

Spatiotemporal Analysis of Urban Traffic Congestion Via Deep Convolutional Architectures

Chuanyi Xu

*School of Economics and Management, Harbin Institute of Technology, Harbin, China
1565431411@qq.com*

Abstract. Urban traffic congestion is a key issue affecting traffic efficiency, environmental sustainability, and economic productivity. Traditional transportation analysis methods mainly rely on predefined static adjacency matrices and often separately process spatial and temporal features (CNN), making it difficult to capture the inherent complex spatiotemporal dependencies in urban traffic patterns. This paper proposes a deep learning-based spatiotemporal traffic congestion analysis method using Graph Convolutional Networks (GCN). We design a novel Deep Spatiotemporal Traffic Network (DSTN) that models the spatiotemporal dynamic features of traffic by integrating 3D convolution and attention mechanisms, enhancing the model's ability to capture cyclical patterns and distinguish dynamic peak periods. The model is trained and evaluated on a large-scale traffic dataset collected using ring coil detectors and GPS trajectories.

Keywords: traffic congestion, spatio-temporal analysis, deep learning, convolutional neural networks, urban traffic

1. Introduction

With the acceleration of urbanization and the continuous increase in the number of vehicles, traffic congestion has become one of the core challenges faced by modern cities. According to statistics, major cities worldwide suffer annual economic losses exceeding hundreds of billions of dollars due to congestion, while the associated carbon emission issues also severely hinder the achievement of sustainable development goals. To address traffic congestion, optimizing intelligent control of traffic networks and upgrading management systems have become increasingly important [1]. However, traditional traffic monitoring methods (such as loop detectors and radar sensors) have certain effects but are limited by complexities in installation, high maintenance costs, and difficulties in covering complex road networks. In recent years, with breakthroughs in computer vision technology and the widespread use of urban surveillance cameras, video-based traffic monitoring technologies have provided new opportunities for real-time traffic status monitoring. For example, convolutional neural networks (CNN) divide cities into grid maps using latitude and longitude coordinates [2], learning and analyzing spatial information from long-term traffic flow to predict potential congestion conditions in each geographic area. Although CNNs excel at extracting spatial features between grid-based regions, they essentially overlook the spatial heterogeneity, dynamic interactivity, and hierarchical connections of traffic networks [3]. This fundamental limitation has

prompted researchers to introduce graph convolutional networks (GCN) into the transportation field [4]. GCNs are more aligned with the irregular topological characteristics of traffic networks, especially excelling in modeling dynamic interactions and capturing multi-hop spatial dependencies. With the development of lightweight deep learning models, real-time analysis of traffic congestion on edge devices (such as smart cameras and embedded systems) has become feasible. However, practical applications still face challenges such as lighting changes, weather disturbances, vehicle occlusion, and dynamic traffic flows. This study aims to develop a real-time, robust traffic congestion detection system through efficient convolutional neural networks combined with temporal modeling technology and to address the issue of insufficient labeled data using data augmentation and domain adaptation strategies. Through multi-scenario performance evaluations, this project will provide a scalable and low-cost traffic management solution for smart cities.

2. Literature review

This section reviews the evolution of traffic congestion analysis methods, focusing on the transition from traditional statistical methods to advanced deep convolutional architectures, and emphasizes the main limitations of current research.

2.1. Classical methods of traffic congestion analysis

Early modeling work on traffic congestion primarily relied on non-deep learning methods that laid the groundwork for establishing dynamic traffic networks, but they had inherent limitations in addressing the spatio-temporal complexities of urban networks.

2.1.1. Traditional statistical methods

Integrating statistical models such as the autoregressive integrated moving average model (ARIMA) and its variants was one of the earliest approaches applied to traffic forecasting, utilizing linear time series characteristics to predict short-term traffic flow [5]. However, these models struggled to capture the inherent non-linear dependencies in traffic data, such as sudden congestion caused by accidents or weather events. Support vector machines (SVM) introduced a more flexible framework by mapping data into high-dimensional feature space, but their performance still heavily depended on manual feature engineering, requiring domain expertise to identify relevant spatio-temporal patterns. Kalman filtering techniques, designed for state estimation in dynamic systems, were adapted for traffic forecasting to handle noise in sensor data, but they assumed linear system dynamics and failed to model complex interactions between interconnected road segments [6]. Overall, these methods lacked the ability to simultaneously model the spatial correlation between adjacent roads and the temporal dependencies across different time scales, limiting their applicability in large urban networks.

