the default.

We compare MATGCN with the other five learning models on the processed station delay dataset. Table 9 shows the results of train arrival delay prediction performance in the next hour. Among them, the best two scores are displayed in bold.

It can be observed that, among the five benchmark models, the best MAE value is 0.444 7 (SVR), the best RMSE value is

Table 9: Comparison of one-hour prediction performance of six models.

Model	MAE	RMSE	MAPE
ANN	0.6309	0.8499	53.6608
SVR	0.4447	0.8299	63.7141
RF	0.6146	0.9039	54.9183
LSTM	0.4960	0.8507	61.4930
TSTGCN	0.1600	0.4500	34.3600
MATGCN without MAtt	**0.1500**	**0.4200**	**24.8300**
MATGCN	**0.1000**	**0.3100**	**15.9300**

The best two scores are displayed in bold to show the results clearly.

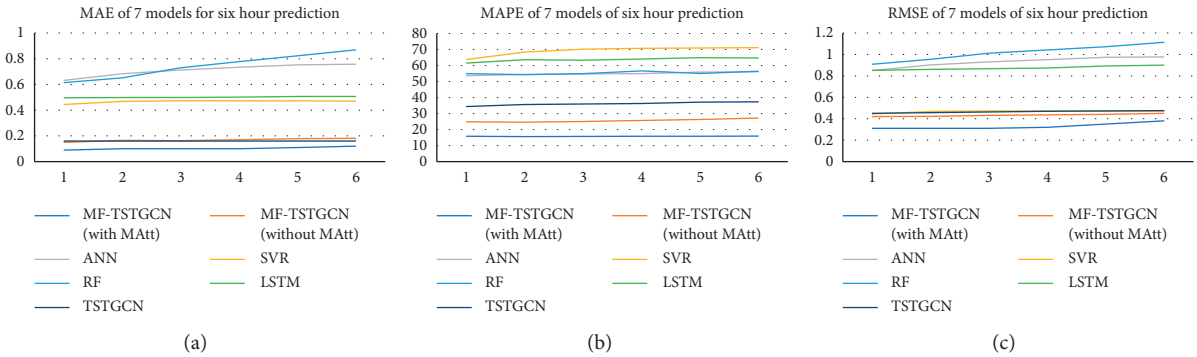


Figure 2: Execution effect of 7 methods for six-hour prediction. (a) MAE. (b) MAPE. (c) RMSE.

0.8299 (SVR), the best MAPE value is 53.660 8 (ANN), and the TSTGCN score is 0.160 0, 0.450 0, and 34.360 0; the effects of ANN, SVR, RF, and LSTM that only use train delay data as time series data for prediction are far inferior to TSTGCN. Although TSTGCN considers that train station delay data is spatiotemporal data, it does not consider the external factors of train operation. It can be seen that, compared with TSTGCN, MATGCN without MAtt has a 6.66% decrease in MAE, a 6.66% decrease in RMSE, and a 27.73% decrease in MAPE, and MATGCN with MAtt has a 33.33% decrease in MAE, a 26.19% decrease in RMSE, and a 35.84% decrease in MAPE and obtains the best prediction performance.

Figures 2(a)–2(c) show the performance of various methods to predict the number of train delays at stations in the next 1 to 6 hours. We can observe the changes in the prediction performance of each method as the prediction duration increases. In general, as the prediction duration increases, the corresponding prediction difficulty becomes greater, so the prediction error is also increasing. The errors of ANN, SVR, RF, and LSTM are always maintained at a high level. The prediction ability of RF decreases sharply. In contrast, the performance of LSTM decreases slowly. It can be seen from the figure that the MATGCN proposed in this paper has also obtained better prediction results than TSTGCN and can achieve the best prediction performance almost at any time. Even in the long-term prediction, the error remains at a low level. This is because the spatiotemporal correlation and external factors are particularly important in the long-term prediction.

Through the above analysis, we find that, compared with other existing methods, MATGCN can more comprehensively consider the spatiotemporal and external factors that affect train operation and shows excellent performance in station delay prediction.

5. Conclusions

Focusing on the spatiotemporal and dynamic correlation of high-speed railway train operation data, this paper constructs MATGCN model based on multiattention mechanism to predict the train delay at high-speed railway stations. This model combines multiattention mechanism and spatiotemporal convolution, including spatial dimension graph convolution and temporal dimension standard convolution, to capture the spatiotemporal characteristics of train operation data at the same time, and adds multifeature attention mechanism to process the external factors such as weather conditions, wind level, and major holiday to achieve more accurate prediction. In the experimental stage, we compare and evaluate the MATGCN model proposed in this paper with the ANN, SVR, LSTM, RF, and TSTGCN models and use MAE, RMSE, and MAPE to evaluate the prediction effect of the model. The result shows that the three attention mechanisms play a positive role in improving the performance of the model.

Data Availability

The train operation and external feature data used to support the findings of this study have been deposited in the Figshare repository: https://figshare.com/articles/dataset/A_high-speed_railway_network_dataset_from_train_operation_records_and_weather_data/15 087 882.

Additional Points

The focus is to propose a multifeature attention mechanism to capture the different effects of different external factors such as weather and holidays on train operation. The results show that the MATGCN is better than TSTGCN.

