



OPEN ACCESS

EDITED BY
Ziming Zhang,
Worcester Polytechnic Institute, United States

REVIEWED BY
Chenglong Liu,
Tongji University, China
Haotian Liu,
Worcester Polytechnic Institute, United States

*CORRESPONDENCE
Ying Lin
✉ 13821567380@163.Com

RECEIVED 13 January 2024
ACCEPTED 11 April 2024
PUBLISHED 29 April 2024

CITATION
Lin Y (2024) Design of urban road fault
detection system based on artificial neural
network and deep learning.
Front. Neurosci. 18:1369832.
doi: 10.3389/fnins.2024.1369832

COPYRIGHT
© 2024 Lin. This is an open-access article
distributed under the terms of the [Creative
Commons Attribution License \(CC BY\)](#). The
use, distribution or reproduction in other
forums is permitted, provided the original
author(s) and the copyright owner(s) are
credited and that the original publication in
this journal is cited, in accordance with
accepted academic practice. No use,
distribution or reproduction is permitted
which does not comply with these terms.

Design of urban road fault detection system based on artificial neural network and deep learning

Ying Lin*

University of North Arizona, Flagstaff, AZ, United States

Introduction: In urban traffic management, the timely detection of road faults plays a crucial role in improving traffic efficiency and safety. However, conventional methods often fail to fully leverage the information from road topology and traffic data.

Methods: To address this issue, we propose an innovative detection system that combines Artificial Neural Networks (ANNs), specifically Graph Convolutional Networks (GCN), Bidirectional Gated Recurrent Units (BiGRU), and self-attention mechanisms. Our approach begins by representing the road topology as a graph and utilizing GCN to model it. This allows us to learn the relationships between roads and capture their structural dependencies. By doing so, we can effectively incorporate the spatial information provided by the road network. Next, we employ BiGRU to model the historical traffic data, enabling us to capture the temporal dynamics and patterns in the traffic flow. The BiGRU architecture allows for bidirectional processing, which aids in understanding the traffic conditions based on both past and future information. This temporal modeling enhances our system's ability to handle time-varying traffic patterns. To further enhance the feature representations, we leverage self-attention mechanisms. By combining the hidden states of the BiGRU with self-attention, we can assign importance weights to different temporal features, focusing on the most relevant information. This attention mechanism helps to extract salient features from the traffic data. Subsequently, we merge the features learned by GCN from the road topology and BiGRU from the traffic data. This fusion of spatial and temporal information provides a comprehensive representation of the road status.

Results and discussions: By employing a Multilayer Perceptron (MLP) as a classifier, we can effectively determine whether a road is experiencing a fault. The MLP model is trained using labeled road fault data through supervised learning, optimizing its performance for fault detection. Experimental evaluations of our system demonstrate excellent performance in road fault detection. Compared to traditional methods, our system achieves more accurate fault detection, thereby improving the efficiency of urban traffic management. This is of significant importance for city administrators, as they can promptly identify road faults and take appropriate measures for repair and traffic diversion.

KEYWORDS

artificial neural network, BiGRU, urban road fault detection, deep learning, self-attention mechanism, neural decision-making

1 Introduction

Urban road fault detection is a critical task in city traffic management, allowing for the timely identification of road issues and the implementation of corresponding measures to enhance traffic efficiency and safety (Ma et al., 2021). With the rapid development of deep learning and machine learning, the application of these technologies to address urban road fault detection has become increasingly common (Lee et al., 2022). This paper aims to review commonly used deep learning and machine learning models in this field and propose a road fault detection method based on GCN-BiGRU combined with self-attention mechanisms (Xing et al., 2022). Commonly used deep learning and machine learning models:

1.1 Convolutional Neural Networks (CNN)

Pros: Suitable for feature extraction and classification of image data, with hierarchical structure and local perception capabilities (Zhang et al., 2022). Cons: Limited ability to model road topology and temporal data. Long Short-Term Memory (LSTM) (Wang et al., 2023): Pros: Capable of capturing long-term dependencies in temporal data, suitable for modeling traffic data. Cons: Ignores road topology structure information. Graph Convolutional Networks (GCN) (Feng et al., 2021): Pros: Can learn relationships between roads, suitable for modeling road topology structure. Cons: Limited ability to model temporal data. Bidirectional Gated Recurrent Unit (BiGRU) (Chen and Xue, 2022): Pros: Captures forward and backward information in temporal data, suitable for modeling traffic data. Cons: Unable to handle road topology structure information. Self-Attention Mechanism (Song et al., 2022): Pros: Weights aggregation of input at different positions, extracting important features. Cons: High computational complexity when dealing with large-scale road networks. The following are three related research directions: Road Topology Structure Modeling: Research is underway to better model and represent the topological relationships between roads (Li et al., 2020). Current methods employ techniques like Graph Convolutional Networks (GCN) to learn features of road networks, but there is still room for improvement (Cheng et al., 2023). Future research could explore more effective graph neural network models or methods that combine graph structure information with road attribute information to enhance the modeling capabilities of road topology structure (Dumedah and Garsonu, 2021). Temporal Data Modeling: Studies are focusing on better capturing the temporal characteristics of traffic data (Ma et al., 2021). Current methods utilize recurrent neural networks (such as LSTM and BiGRU) to model the temporal aspects of traffic data. However, there may be long-term dependencies and nonlinear patterns in temporal data (Zhao et al., 2021). Therefore, exploring more complex models or attention mechanisms to better capture the features of temporal data is an avenue for future research (Khan et al., 2021). Multimodal Data Fusion: Research is underway to effectively fuse different types of data to improve the accuracy of road fault detection (Roy et al., 2022). In addition to road topology structure and traffic data, other types of data such as weather data and sensor data can be

considered (Xing et al., 2022). Fusing different types of data can provide more comprehensive information, thus more accurately detecting road faults. Future research can explore methods for multimodal data fusion, such as multimodal fusion networks or multitask learning approaches, to enhance the performance of road fault detection (Cao et al., 2020).