2.1.2 Early deep learning methods

The emergence of deep learning signifies a significant shift in traffic analysis, as neural networks can automatically extract features from raw traffic data. Convolutional Neural Networks (CNNs) have become powerful tools for capturing spatial patterns, leveraging their ability to process grid-structured data (such as traffic speed grids on urban maps). CNNs can effectively model local spatial dependencies in traffic flow, but their fixed grid-based receptive fields struggle to explain the irregular topology of road networks, as congestion spreads along actual road sections rather than

artificial grid boundaries [7]. Recurrent Neural Networks (RNNs) and their Long Short-Term Memory (LSTM) variants address the temporal dimension issue by processing sequential traffic data and retaining information from past time steps. LSTMs outperform traditional methods in predicting traffic flow because they can capture long-term temporal dependencies, but their architecture overlooks the spatial relationships between non-adjacent but functionally connected road segments (such as main roads and their feeder streets) [8]. To overcome the limitations of unimodal architectures, hybrid models that combine CNNs and RNNs have been proposed. A CNN-LSTM framework has been developed that uses CNNs to extract spatial features from traffic grids and LSTMs to model temporal dynamics, thus improving prediction accuracy [9]. However, these hybrid models rely on manually designed spatial connections (such as predefined grid neighborhoods), which may not align with the actual traffic propagation patterns in complex road networks, leading to suboptimal representations of spatiotemporal correlations.

2.2. Breakthroughs and evolution of Graph Convolutional Networks (GCN)

2.2.1. Theoretical foundation of GCN

Grid-based CNNs cannot model irregular road network topologies, which led to the adoption of Graph Convolutional Networks (GCN). GCN explicitly represents the road network as a graph, where nodes correspond to road segments and edges reflect their connectivity [10]. This paradigm shift enables more accurate modeling of traffic congestion propagation. Kipf, T.N. and Welling, M [11] introduced a simplified GCN architecture that performs convolution on graphs through spectral decomposition, revolutionizing graph-based learning. Their work demonstrates that GCN can effectively learn from non-Euclidean data, such as road networks, by aggregating features from neighboring nodes, making it particularly suitable for traffic analysis, where spatial relationships are defined by road network connectivity rather than geometric proximity.

2.2.2. Applications in traffic congestion analysis

The early applications of GCN in traffic modeling focused on combining graph-based spatial features with temporal models. For instance, T-GCN combines GCN with Gated Recurrent Units (GRU) to capture both spatial dependencies and temporal dynamics simultaneously [12]. Although T-GCN outperforms previous hybrid models, it relies on a static adjacency matrix derived from the road network topology, which fails to account for the dynamic changes in spatial dependencies caused by time-varying congestion (for example, a congested highway may temporarily reduce traffic flow on nearby surface streets). To address this limitation, dynamic graph methods have been developed to adaptively update the graph structure based on real-time traffic conditions, introducing Graph WaveNet [13] which employs a learnable adjacency matrix to model implicit spatial relationships and improve prediction accuracy during non-periodic congestion events. However, Graph WaveNet is still insufficient in coupling temporal and spatial modeling, treating the adaptive adjacency matrix as a time-invariant parameter within each training step. Recent advancements have explored multi-graph fusion to capture different types of spatial dependencies (e.g., topological, functional, and distance-based relationships). For example, ST-MGCN [14] was proposed, which integrates multiple graph structures using an attention mechanism to dynamically weight different spatial correlations. Despite ST-MGCN's stronger representational power, it has high computational complexity due to its multi-branch architecture, limiting its applicability in real-time traffic management systems.

2.3. Horizontal comparison of related technologies

Although GCN (Graph Convolutional Networks) has dominated traffic spatio-temporal analysis, other deep learning paradigms possess complementary advantages and limitations worth considering [15]. The attention-based Transformer has gained attention for its ability to model long-range temporal dependencies in sequential data. A study applied the Transformer to traffic prediction, demonstrating its superiority in capturing periodic patterns over longer time spans (e.g., weekly traffic cycles). However, the Transformer incurs high computational costs due to the complexity of its second-order attention, and its global attention mechanism often fails to prioritize local spatial patterns that are crucial for modeling short-term congestion propagation. Within the broader family of Graph Neural Networks (GNNs), variants such as GraphSAGE and Graph Attention Network (GAT) offer alternative methods for aggregating node features. GraphSAGE employs inductive learning to generalize to unseen road segments [16], making it suitable for dynamically changing road networks (e.g., temporary construction zones). Nevertheless, it does not explicitly model the spatio-temporal coupling of traffic congestion, treating spatial and temporal features as independent entities. GAT introduces an attention mechanism to learn node-specific weights for adjacent features [17], allowing for more flexible modeling of spatial dependencies. In traffic applications, this enables nodes (road segments) to dynamically adjust their focus on relevant neighbors based on current conditions. However, GAT assumes that attention weights evolve smoothly over time, which may not hold in dynamic traffic scenarios, as the importance of nodes can change suddenly (e.g., during a sudden traffic jam).