Disclosure

This paper is based on the authors' earlier work TSTGCN: https://ieeexplore.ieee.org/document/9511 425.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The paper was supported by the National Natural Science Foundation of China (no. 61803020) and Fundamental Research Funds for the Central Universities (no. 2021QY010).

References

[1] Z. Jiang and Q. Miao, "Delay Influence and Its Mitigation Measures of Train Operation in Urban Rail Transit," *Modern Urban Transit*, vol. 5, 2009.

[2] Y. Feng, "High Speed Railway Delay Forecasting Method Based on Artificial Neural network," *Southwest Jiaotong University*, Chengdu, China, 2019.

[3] Y. Yu, Y. Zhang, S. Qian, and Y. Hu, "A Low Rank Dynamic Mode Decomposition Model for Short-Term Traffic Flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, 2020.

[4] J. Wang, Y. Zhang, Y. Wei, and Y. Hu, "Metro Passenger Flow Prediction via Dynamic Hypergraph Convolution Networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, 2021.

[5] P. Huang, C. Wen, L. Fu, Q. Peng, and Y. Tang, "A deep learning approach for multi-attribute data: a study of train delay prediction in railway systems," *Information Sciences*, vol. 516, pp. 234–253, 2020.

[6] Y. Wang, Y. Zhang, X. Piao, and Y. Hu, "Traffic data reconstruction via adaptive spatial-temporal correlations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 4, pp. 1531–1543, 2018.

[7] J. Ludvigsen and R. Klæboe, "Extreme weather impacts on freight railways in Europe," *Natural Hazards*, vol. 70, no. 1, pp. 767–787, 2014.

[8] Y. Li and R. Yu, "Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting," GitHub, 2018, https://github.com/liyaguang/DCRNN.

[9] Y. Liu, J. Guo, C. Luo, and L. Meng, "Big data analysis and application prospect of train operation performance," *CHINA RAILWAY*, vol. 000, no. 6, pp. 70–73, 2015.

[10] S. Milinković and M. Marković, "A fuzzy Petri net model to estimate train delays," *Simulation Modelling Practice and Theory*, vol. 33, pp. 144–157, 2013.

[11] N. Marković and S. Milinković, "Analyzing passenger train arrival delays with support vector regression," *Transportation Research Part C: Emerging Technologies*, vol. 56, pp. 251–262, 2015.

[12] J. Lessan, L. Fu, and C. Wen, "A hybrid Bayesian network model for predicting delays in train operations," *Computers & Industrial Engineering*, vol. 127, pp. 1214–1222, 2019.

[13] D. Zhang, Y. Peng, and Y. Zhang, "Train Time Delay Prediction for High-Speed Train Dispatching Based on Spatio-Temporal Graph Convolutional Network," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[14] D. Zhang, Y. Peng, and Y. Xu, "A High-Speed Railway Network Dataset from Train Operation Records and Weather Data," *Figshare. Dataset.*, https://doi.org/10.6084/m9.figshare.15087882.v3, 2021.

[15] J. Wang, Y. Peng, and J. Lu, "Ism-based analysis of causes of train delay," *CHINA RAILWAY*, vol. 000, no. 001, pp. 48–52, 2020.

[16] Q. Ma, "Research on Adjustment and Optimization of Train Delay Based on Scenario Computing," *Lanzhou Jiaotong University*, Lanzhou, China, 2016.

[17] P. Huang, Q. Peng, and C. Wen, "Random forest prediction model for Wuhan-Guangzhou HSR primary train delays recovery," *Journal of the China Railway Society*, vol. 40, no. 7, pp. 1–9, 2018.

[18] L. Oneto, E. Fumeo, G. Clerico et al., "Train delay prediction systems: a big data analytics perspective," *Big data research*, vol. 11, pp. 54–64, 2018.

[19] L. Oneto, E. Fumeo, and G. Clerico, "Advanced Analytics for Train Delay Prediction Systems by Including Exogenous Weather data," in *Proceedings of the 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 458–467, IEEE, Montreal, QC, Canada, October 2016.

[20] Q. Zhang, F. Chen, and T. Zhang, "Intelligent prediction and characteristic recognition for joint delay of high speed railway trains," *Acta Automatica Sinica*, vol. 45, no. 12, pp. 2251–2259, 2019.

[21] Y. Zeng, F. Chen, and C. Shahabi, "A prediction model for timetable delays in dispatching area using neural network," *Railway Standard Design*, vol. 63, no. 3, p. 6, 2019.

[22] Y. Hu, Q. Peng, and G. Lu, "Train delay recovery time prediction model based on initial late point and redundant time," *Journal of transportation engineering and information*, vol. 18, no. 2, pp. 93–102, 2020.

[23] F. Corman and P. Kecman, "Stochastic prediction of train delays in real-time using Bayesian networks," *Transportation Research Part C: Emerging Technologies*, vol. 95, pp. 599–615, 2018.

[24] Y. Hou, C. Wen, P. Huang, L. Fu, and C. Jiang, "Delay recovery model for high-speed trains with compressed train dwell time and running time," *Railway Engineering Science*, vol. 28, no. 4, pp. 424–434, 2020.

[25] P. Huang, C. Wen, L. Fu et al., "Modeling train operation as sequences: a study of delay prediction with operation and weather data," *Transportation Research Part E: Logistics and Transportation Review*, vol. 141, Article ID 102022, 2020.