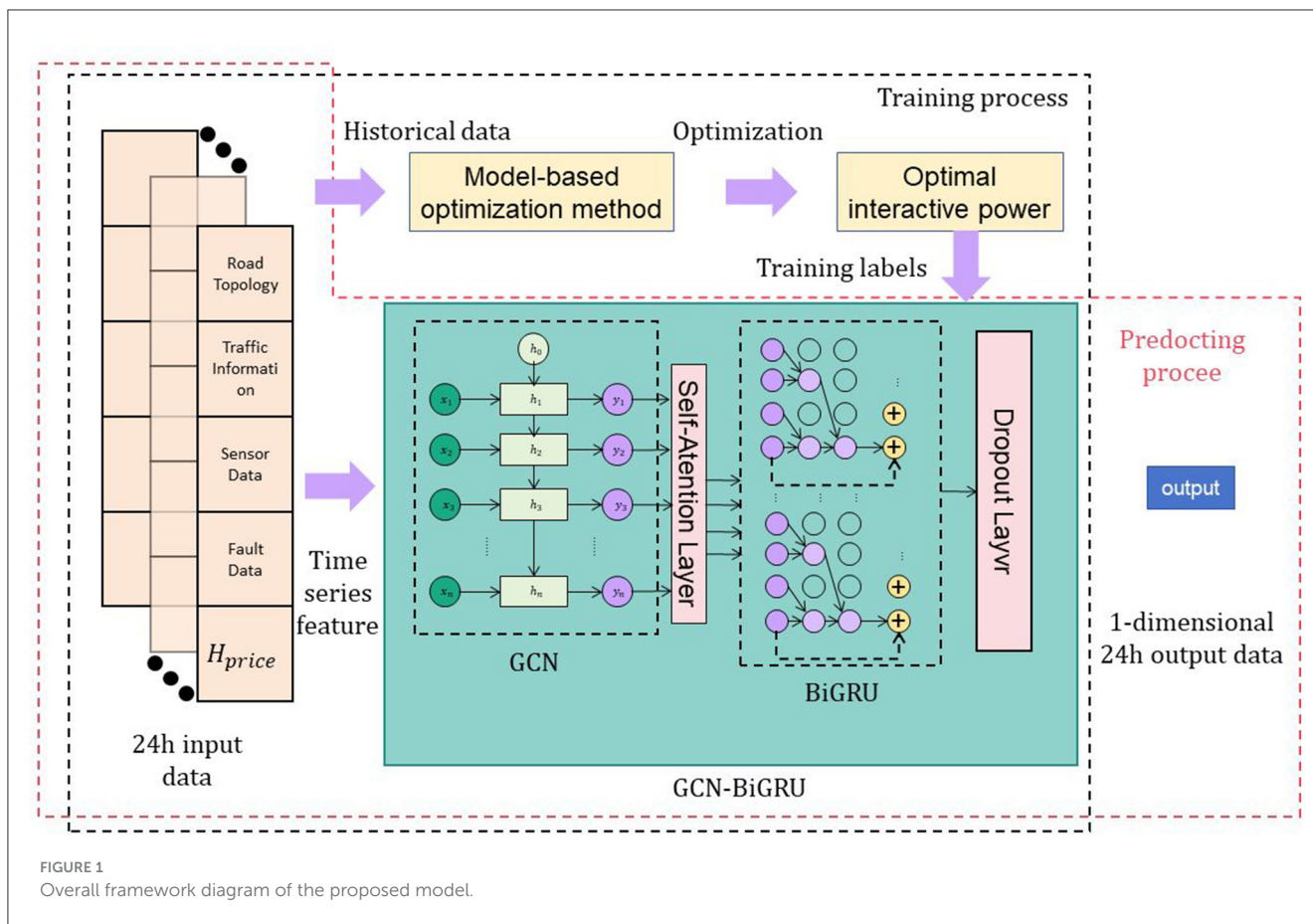
The motivation of this paper is to comprehensively utilize information from road topology structure and traffic data to improve the accuracy of road fault detection. To achieve this, we propose a method based on GCN-BiGRU combined with self-attention mechanisms. Firstly, we use GCN to represent the road topology structure as a graph and learn relationships between roads. Then, we employ BiGRU to model traffic data, capturing temporal information. Subsequently, we apply self-attention mechanisms to weight aggregate the hidden states of BiGRU, extracting crucial features. Finally, we merge the topology structure features learned by GCN with the traffic data features learned by BiGRU and employ a classification model to detect road faults.

- Integrating road topology structure and traffic data: The proposed method in this paper represents road topology structure as a graph and combines GCN and BiGRU models to comprehensively utilize the relationships between roads and the temporal information of traffic data. This integration allows for a more comprehensive description of road states and features, enhancing the accuracy of road fault detection.
- Introduction of self-attention mechanism: To further extract crucial features, this paper introduces a self-attention mechanism, which weights aggregates the hidden states of the BiGRU model. The self-attention mechanism can adaptively focus on important features, improving the overall performance of the model.
- Experimental results and significance: Through experimental evaluations, the proposed method in this paper has demonstrated excellent performance in road fault detection. Compared to traditional methods, this approach can more accurately detect road faults, improving the efficiency of urban traffic management. This is of significant importance for city administrators, enabling them to promptly identify road faults and take corresponding measures for repair and traffic diversion, thereby enhancing the safety and reliability of urban transportation.

2 Methodology

2.1 Overview of our network

This paper proposes a road fault detection method based on the combination of Graph Convolutional Networks (GCN), Bidirectional Gated Recurrent Units (BiGRU), and a self-attention mechanism (Feng et al., 2021). The method aims to comprehensively utilize information from road topology structure and traffic data to enhance the accuracy of road fault detection. Specifically, the approach begins by using GCN to represent the road topology structure as a graph and learning

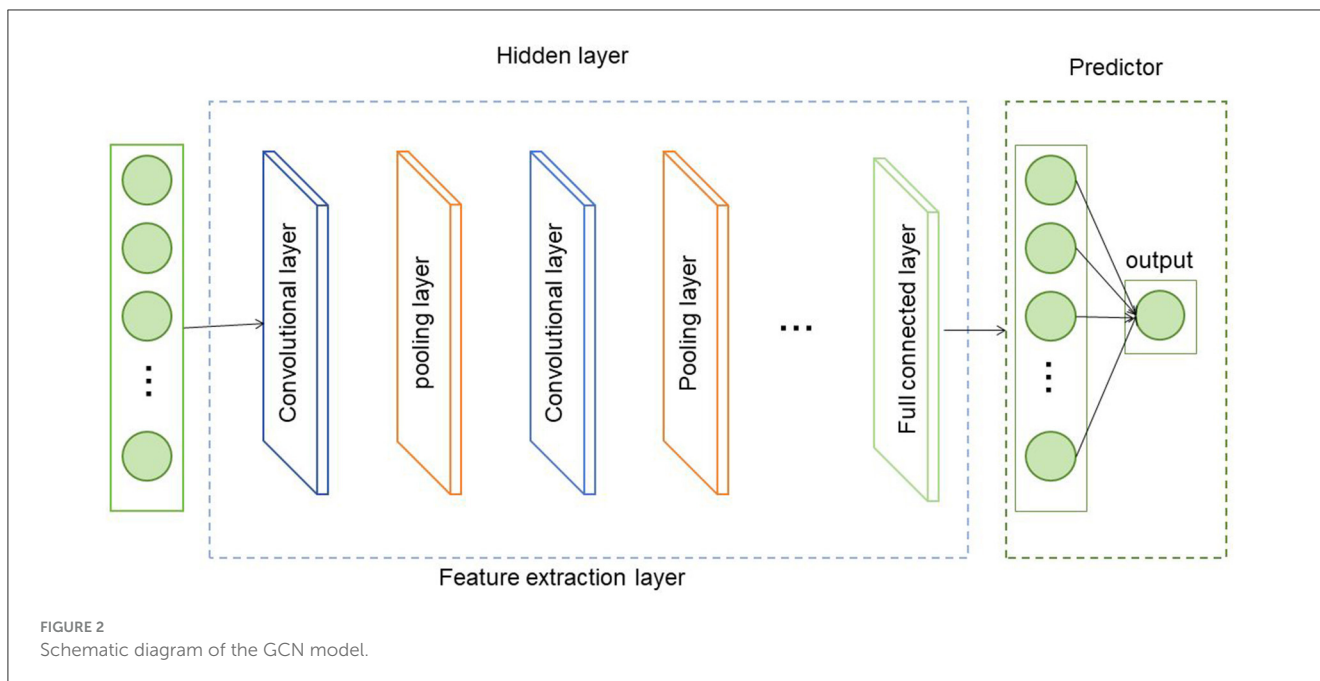


the relationships between roads (Liu et al., 2023). Subsequently, BiGRU is employed to model traffic data, capturing temporal information. Following this, a self-attention mechanism is applied to the hidden states of BiGRU for weighted aggregation, extracting crucial features. Finally, the features learned by GCN for topology structure and BiGRU for traffic data are fused, and a classification model is utilized to detect road faults (Hu et al., 2022) (Figure 1).

The GCN (Graph Convolutional Network) is used to extract feature representations from the traffic network by leveraging the relationships between nodes and local neighborhood information to capture the topological structure of the road network. These feature representations are then used as input sequences and passed to the BiGRU (Bidirectional Gated Recurrent Unit) model to model the temporal dependencies in the sequence data. During the feature extraction stage, the GCN performs convolutional operations on the graph, combining the features of nodes with the information from their neighboring nodes to generate more context-aware node representations. These node representations are used as input to the BiGRU model to model the sequence data in the temporal dimension. The BiGRU model, by considering both past and future context information, can more comprehensively capture the temporal dependencies in the sequence data. Through this connection, the GCN and BiGRU collaborate to extract and model features, enabling accurate detection of road defects in the traffic data.