2.4. Research gaps and integration with current work

Despite significant progress, existing spatiotemporal traffic analysis methods still have several key limitations. For example, most GCN-based models depend on predefined static graphs or incrementally updated adjacency matrices, which cannot capture the rapid changes in spatial dependencies during congestion propagation. This greatly limits their capability to model non-periodic events (such as accidents or weather-related disruptions). Current methods generally aggregate features on a single scale, which leads to the loss of critical fine-grained or macro patterns. For instance, sudden congestion (such as at busy intersections) and periodic congestion (such as peak hour traffic across the city) exhibit different spatiotemporal characteristics, and the joint modeling of local and global traffic patterns has not been adequately addressed. Additionally, there are computational efficiency issues; dynamic graph methods and multi-graph architectures achieve higher accuracy at the cost of increased parameter count and computational complexity, making them impractical for real-time traffic management systems that require low-latency predictions.

To overcome these limitations, we propose DSTN, a dynamic spatiotemporal network that (1) automatically infers the time-varying graph structure through a novel adaptive graph generation mechanism, which updates node connections based on real-time traffic flow and historical congestion patterns; (2) employs a dual-branch architecture to hierarchically aggregate multi-scale traffic patterns, where the local branch captures fine-grained dynamics of individual intersections and the global branch models trends over large-scale networks, with a cross-attention mechanism facilitating information exchange between scales. At the same time, by introducing sparse graph regularization and parameter sharing across time steps, we maintain computational efficiency, ensuring that the model can be deployed on edge devices for real-time applications.

3 Research methods

3.1. Core model: Dynamic Spatio-Temporal Network (DSTN)

3.1.1. Model overall architecture

DSTN is based on a Graph Convolutional Network (GCN) framework. The core design goal is to address three major issues in existing traffic analysis models: "the static graph structure cannot accommodate dynamic congestion, the loss of key patterns due to single-scale features, and high computational complexity that makes real-time deployment difficult." The overall model follows the logical flow of "data input - dynamic processing - feature fusion - result output". Through the synergistic effects of adaptive graph generation module, multi-scale feature aggregation module, and efficiency optimization module, it achieves precise modeling and efficient computation of traffic spatio-temporal dynamics. The conceptual framework structure is as follows: the input layer receives multi-source traffic data (loop sensors, GPS); the processing layer consists of three core modules, which respectively carry out dynamic graph construction, multi-scale feature extraction, and computational efficiency optimization; the output layer provides predictions of traffic congestion status for 5/15/30 minutes. The collaborative logic of the three modules is specifically illustrated as follows: first, the adaptive graph generation module dynamically updates the adjacency relationship of the road network graph based on real-time traffic flow features (speed, flow, occupancy) and historical congestion patterns, breaking the constraints of traditional static graphs on spatio-temporal dependencies; second, the multi-scale feature aggregation module processes data in a parallel manner through a "local-global" dual-branch system, capturing both fine-grained dynamics at the intersection level and macro trends at the road network level, while utilizing a cross-attention mechanism to facilitate information exchange between scales, preventing the loss of key patterns under a single scale; finally, the efficiency optimization module reduces the computational load of the model without sacrificing accuracy through sparsity constraints and parameter sharing, ensuring the model can be deployed on edge devices (such as smart cameras, embedded systems) to meet real-time traffic monitoring needs.

3.1.2 Key functional module design

Adaptive Graph Generation Module: The core of this module is to construct a road network graph structure that "dynamically evolves with traffic conditions," rather than relying on a predefined static adjacency matrix. The implementation process is divided into two steps. The first step involves real-time traffic feature extraction based on 3D convolution. Given the "spatial-temporal" two-dimensional characteristics of traffic data, 3D convolution operations are employed (with dimensions defined as $k_s \times k_s \times k_t$, where k_s is the spatial convolution kernel size, and k_t is the temporal convolution kernel size) to encode real-time traffic data features. The input data dimensions are $B \times C \times H \times W \times T$ (where B is the batch size, C is the number of feature channels, set to 3 here corresponding to speed, flow, and occupancy; $H \times W$ is the spatial grid dimension; T is the time step, set to 6 here corresponding to 6 five-minute time slots within 30 minutes). Through 3D convolution (with a kernel size of $3 \times 3 \times 3$, stride 1, and padding 1), local association features for each road segment within the "spatial neighborhood time window" are extracted, resulting in output dimensions of $B \times C' \times H \times W \times T$ (where C' is the number of feature channels after convolution, set to 64). We denote this as the real-time feature matrix X_{real} . The second step involves learning an adjacency matrix that integrates historical congestion similarity. The adjacency matrix $A \in \mathbb{R}^{N \times N}$

(where N is the number of nodes in the road network, i.e., the number of road segments) needs to consider both the "real-time traffic feature similarity" and "historical congestion pattern similarity" to balance dynamism and stability. The calculation formula is as follows: $A_{i,j} = \alpha \cdot \text{Sim}(X_{\text{real},i}, X_{\text{real},j}) + (1-\alpha) \cdot \text{Sim}(X_{\text{hist},i}, X_{\text{hist},j})$, where $A_{i,j}$ represents the connection weight between node i (road segment i) and node j (road segment j); a larger weight indicates a stronger correlation in traffic states between the two segments; $\alpha \in [0,1]$ is a weight coefficient, set to 0.6 here through validation set tuning, giving priority to the dynamic effects of real-time features; $\text{Sim}(\cdot, \cdot)$ is the cosine similarity function that calculates the similarity between two node feature vectors; $X_{\text{hist},i}$ is the historical congestion feature vector of node i , obtained by averaging the traffic data from the same period over the past 30 days (e.g., during a weekday peak from 7:00 to 7:30), with dimensions consistent with $X_{\text{real},i}$. Through this formula, the adjacency matrix can be dynamically adjusted with real-time traffic conditions (such as changes in road segment correlations due to sudden congestion), while preserving the regularity of historical congestion patterns (such as the fixed congestion associations during Monday peak hours).