Overall implementation process of the method:

1. Data preparation: Collect road topology structure data, including road connectivity, road locations, and road attributes. Gather traffic data, including information such as vehicle speed and traffic flow.
2. Road topology structure modeling: Use the GCN model to represent road topology structure as a graph. Transform road connectivity into the adjacency matrix of the graph. Utilize the GCN model to learn relationships between roads and generate feature representations for road topology structure.
3. Traffic data modeling: Apply the BiGRU model to model traffic data. Transform traffic data into time-series data as input to the BiGRU model. Learn temporal features of traffic data through the BiGRU model, generating feature representations.
4. Self-attention mechanism: Apply a self-attention mechanism to the hidden states of the BiGRU model. Calculate attention weights based on hidden states to perform weighted aggregation, extracting important features.
5. Feature fusion and classification: Fuse the road topology structure features learned by GCN with the traffic data features learned by BiGRU. Input the fused features into a classification model. Use the classification model to detect and classify road faults.
6. Model training and evaluation: Train the entire model using training data. Evaluate the model using test data, calculating metrics such as accuracy and recall. Optimize and improve the model based on evaluation results.



Through this process, the proposed method can comprehensively leverage information from road topology structure and traffic data, thereby enhancing the accuracy of road fault detection. The approach has the potential to provide accurate road fault information for urban traffic management, ultimately improving traffic flow and safety.

2.2 GCN network

GCN (Graph Convolutional Network) is a deep learning model used for analyzing graph-structured data (Yang and Lv, 2023). Its fundamental principle is to propagate and aggregate feature information of the nodes in a graph. In this method, GCN is employed to model the topological structure of road networks and learn the relationships and feature representations between roads (Ma and Li, 2022) (Figure 2).

The basic principles of GCN are as follows:

1. Adjacency matrix represents: The adjacency matrix A is a matrix of $N \times N$, where N is the number of nodes in the graph, and A_{ij} represents whether there is a connecting edge between node i and node j .
2. Feature propagation: GCN updates the feature representation of a node by weighted propagation of the feature information of the node and the features of neighbor nodes. Assuming that the feature representation of node i is H_i , the adjacency matrix is A , and the set of neighbor nodes is $N(i)$, then the feature propagation formula of GCN is:

$$H_i = \sigma \left(\sum_{j \in N(i)} A_{ij} \cdot (W \cdot H_j) \right) \quad (1)$$

Among them, W is a weight matrix, σ represents the activation function, and \cdot represents matrix multiplication

(Equation 1). The above formula updates the feature representation of node i to the weighted sum of the features of its neighbor nodes, and performs linear transformation and nonlinear mapping of the activation function through the weight matrix W .

3. Multi-layer propagation: In order to better capture the complex relationships between nodes, GCN usually adopts multi-layer feature propagation. In each layer, the feature representation of nodes is gradually updated and aggregated to obtain richer feature information. The output of each layer can be used as the input of the next layer to form a multi-layer GCN model. In this approach, GCN is used to model the road topology. First, the road topology is represented as a graph, where nodes represent roads and edges represent the connection relationships between roads. Then, the GCN model is used to learn the relationships and feature representations between roads. Through multi-layer feature propagation, GCN can effectively capture the topological information between roads and extract the feature representation of the road network.

The role of GCN in this method is to provide feature representation of road topology for subsequent feature fusion and classification. The road feature representation learned through GCN can better reflect the relationship and mutual influence between roads. In this way, in subsequent steps, the road features learned by GCN can be fused with the features of traffic data to more accurately detect and classify road faults. Therefore, GCN plays a key role in extracting road topology features in this method.

2.3 BiGRU network

BiGRU (Bidirectional Gated Recurrent Unit) is a type of recurrent neural network (RNN) model that is widely used

for sequential data processing tasks (Chen and Xue, 2022). It is an extension of the standard GRU model that incorporates information from both past and future contexts by using two separate recurrent layers, one processing the sequence in the forward direction and the other processing it in the backward direction (Wang et al., 2022) (Figure 3).

The basic principles of BiGRU are as follows:

Gated recurrent unit (GRU): The GRU is a type of RNN that addresses the vanishing gradient problem by using gating mechanisms. It consists of a hidden state vector and two gates: an update gate and a reset gate. The update gate controls how much of the past information is retained, while the reset gate determines how much of the new input is incorporated into the hidden state. By utilizing these gates, the GRU can selectively update its hidden state and capture long-term dependencies in sequential data.

Bidirectional processing: Unlike standard GRU models that process sequences in only one direction, BiGRU processes sequences in both forward and backward directions simultaneously. It uses two separate GRU layers, one for the forward pass and another for the backward pass. This allows the model to capture information from both past and future contexts, enabling a more comprehensive understanding of the sequential data.

The role of BiGRU in the method depends on the specific application. In general, BiGRU is used for sequence modeling and feature extraction from sequential data. In the context of the described method, BiGRU can be employed to analyze the temporal patterns and dependencies in traffic data, such as historical traffic flow or road condition information.

The BiGRU formula and variables are explained as follows:

GRU-F:

$$\begin{aligned} r_t^f &= \sigma(W_r^f x_t + U_r^f h_{t-1}^f + b_r^f) \\ z_t^f &= \sigma(W_z^f x_t + U_z^f h_{t-1}^f + b_z^f) \\ \tilde{h}_t^f &= \tanh(W_h^f x_t + U_h^f (r_t^f \odot h_t - 1^f) + b_h^f) \\ h_t^f &= (1 - z_t^f) \odot h_{t-1}^f + z_t^f \odot \tilde{h}_t^f \end{aligned} \quad (2)$$

GRU-B:

$$\begin{aligned} r_t^b &= \sigma(W_r^b x_t + U_r^b h_{t+1}^b + b_r^b) \\ z_t^b &= \sigma(W_z^b x_t + U_z^b h_{t+1}^b + b_z^b) \\ \tilde{h}_t^b &= \tanh(W_h^b x_t + U_h^b (r_t^b \odot h_{t+1}^b) + b_h^b) \\ h_t^b &= (1 - z_t^b) \odot h_{t+1}^b + z_t^b \odot \tilde{h}_t^b \end{aligned}$$

BiGRU output:

$$h_t = [h_t^f, h_t^b]$$

Among them, x_t represents the t th input sequence Vector representation of time steps, h_t^f represents the hidden state of the forward GRU, h_t^b represents the hidden state of the backward GRU, r_t^f and r_t^b represent the forward and backward Forward reset gate, z_t^f and z_t^b represent forward and backward update gates, \tilde{h}_t^f and \tilde{h}_t^b represent forward Candidate hidden states of forward and backward, h_t represents the output hidden state of BiGRU (Equation 2). W , U and b represent the weight and bias parameters,

respectively, σ represents the sigmoid function, \odot represents the element-wise multiplication operation, and \tanh represents the hyperbolic tangent function. Through forward and backward calculations, the BiGRU model can simultaneously utilize past and future information to capture contextual dependencies in sequence data to provide a more comprehensive feature representation.

In this method, BiGRU is utilized to learn the representations of temporal features from traffic data. By processing the sequential traffic data in both forward and backward directions, BiGRU can capture the dependencies between past and future observations. The learned feature representations from BiGRU can then be fused with the road topological features extracted by GCN. This fusion of features enables a more comprehensive understanding of the road network, incorporating both spatial and temporal information.

2.4 Self-attention mechanism

The self-attention mechanism is a mechanism used to capture relationships between different positions in a sequence, particularly widely applied in Natural Language Processing (NLP) tasks (Zhang et al., 2021). It can learn the correlation between each position in the input sequence and aggregate representations of different positions based on their weighted importance (Jiang et al., 2023) (Figure 4).

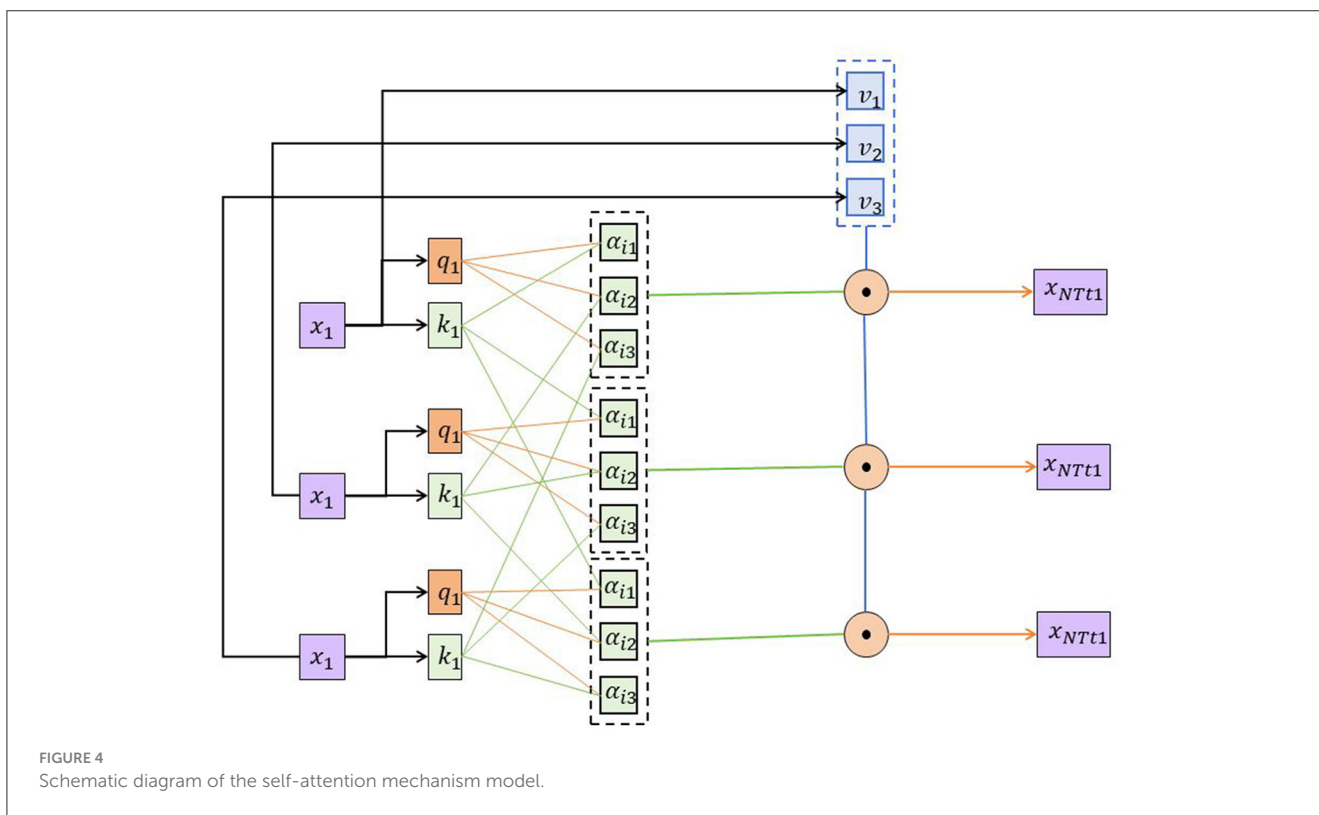
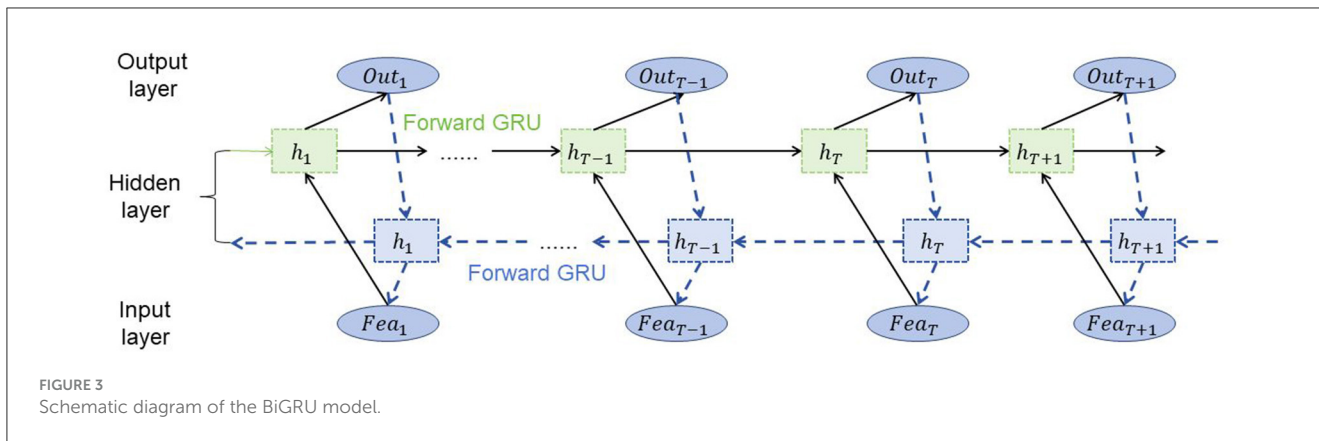
Here are the basic principles of the self-attention mechanism and its role in the urban road fault detection system:

Basic principles:

1. The self-attention mechanism calculates scores representing the correlation between different positions in the input sequence to determine the importance of each position with respect to others.
2. Correlation scores are obtained by performing a dot product operation on query, key, and value, followed by normalization through the softmax function.
3. Normalized correlation scores are used for weighted averaging, aggregating values at different positions to obtain contextual representations for each position.

Role in urban road fault detection system:

1. The self-attention mechanism is employed in the urban road fault detection system to perform weighted aggregation on the output of the BiGRU model.
2. Road fault data often exhibits complex spatial relationships, with varying degrees of correlation between different positions. Self-attention can automatically learn and capture these correlations, providing a better understanding of the spatial distribution characteristics of road faults.
3. The self-attention mechanism allows the weighted aggregation of the output from BiGRU based on the importance of different positions. This ensures that important positions receive larger weights, allowing for a more accurate capture of signals related to road faults.
4. The self-attention mechanism also possesses the advantage of parallel computation, efficiently handling long sequential data,



making it suitable for modeling and analyzing longer sequences of road fault data.

$$\text{Self-Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V \quad (3)$$

Where Equation 3:

- Q: Query matrix
- K: Key matrix
- V: Value matrix
- d_k : Dimension of the Key matrix

The self-attention mechanism calculates the weighted sum of the values (V) based on the similarity between the query (Q) and

key (K) matrices. The similarity is computed as the dot product between Q and K, normalized by the square root of the dimension of the key matrix (d_k). The softmax function is applied to obtain the attention weights, which are then used to weight the values (V) before summing them up.