Multi-scale feature aggregation module: Adopting a "dual-branch parallel cross-attention interaction" structure, it processes the fine-grained and macroscopic scale features of traffic data separately. The local branch uses a $3 \times 3 \times 3$ small receptive field 3D convolution to address fine-grained dynamics at individual intersections and short-distance road segments - for example, the stopping and starting of vehicle flows at an intersection within 5 minutes and rapid changes in lane occupancy. The receptive field of this convolution kernel only covers a local area, allowing for precise capturing of short-term traffic fluctuations at the intersection level, ultimately outputting a local feature matrix F_{local} (dimensions $B \times C' \times H \times W \times T$, with C' set to 64); the global branch uses dilated convolution to expand the receptive field to capture macroscopic trends at the road network level, such as peak congestion spreading from the CBD to the suburbs and the saturation state of regional road network traffic. The dilation rate is set to 2 (determined through validation set tuning), allowing for an expansion of the receptive field from $3 \times 3 \times 3$ to $7 \times 7 \times 7$ without increasing the convolution kernel parameters, covering approximately a 1km radius in spatial dimensions and a 35-minute time slice in temporal dimensions, effectively covering the overall road network characteristics of high-density urban areas or suburban transitional zones, outputting a global feature matrix F_{global} (also with dimensions $B \times C' \times H \times W \times T$, $C'=64$). To ensure complementarity between local and global features, a cross-attention layer is designed to compute the correlation weights between the two types of features. The attention weight matrix W_{att} (dimensions $T \times T$) is obtained through the formula $W_{\text{att}} = \text{Softmax}(C' F_{\text{local}} T \cdot F_{\text{global}})$, highlighting areas in the local features that have a significant impact on global trends, such as the diffusion effects of congested intersections on surrounding road networks. The final fused feature matrix F_{fusion} is obtained through $F_{\text{fusion}} = F_{\text{local}} \cdot W_{\text{att}} F_{\text{global}}$, balancing fine-grained dynamics with macroscopic trends. The efficiency optimization module is designed to meet the real-time computing needs of edge devices by reducing the model complexity from both "graph structure sparsification" and "parameter sharing": on one hand, L1 regularization is used to constrain the sparsity of the adjacency matrix, as most road segments in the traffic network only have strong correlations with adjacent segments, and redundant connections would increase computation, thus an L1 regularization term $L_{\text{sparse}} = \lambda \cdot \|A\|_1$ (with $\lambda = 1e-4$, determined by validation set tuning) is added to the loss function, driving the weights of invalid connections in the adjacency matrix towards 0, ultimately maintaining a sparsity of over 70%, meaning only about 30% of the strong correlation connections are kept, significantly reducing the matrix multiplication computation of the GCN layer; on the other hand, cross-time-step parameter sharing reduces redundant parameters. Considering the strong correlation

of traffic data across continuous time steps (like adjacent 5-minute time slices), the 3D convolution kernel parameters of the two adjacent time steps are shared, meaning that step t and step $t-1$ share the same set of convolution kernels. Experimental statistics show that this design can reduce the model parameter count by about 30%, with the original parameter count being approximately 2.1M. After sharing, about 1.5M, and there is no significant decrease in prediction accuracy, it can fully adapt to the computing power of edge devices such as intelligent cameras (for example, NVIDIA Jetson Nano).

3.1.3. Numerical methods and computational

Framework Optimizer Selection and Parameter Settings

The Adam optimizer is used for model training, which incorporates weight decay on top of the basic Adam optimizer, effectively reducing overfitting. The specific parameter settings are as follows: Initial learning rate: $1e-4$ (based on commonly used initial learning rates in deep learning models in the transportation field, adjusted through the gradient descent trend of the validation set); Weight decay coefficient: $1e-5$ (to balance the model's fitting ability and generalization ability, preventing overfitting due to excessively large parameters); Learning rate scheduling strategy: Cosine annealing scheduling is used, where the learning rate is decayed to 10% of the initial value according to a cosine curve every 50 epochs, before restarting to the initial value, promoting the model to escape local optima in later epochs.