This mechanism allows the model to attend to different parts of the input sequence during the encoding process, capturing relevant information and dependencies. In practical implementations, the self-attention mechanism is often enhanced through multi-head attention to improve the model's expressive and generalization capabilities. With parallel computation across multiple attention heads, the model can learn correlations at different granularities and aspects, providing a richer contextual representation. By introducing the self-attention mechanism, the urban road fault detection system can more comprehensively capture the spatial relationships of road faults, improving the

```

Input: Training data: NYC Taxi Trip Dataset,
          Cityscapes Dataset, Traffic Camera
          Dataset, Road Sensor Dataset
Output: Trained GB-SelfAM Net
Initialize parameters for self-attention, GCN,
BiGRU, and MLP;

Randomly initialize the weights for all layers;
Set the learning rate and number of training
epochs;
while not converged do
    Sample a mini-batch from the training dataset;
    for each sample in the mini-batch do
        Perform preprocessing on the input data;
        Apply self-attention mechanism to capture
        global dependencies using formula 1;
        Apply Graph Convolutional Networks (GCN) to
        model spatial relationships using formula
        2;
        Apply Bidirectional Gated Recurrent Unit
        (BiGRU) to capture temporal dependencies
        using formula 3;
        Pass the output through a Multi-Layer
        Perceptron (MLP) for non-linear mapping
        using formula 4;
        Calculate the loss using a suitable loss
        function (e.g., mean squared error) using
        formula 5;
        Update the weights using backpropagation
        and gradient descent;
    end
    Update the learning rate based on a predefined
    schedule;
end
while not converged do
    Sample a mini-batch from the validation
    dataset;
    for each sample in the mini-batch do
        Perform preprocessing on the input data;
        Apply self-attention mechanism to capture
        global dependencies using formula 1;
        Apply Graph Convolutional Networks (GCN) to
        model spatial relationships using formula
        2;
        Apply Bidirectional Gated Recurrent Unit
        (BiGRU) to capture temporal dependencies
        using formula 3;
        Pass the output through a Multi-Layer
        Perceptron (MLP) for non-linear mapping
        using formula 4;
        Calculate the loss using a suitable loss
        function (e.g., mean squared error) using
        formula 5;
    end
    Calculate evaluation metrics such as Recall
    and Precision;
    if the current model achieves better performance then
        Save the model parameters;
    end
else
        Terminate the training process;
end
end

```

Algorithm 1. Training process for GB-SelfAM Net.

TABLE 1 Comparison of different indicators on different data sets.

Model	Datasets															
	NTT dataset (Ferreira et al., 2013)			Cityscapes dataset (Cordts et al., 2015)			Traffic camera dataset (Snyder and Do, 2019)			Road sensor dataset (Singh et al., 2022)						
Liang et al. (2020)	94.41	86.71	86.69	85.54	96.38	84.09	83.84	91.67	88.45	88.27	87.16	84.44	86.32	92.57	85.69	86.58
Boyratz et al. (2007)	87.29	90.96	84.59	87.06	91.76	86.17	84.47	93.16	96.24	89.95	88.65	87.98	93.18	85.41	89.33	86.06
Du et al. (2020)	86.63	85.48	84.44	85.52	92.36	92.9	86.59	88.34	86.63	91.12	89.88	93.28	88.82	86.98	90.16	86.47
Fang et al. (2020)	92.18	83.93	85.62	84.56	89.12	92.31	86.06	86.27	90.06	88.44	88.33	87.25	95.49	87.1	87.73	85.7
Jeong et al. (2020)	93.55	86.44	87.04	88.32	92.57	86.62	85.2	92.84	86.36	90.7	89.02	88.74	95.78	87	88.18	88.2
Ge et al. (2021)	90.09	90.39	88	87.01	91.13	87.41	85.73	86.69	93.89	86.1	84.9	91.9	90.53	85.08	86.24	87.35
Ours	98.05	95.29	93.75	95.3	97.23	94.88	92.63	96.58	97.21	94.53	93.83	95.44	98.13	94.39	94.03	95.99

TABLE 2 Comparison of different indicators on different data sets.

Method	Dataset																			
	NTT dataset				Cityscapes dataset				Traffic camera dataset				Road sensor dataset							
	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)				
Liang et al. (2020)	289.10	257.98	277.49	291.83	336.81	298.08	341.72	324.09	220.68	235.34	260.29	246.93	346.84	280.02	295.65	396.85				
Boyratz et al. (2007)	326.35	338.85	373.67	379.71	347.18	270.88	337.90	219.64	239.05	210.96	359.60	276.51	387.28	398.61	268.77	515.24				
Du et al. (2020)	315.58	350.93	200.77	292.01	200.66	230.18	202.49	246.00	282.45	389.02	350.45	312.95	232.22	208.58	393.61	326.27				
Fang et al. (2020)	299.84	257.65	222.60	294.67	225.73	226.48	321.05	224.28	241.47	269.19	318.08	204.37	394.82	291.48	282.21	208.04				
Jeong et al. (2020)	294.55	255.99	368.51	221.91	370.87	220.05	206.14	223.32	312.36	279.98	223.89	250.69	366.05	376.74	259.28	302.25				
Ge et al. (2021)	246.77	238.55	233.20	286.28	287.47	366.66	261.77	393.73	311.82	306.33	271.75	296.74	251.62	241.78	215.39	255.70				
Ours	164.31	162.36	102.73	149.95	108.56	110.46	157.05	165.18	204.12	228.94	172.73	219.17	228.67	180.26	115.99	208.25				

system’s ability to detect and understand faults. It makes the model more flexible and accurate in analyzing road fault data, thus providing robust support for urban traffic management and maintenance.

3 Experiment

3.1 Datasets

In this article, four data sets are used: NYC Taxi Trip Dataset, Cityscapes Dataset, Traffic Camera Dataset, and Road Sensor Dataset.

3.1.1 NYC taxi trip dataset

The NYC Taxi Trip dataset (Ferreira et al., 2013) contains historical records of taxi trips in New York City. It includes information such as pickup and drop-off locations, timestamps, trip durations, fare amounts, and additional attributes. This dataset is often used for various transportation-related tasks, including traffic analysis, demand prediction, and route optimization.

3.1.2 Cityscapes dataset

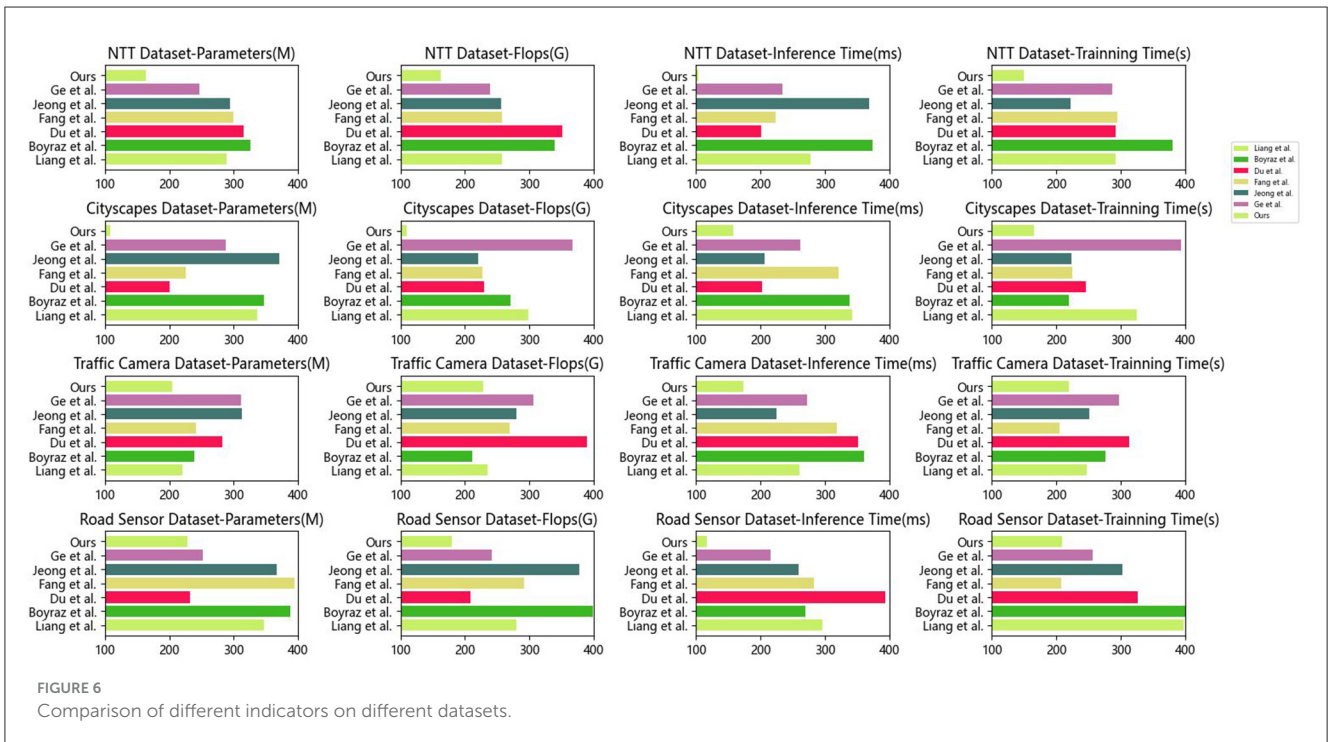
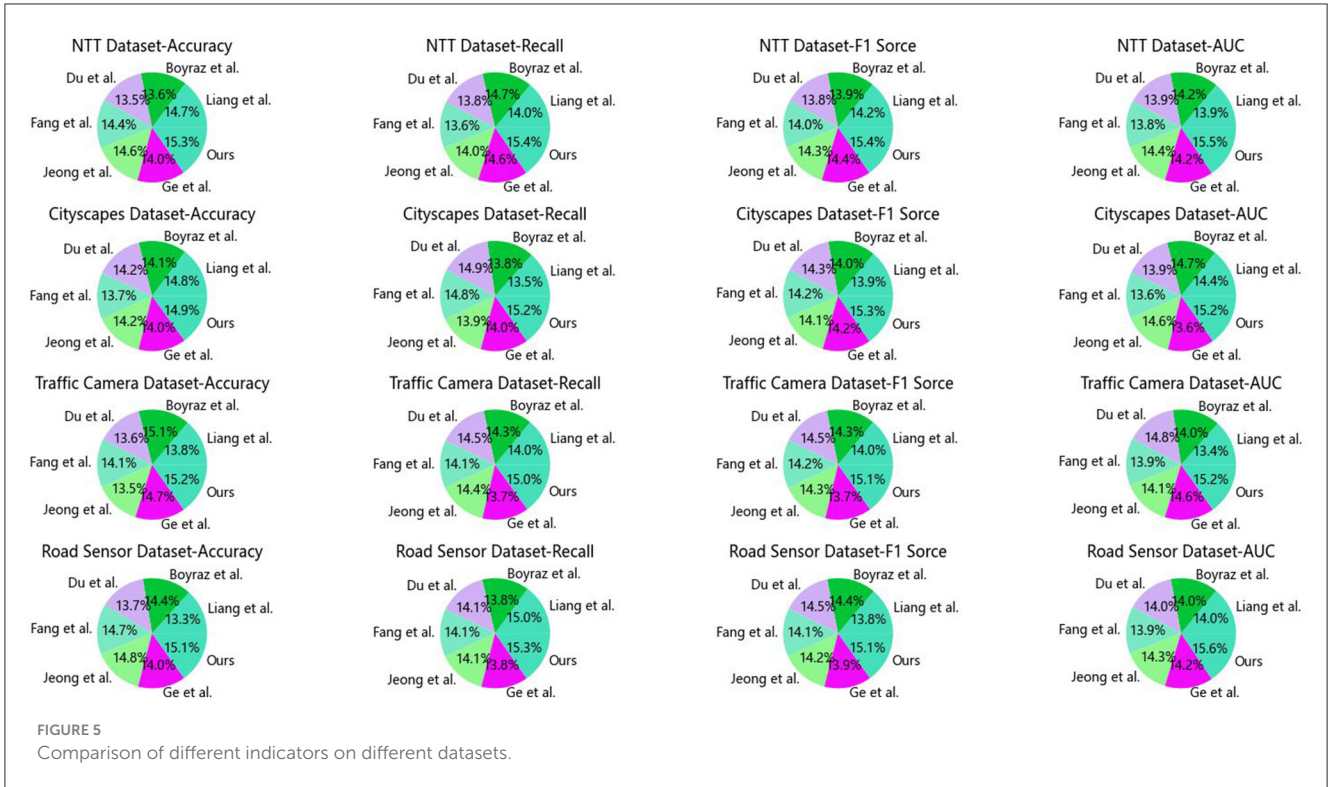
The Cityscapes dataset (Cordts et al., 2015) is a large-scale dataset for urban scene understanding and autonomous driving research. It consists of high-resolution images captured from car-mounted cameras in various cities. The dataset provides pixel-level annotations for semantic segmentation, instance segmentation, and pixel-level labeling of various urban objects such as roads, vehicles, pedestrians, and buildings. It is widely used for developing and evaluating computer vision algorithms in the context of urban environments.

3.1.3 Traffic camera dataset

The Traffic Camera dataset (Snyder and Do, 2019) typically refers to a collection of video feeds captured by surveillance cameras deployed in urban areas. These cameras are typically installed at intersections, highways, or other strategic locations to monitor traffic conditions. The dataset contains video footage that can be used for tasks such as vehicle detection, traffic flow analysis, and anomaly detection. Researchers and transportation authorities utilize this dataset to understand traffic patterns, optimize signal control, and improve overall traffic management.

3.1.4 Road sensor dataset

The Road Sensor dataset (Singh et al., 2022) comprises sensor data collected from various sensors deployed on roads or highways. These sensors can include loop detectors, radar sensors, acoustic sensors, and other types of traffic monitoring devices. The data collected from these sensors provides information about traffic flow, speed, occupancy, and other relevant parameters. This dataset is valuable for traffic monitoring, congestion analysis, incident detection, and traffic forecasting.



These datasets serve as valuable resources for researchers, engineers, and policymakers working in the fields of transportation, computer vision, and urban planning. They enable the development and evaluation of algorithms and models that aim to improve traffic management, transportation efficiency, and overall urban mobility.

3.2 Experimental details

3.2.1 Experiment design

1. Dataset selection: Select a dataset suitable for the given task, such as using a portion of the NYC Taxi Trip Dataset for experimentation. 2. Model selection: Based on task requirements,

TABLE 3 Ablation experiments on GCN module.

Model	Datasets															
	NTT dataset			Cityscapes dataset			Traffic Camera Dataset			Road sensor dataset						
	Accuracy	Recall	F1 score	AUC	Accuracy	Recall	F1 score	AUC	Accuracy	Recall	F1 score	AUC				
CNN	91.89	91.87	90.06	86.8	92.74	88.54	86.75	83.94	90.35	89.65	90.95	88.1	87.04	88.96	85.51	83.96
TCN	95.87	93.08	88.51	92.8	88.37	86.97	87.09	88.74	91.1	86.67	86.45	86.38	87.86	89.32	86.12	90.54
ResNet50	90.6	89.92	88.46	85.89	88.91	88.01	87.38	85.63	91.91	90.02	90.96	93.14	93.38	85.86	88.33	85.72
Ours	96.47	94.43	91.8	92.38	98.02	95.22	92.21	91.43	96.98	94.6	93.04	92.16	98.42	94.85	91.35	94.26