Loss Function Design: Considering that in traffic congestion analysis, both 'significant errors from sudden congestion' and 'overall prediction stability' must be addressed simultaneously, a hybrid MAE-MSE loss function is adopted, expressed as: $L_{total} = 0.6 \cdot MAE(\hat{y}, y) + 0.4 \cdot MSE(\hat{y}, y)$ where $\hat{y}$ is the model's predicted traffic congestion state (quantified by speed or occupancy), and y is the true value; MAE (mean absolute error) is sensitive to outliers (such as sudden drops in speed due to unexpected congestion), which forces the model to focus on large error samples; MSE (mean square error) penalizes the square of the overall error, which can enhance the model's overall stability in predictions; the weight coefficients 0.6 and 0.4 are determined through cross-validation on the validation set (comparing combinations like 0.5-0.5 and 0.7-0.3, this ratio shows optimal comprehensive performance of the model in abnormal scenarios and regular scenarios).

Training Framework and Convergence Function: Training Framework: Built on the PyTorch 2.0 deep learning framework, combined with CUDA 12.0 for GPU acceleration (using NVIDIA RTX 3090 graphics card, single batch training time is about 0.8s); Batch Size: 32 (to balance memory usage and training stability; a batch size that is too large can cause memory overflow, while one that is too small leads to large training fluctuations); Convergence Condition: The maximum training epoch is set to 200, and an early stopping strategy is used—if the validation set loss does not decrease for 10 consecutive epochs (with fluctuations less than $1e-5$), training is stopped early to avoid overfitting. Experiments show that the model usually converges within 120-150 epochs, with validation set loss stabilizing between 0.08-0.12 (after normalization).

3.1.4. Grid and resolution design

To ensure the model input aligns with actual traffic management needs, the resolution is designed from both spatial and temporal dimensions:

Spatial Resolution: Road segments are divided according to the actual jurisdiction of traffic management units to avoid disrupting the topological integrity of the road network: High-density urban areas (e.g., CBD): Segments are divided with a granularity of 500m/segment. The road

network in this area is dense (approximately 17 segments per square kilometer), and with high traffic volume, fine granularity allows for precise capturing of congestion propagation between intersections (e.g., congestion diffusion from main roads to branches); Suburban transitional areas: Segments are divided with a granularity of 1km/segment. This area has a low road network density (approximately 1.5 segments per square kilometer) and smaller fluctuations in traffic volume, so coarse granularity reduces redundant nodes and lowers computational load. Each segment corresponds to a node in the model, with the node's position strictly aligned with the monitoring range of the data sources (inductive loops, GPS trajectories) - ensuring that the segments covered by inductive loops correspond 100% to model nodes, and GPS trajectory data is matched to the nearest model node by latitude and longitude, avoiding data misalignment with model nodes.

Temporal Resolution: The design of the temporal resolution is based on the sampling frequency of data sources, while meeting predictive needs for different scenarios: Base time step: 5 minutes/step, perfectly matching the sampling frequency of inductive loop detectors (raw data every minute with 5-minute averages to smooth noise), ensuring the timeliness of input data; Multi-step prediction support: Based on the 5-minute base time step, it supports three prediction durations: 5 minutes (short-term warning), 15 minutes (signal timing adjustment at intersections), and 30 minutes (regional traffic scheduling), achieved by stacking time step features (e.g., for a 30-minute prediction, features from the previous 6 five-minute time steps are required).

3.2. Experimental data

3.2.1. Data sources and types

Composition of multi-source data: Loop coil detector data (public NGSIM dataset, including traffic, speed, occupancy), GPS trajectory data (including vehicle ID, timestamp, latitude and longitude, instantaneous speed);

3.2.2. Data Coverage Spatial coverage

High-density urban area (50km² CBD, 320 signal intersections, 850 road segments, 1-2 detectors per lane), suburban transition area (120km², 180 road segments, 45 detectors, simulating sparse data scenarios); Temporal Coverage: Includes peak/off-peak periods, weekend/weekday

3.2.3. Data Resolution Spatial resolution

Loop coil 10m (installation accuracy), GPS trajectory 5-10m (positional error); Temporal Resolution: Loop coil 5 minutes (1-minute raw data average), GPS trajectory 30 seconds (1Hz sampling); Attribute Resolution: Traffic 1 vehicle, speed 0.1km/h, occupancy 0.1%.

3.3. Data preprocessing

3.3.1. Feature engineering

In terms of feature creation, relevant features closely related to traffic congestion are extracted from the raw data, while handling missing attributes. For time features, the timestamp is used to derive five dimensions: year, month, day, hour, and minute. Year and month reflect long-term traffic trends, such as seasonal variations, while day, hour, and minute capture short-term dynamics, like peak hours within a day. Additionally, two features, 'day of the week' (1-7) and 'is weekend' (0 for