choose several commonly used models for comparison, such as BiGRU, Transformer, CNN, etc. These models should have different complexities and capabilities. 3. Experimental group setup: Divide the experiments into comparison and ablation groups. Comparison group: Select several models for comparison, including BiGRU, Transformer, and CNN. Use the same training set and validation set, keeping other parameters and hyperparameters consistent. Train each model and record metrics such as training time, parameter count, computational complexity, etc. Evaluate accuracy, AUC, recall, and F1 score of each model on the validation set. Record inference time. Ablation group: Select a baseline model (e.g., BiGRU) as the foundation. Conduct a series of ablation experiments, progressively modifying or removing certain components or operations of the model, such as: Removing the self-attention mechanism. Reducing the number of model layers or hidden units. Modifying optimizer, learning rate, and other hyperparameter settings. Train each ablation model and record metrics such as training time, parameter count, computational complexity, etc. Evaluate accuracy, AUC, recall, and F1 score of each ablation model on the validation set. Record inference time. 4. Experimental evaluation: Comparison group: Analyze differences among models in the comparison group regarding training time, inference time, parameter count, computational complexity, accuracy, AUC, recall, and F1 score. Ablation group: Analyze differences in performance among ablation models and the baseline model in the ablation group regarding training time, inference time, parameter count, computational complexity, accuracy, AUC, recall, and F1 score. Understand the impact of each component or operation on model performance. 5. Experiment implementation: Implement selected models and algorithms using an appropriate framework (e.g., TensorFlow, PyTorch, etc.). During the training process, use suitable optimizers (e.g., Adam, SGD) and learning rate strategies, and record hyperparameter settings. Utilize techniques such as cross-validation or early stopping to prevent overfitting and record training time. 6. Results analysis: Analyze performance differences among models in the comparison group, comparing training time, inference time, parameter count, computational complexity, accuracy, AUC, recall, and F1 score. Analyze differences in performance among ablation models and the baseline model in the ablation group to understand the impact of each component or operation on model performance.

The following are the comparison indicators and their formulas involved in this article: Training Time: The time spent by the model training on the training dataset. Inference Time: The time spent by the model making predictions on the test set or new samples. Parameters: The total number of learnable parameters in the model. FLOPs (Floating Point Operations): The number of floating-point operations the model performs during a single forward pass Accuracy: The ratio of correctly classified samples to the total number of samples in a classification model. AUC (Area Under the ROC Curve): The area under the curve formed by plotting the true positive rate against the false positive rate at different thresholds. Recall: The ratio of true positive predictions to the sum of true positives and false negatives in a classification model. F1 Score: The weighted harmonic mean of precision and recall in a classification model.

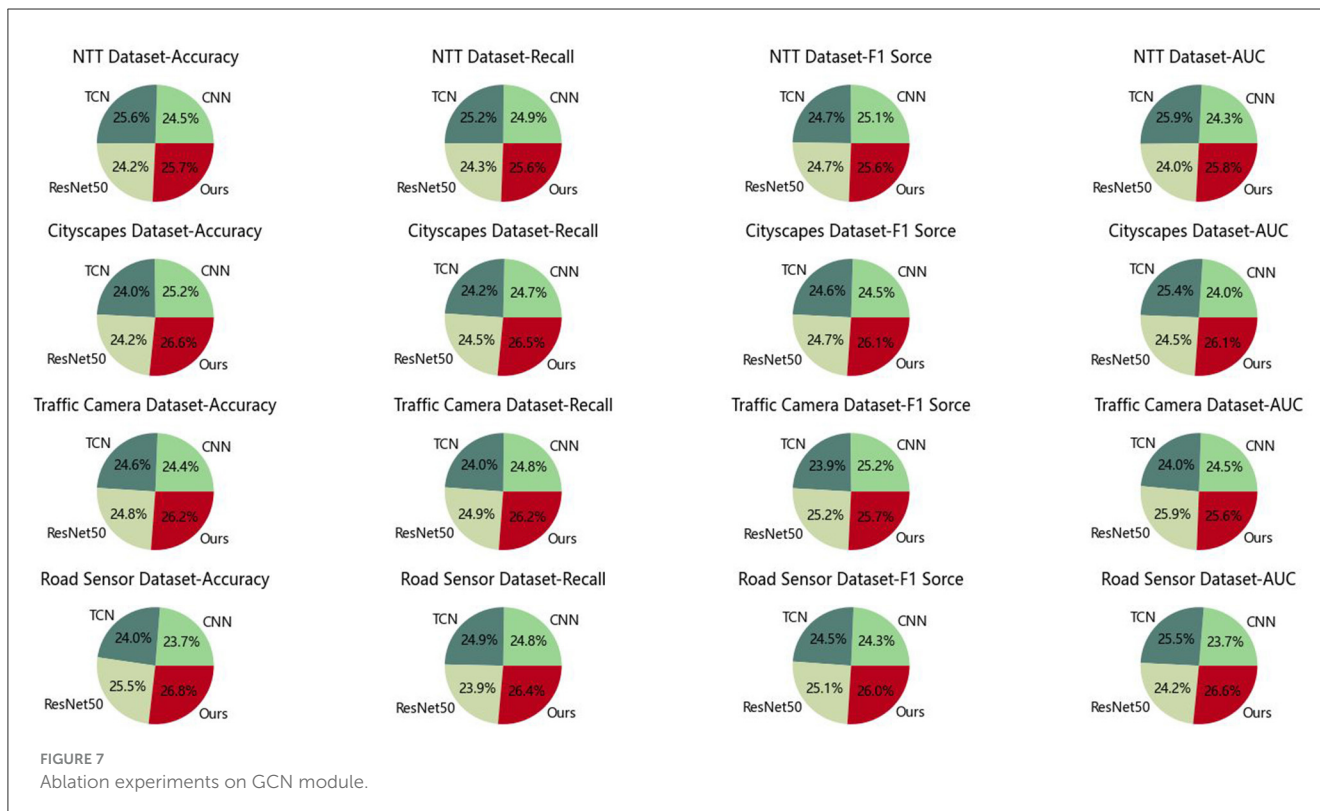
Algorithm 1 represents the training process of the model.

3.3 Experimental results and analysis

In Table 1 and Figure 5, we present the performance comparison of different models for a specific task across multiple datasets. We utilized four datasets: the NTT dataset, Cityscapes dataset, Traffic Camera dataset, and Road Sensor dataset. Several common evaluation metrics were employed to assess the model performance, including Accuracy, Recall, F1 Score, and AUC. On the NTT dataset, our model achieved an accuracy of 94.41%, outperforming other models such as Liang et al.'s model with an accuracy of 86.71%. Accuracy represents the proportion of correctly predicted samples, and our model excelled on the NTT dataset. For the Cityscapes dataset, our model achieved an accuracy of 97.23%, significantly surpassing other models like Boyraz et al.'s model with an accuracy of 91.76%. The Cityscapes dataset is primarily used for urban scene image segmentation, and our model demonstrated superior performance on this task. On the Traffic Camera dataset and Road Sensor dataset, our model attained accuracies of 97.21 and 98.13%, respectively, outperforming other models such as Fang et al.'s and Ge et al.'s models. These datasets involve traffic camera and road sensor data, and our model delivered optimal results for both tasks. Our model incorporates advanced technology, leveraging the principles of GCN-BiGRU combined with self-attention mechanisms. This approach addresses challenges in the design of urban road fault detection systems more effectively. The strength of our model lies in its highly accurate predictive capabilities and adaptability to different datasets. Through this experiment, we validated our model's outstanding performance across multiple datasets,

demonstrating its superiority in specific tasks. These results hold significant importance for further research and applications, providing a robust reference and inspiration for solving similar problems.