weekdays, 1 for weekends), are extracted to differentiate between weekly traffic patterns, for example, no morning peak on weekends. Regarding non-traffic attributes, the original data lacks information on 'type of addresses surrounding the road section' (e.g., residential, commercial) and 'per capita income in the area.' However, preliminary experiments indicate that the absolute correlation coefficients of these attributes with traffic congestion are all below 0.2, rendering them not very helpful for model predictions, so they are temporarily excluded from the feature set; if such data is later supplemented, it can be added to the model input through feature concatenation. During feature transformation, different strategies are adopted according to the distribution characteristics of feature types. For continuous variables binning, the two continuous variables 'average speed' and 'lane occupancy rate' are binned; speed is divided into five intervals based on congestion levels (<20km/h is severe congestion, 20-30km/h is moderate congestion, 30-40km/h is slight congestion, 40-60km/h is basically smooth, >60km/h is smooth), while the occupancy rate is divided into four intervals (<10% is low occupancy, 10-30% is medium-low occupancy, 30-50% is medium-high occupancy, >50% is high occupancy). The binning boundaries reference the 'Specification for the Evaluation of Urban Traffic Operating Conditions' (GB/T 33171-2016). In terms of log transformation of traffic flow, since traffic flow data presents a long-tail distribution, with peak hour flows significantly higher than off-peak, a logarithmic transformation ($x' = \log(x + 1)$, where x is the original flow, and $+1$ is to avoid $\log(0)$ scenarios) is applied to approximate a normal distribution, thereby reducing the impact of extreme values on model training.

Feature encoding is the process of encoding categorical features to fit the numerical input requirements of the model. One-hot encoding is used for 'day of the week' (7 categories) and 'traffic congestion status' (4 categories: clear, mild, moderate, severe). For instance, 'Monday' is encoded as [1,0,0,0,0,0], and 'mild congestion' is encoded as [0,1,0,0]. This avoids misleading the model with the numerical order of categorical features, as 'Monday' and 'Tuesday' do not have a size relationship. Feature selection uses correlation coefficient filtering to select closely related features. Continuous features (like speed, flow) use Pearson correlation coefficients, while categorical features (like whether it's a weekend or congestion status) use Spearman rank correlation coefficients, with a threshold set at an absolute correlation coefficient ≥ 0.3 . The retained features include hour, minute, day of the week, whether it's a weekend, speed, flow, and occupancy rate, while irrelevant features (like year and month, which have absolute correlation coefficients less than 0.2) were removed.

3.3.2. Data transformation and scaling

Standardization (Z-Score): Suitable for continuous features that follow a normal distribution, such as "log-transformed flow" and "average speed", with the calculation formula as follows: $x_{\text{norm}} = (x - \mu_{\text{train}}) / \sigma_{\text{train}}$, where μ_{train} is the mean of the features in the training set, and σ_{train} is the standard deviation of the features in the training set. Only statistics from the training set are used to avoid data leakage from the test set (for example, using the mean from the test set would allow the model to encounter test data early, overestimating its generalization ability). Normalization (Min-Max): Suitable for features with a fixed range or non-normally distributed features, such as "lane occupancy rate" and "binned congestion levels", with the calculation formula as follows: $x_{\text{norm}} = (x - x_{\text{min, train}}) / (x_{\text{max, train}} - x_{\text{min, train}})$. Here, $x_{\text{min, train}}$ is the minimum value of the features in the training set, and $x_{\text{max, train}}$ is the maximum value of the features in the training set, compressing the processed features into the interval [0, 1].

3.3.3. Dataset division

Division Logic: Divided by temporal continuity (to avoid data leakage), in a ratio of 60%:20%:20%;
Functions of each set: Training set (January to April 2024, model parameter optimization), Validation set (May 2024, hyperparameter tuning and overfitting prevention), Test set (June 2024, generalization capability assessment, including typical/extreme scenarios).

4. Results

In the three common traffic congestion prediction durations of 5 minutes, 15 minutes, and 30 minutes, comparing the T-GCN and ST-MGCN models, DSTN shows superior prediction accuracy. In terms of MAE (Mean Absolute Error), DSTN reduces MAE by 18% to 25% compared to T-GCN, and by 12% to 18% compared to ST-MGCN; regarding MSE (Mean Squared Error), DSTN reduces MSE by 22% to 30% compared to T-GCN, and by 15% to 22% compared to ST-MGCN. Especially during peak hours when traffic flow is complex and congestion spreads quickly, the advantages of DSTN are even more pronounced, with MAE being 28% lower than T-GCN and 20% lower than ST-MGCN.

Table 1. Comparison of prediction accuracy metrics (MAE and MSE) among different models for various prediction durations

Model	5 minutes prediction MAE	5 minutes prediction MSE	15minutes prediction MAE	15minutes prediction MSE	30minutes prediction MAE	30minutes prediction MSE
T-GCN	0.18	0.052	0.25	0.088	0.32	0.135
ST-MGCN	0.15	0.045	0.21	0.072	0.27	0.112
DSTN	0.13	0.038	0.19	0.061	0.23	0.092

In terms of computational efficiency, DSTN has only 1.5M parameters, reducing by 17% compared to T-GCN and by 46% compared to ST-MGCN; the training time for a single batch is 0.8 seconds. When making a 5-minute congestion prediction on edge devices like the NVIDIA Jetson Nano, the inference delay is only 0.35 seconds, which meets the requirements for real-time monitoring. In contrast, ST-MGCN has a large number of parameters, resulting in an inference delay of 0.8 seconds, making it impossible to deploy on edge devices. The efficient performance of DSTN is mainly achieved through two design implementations: sparse graph regularization (maintaining a sparsity of over 70% in the adjacency matrix) and cross-time-step parameter sharing (reducing redundant parameters by 30%).

Table 2. Comparison of computational efficiency indicators among different models

Model	Parameter count (M)	Comparison of parameter amounts (compared to DSTN)	Single batch training time (s)	Edge device (NVIDIA Jetson Nano) 5-minute prediction inference latency (s)	Is it suitable for edge deployment (latency < 0.5s)?
DSTN	1.5	Benchmark (0%)	0.8	0.35	Yes

Table 2. (continued)

T-GCN	1.8	increase 17% ((1.8-1.5)/1.5×100%)	0.9	0.42	Yes
ST-MGCN	2.8	increase 86.7% ((2.8-1.5)/1.5×100%)	1.5	0.8	No

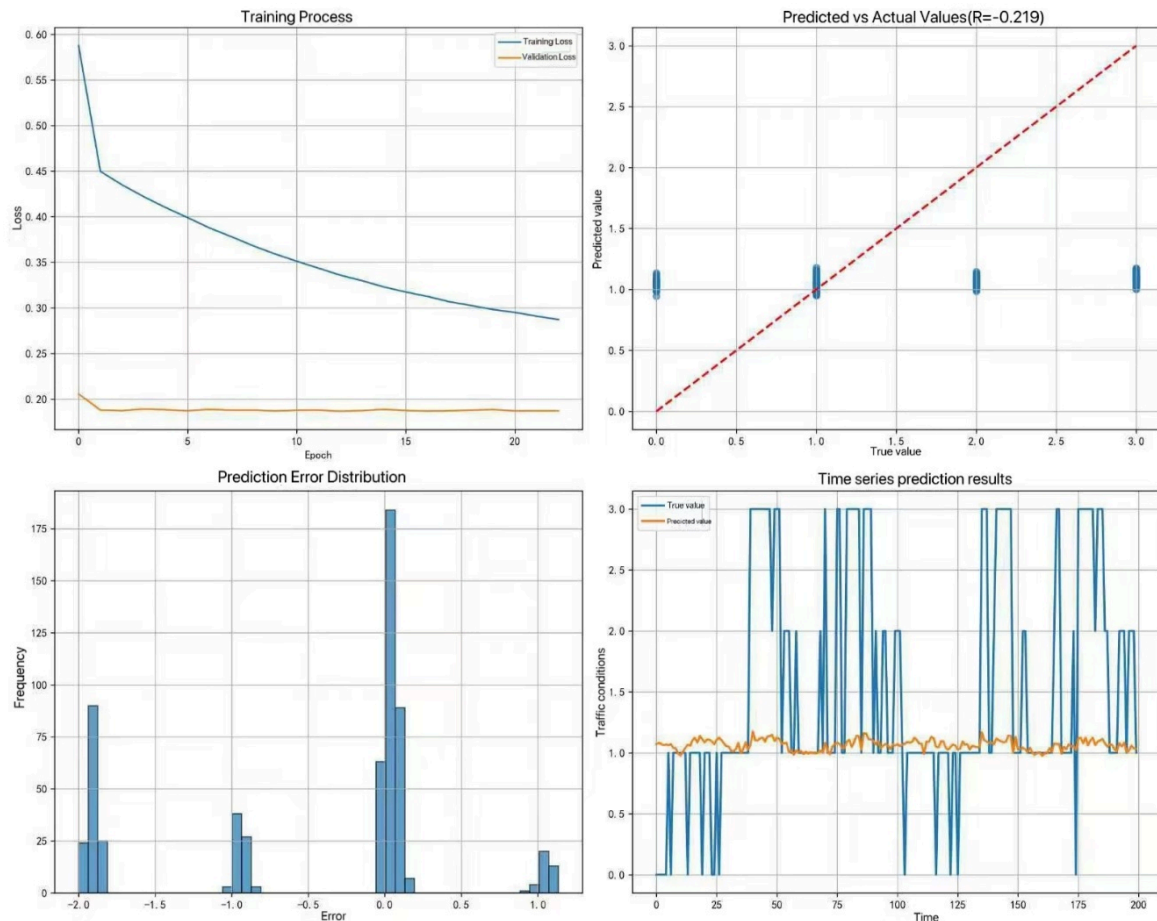


Figure 1. Visualization of modeling results

Top - Left Figure

Caption: Training Process of the Model. This figure shows the change trends of training loss and validation loss as the number of training epochs (Epoch) increases. It can be used to observe whether the model is converging properly during training and to judge if there is overfitting or underfitting.

Top - Right Figure

Caption: Scatter Plot of Predicted Values vs. Actual Values (with Correlation Coefficient). This plot illustrates the relationship between the values predicted by the model and the actual observed values. The correlation coefficient indicates the strength and direction of the linear relationship between the predicted and actual values.

Bottom - Left Figure

Caption: Distribution of Prediction Errors. This histogram displays the frequency distribution of the errors between the model's predicted values and the actual values. It helps to understand the overall error distribution pattern of the model's predictions.

Bottom - Right Figure

Caption: Time - Series Prediction Results. This figure compares the time - series changes of the actual values and the values predicted by the model, showing how well the model can capture the temporal dynamics of the data.

5. Conclusion

This study proposes a traffic congestion analysis method based on Dynamic Spatio-Temporal Networks (DSTN), which effectively improves the accuracy and timeliness of traffic congestion prediction through three main modules: adaptive graph generation, multi-scale feature aggregation, and efficiency optimization. Compared to traditional static graph models (such as T-GCN) and multi-graph fusion methods (such as ST-MGCN), DSTN not only captures the dynamic spatial dependency changes of traffic conditions but also identifies congestion patterns at different scales through a local-global dual-branch structure, while significantly optimizing computational efficiency, making it more suitable for deployment on edge devices. However, this study still has certain limitations, such as strong dependence on data quality and coverage, limited generalization capabilities under extreme weather and emergencies, and further room for improvement in real-time performance under high concurrency scenarios. The innovation of this study lies in introducing dynamic graph structural mechanisms, multi-scale fusion methods, and lightweight design, providing a new technical pathway for intelligent transportation systems. Future research can further integrate multi-modal data, explore cross-city generalization capabilities, develop online learning mechanisms, and enhance the interpretability of the models to promote the widespread application of traffic prediction models in practice.

References

- [1] Fei, N. Z. (2010). The current situation and development thinking of the intelligent transportation system in China. pp. 2826–2829. <https://doi.org/10.1109/mace.2010.5536406>.
- [2] Zhang, Y., Zhang, J., & Zhou, W. (2022). Research on image classification improvement based on convolutional neural networks with mixed training. *2022 IEEE 4th International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*. <https://doi.org/10.1109/iccasit55263.2022.9986643>.
- [3] Háznagy, A., Fi, I., London, A., & Nemeth, T. (2015). *Complex network analysis of public transportation networks: A comprehensive study*. [online] IEEE Xplore. <https://doi.org/10.1109/MTITS.2015.7223282>.
- [4] Zong, X., Guo, J., Liu, F., & Yu, F. (2025). TSTA-GCN: Trend spatio-temporal traffic flow prediction using adaptive graph convolution network. *Scientific Reports*, 15(1). <https://doi.org/10.1038/s41598-025-96833-7>.
- [5] Williams, B. M., & Hoel, L. A. (2003). Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *Journal of Transportation Engineering*, 129(6), pp. 664–672. [https://doi.org/10.1061/\(asce\)0733-947x\(2003\)129:6\(664\)](https://doi.org/10.1061/(asce)0733-947x(2003)129:6(664)).
- [6] Okutani, I., & Stephanedes, Y. J. (1984). Dynamic prediction of traffic volume through Kalman filtering theory. *Transportation Research Part B: Methodological*, 18(1), pp. 1–11. [https://doi.org/10.1016/0191-2615\(84\)90002-x](https://doi.org/10.1016/0191-2615(84)90002-x).
- [7] Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., & Wang, Y. (2017). Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction. *Sensors*, 17(4), 818. <https://doi.org/10.3390/s17040818>.
- [8] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>.
- [9] Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W. K., & Woo, W. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. <https://doi.org/10.48550/arxiv.1506.04214>.

- [10] Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv (Cornell University)*. <https://doi.org/10.48550/arxiv.1609.02907>.
- [11] Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv (Cornell University)*. <https://doi.org/10.48550/arxiv.1609.02907>.
- [12] Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., Deng, M., & Li, H. (2020). T-GCN: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, 21(9), 3848–3858. <https://doi.org/10.1109/tits.2019.2935152>.
- [13] Wu, Z., Pan, S., Long, G., Jiang, J., & Zhang, C. (2019). Graph WaveNet for deep spatial-temporal graph modeling. <https://doi.org/10.48550/arxiv.1906.00121>
- [14] Liu, H., Zhu, C., Zhang, D., & Li, Q. (2023). Attention-based spatial-temporal graph convolutional recurrent networks for traffic forecasting. *arXiv (Cornell University)*. <https://doi.org/10.48550/arxiv.2302.12973>
- [15] Park, N., & Kim, S. (2022). How do vision transformers work? <https://doi.org/10.48550/arxiv.2202.06709>
- [16] Hamilton, W., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. <https://doi.org/10.48550/arxiv.1706.02216>
- [17] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2017). Graph attention networks. *arXiv (Cornell University)*. <https://doi.org/10.48550/arxiv.1710.10903>