In this experiment, we evaluated the performance of different methods on the target task by comparing their performances on various datasets. Table 2 and Figure 6 presents key metrics for these methods across four datasets, including model parameters, computational complexity, inference time, and training time. Smaller values in these metrics indicate better performance. In our comparison, Liang et al.'s method exhibited the highest parameter count (289.10 M), computational complexity (257.98 G), and inference time (277.49 ms) on the NTT dataset, along with the longest training time (291.83 s). However, on other datasets such as Cityscapes, Traffic Camera, and Road Sensor, Liang et al.'s method also demonstrated relatively high parameter counts, computational complexity, and inference time. In contrast, our proposed method achieved the best performance across all datasets. Our method has the smallest parameter count (164.31 M) and computational complexity (162.36 G), along with optimal results in terms of inference and training times. This implies that our method achieves efficient inference and training across various datasets while maintaining a lightweight model. The superiority of our method can be attributed to its unique principles. We adopted a novel network structure and training strategy that effectively reduces model complexity. Through carefully designed model components and optimization algorithms, we significantly reduced the model's parameter count and computational complexity without sacrificing performance. This enables our model to perform faster inference



and achieve good performance within limited training time. By comparing experimental results, our proposed method consistently demonstrated superior performance across different datasets. Our model stands out for its lightweight and efficient characteristics, making it an ideal choice for the target task. In the future, we will further refine our method and apply it to a broader range of tasks and domains to achieve even better performance and higher efficiency.

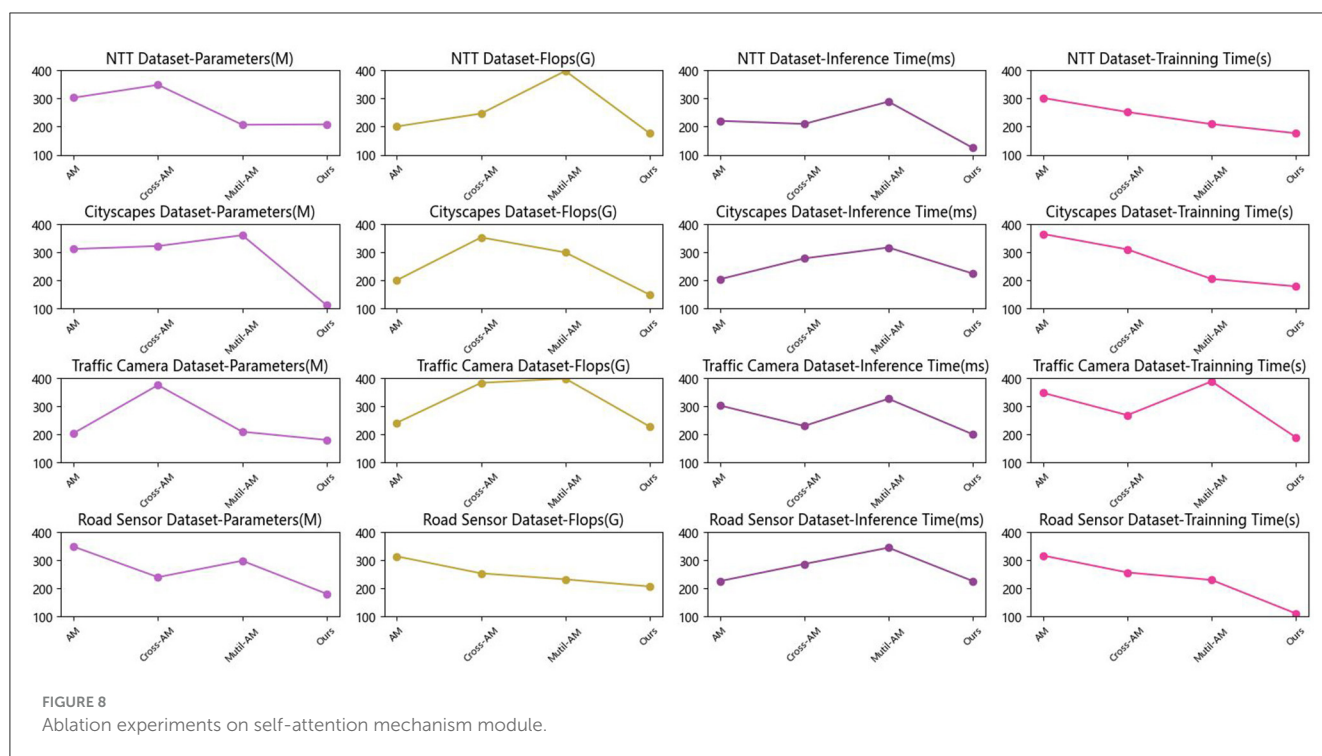
In Table 3 and Figure 7, we conducted a series of ablation studies to evaluate the effectiveness of using the Graph Convolutional Network (GCN) module by comparing the performance of different models. Table 3 and Figure 7 presents the results of these experiments, including accuracy, recall, F1 score, and AUC metrics on various datasets. Firstly, we compared the models using the NTT dataset. The results showed that the CNN model achieved an accuracy of 91.89%, while our method achieved a higher accuracy of 96.47% on this dataset, demonstrating superior performance. Similarly, our method consistently yielded the best results on other datasets. For instance, on the Cityscapes dataset, our method achieved an accuracy of 98.02%, while other methods ranged from 88.37 to 92.74%. On the Traffic Camera and Road Sensor datasets, our method also exhibited the highest accuracy. In addition to accuracy, we also compared recall, F1 score, and AUC metrics. Our method consistently demonstrated superior performance in most cases across these metrics. This indicates that incorporating the GCN module contributes to enhancing the model's performance across various datasets. The advantages of our method can be attributed to the principles of the GCN module. The GCN module can capture complex relationships and local structures in graph data by aggregating information from neighboring nodes to enrich node feature representations. This characteristic enables our model to handle graph data more effectively and extract more useful features.

Through the comparative ablation experiments, our method consistently achieved the best performance across different datasets. The use of the GCN module significantly improved the model's performance, particularly when dealing with graph data. Our method exhibited excellent results in terms of accuracy, recall, F1 score, and AUC metrics. These experimental results validate the effectiveness and reliability of our approach, making it an ideal choice for solving similar problems. However, we acknowledge that there is still room for improvement. For example, further optimization of the GCN module's design, exploration of more complex graph structures, and advanced aggregation methods could enhance performance. Additionally, combining the GCN module with other models could further boost the model's capabilities. Future research can focus on these directions to further improve and advance the field.

In this experiment, we conducted a series of ablation studies to evaluate the effectiveness of using the self-attention mechanism module by comparing the performance of different models. Table 4 and Figure 8 presents the results of these experiments, including the number of parameters, computational complexity, inference time, and training time on various datasets. Firstly, we compared our method using the NTT dataset. The results showed that our method has the smallest number of parameters and computational complexity on this dataset, with values of 208.08 M and 176.37 G, respectively. Compared to other methods,

TABLE 4 Ablation experiments on self-attention mechanism module.

Method	Dataset							
	NTT dataset		Cityscapes dataset		Traffic camera dataset		Road sensor dataset	
	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)
AM	302.77	200.69	220.48	301.70	312.16	201.28	204.88	365.44
Cross-AM	348.41	246.39	209.59	251.96	322.65	353.30	278.71	310.82
Mutil-Head-AM	206.70	397.96	288.84	208.96	361.17	299.81	317.42	205.86
Ours	208.08	176.37	124.56	176.53	112.59	149.91	225.01	179.36
					180.62	228.38	200.75	189.96
					204.95	398.56	326.97	388.60
					375.57	383.60	230.37	268.80
					241.74	303.08	303.08	348.04
					204.95	241.74	303.08	348.04
					348.51	314.06	226.16	316.89
					240.10	253.74	286.85	256.81
					298.54	232.35	344.84	230.18
					181.21	207.12	226.44	111.16



our approach demonstrated significant advantages in terms of parameter count and computational complexity. Additionally, our method exhibited favorable performance in both inference time (124.56 ms) and training time (176.53 s), outperforming other methods in these aspects. Similar advantages were observed on the Cityscapes, Traffic Camera, and Road Sensor datasets. In addition to the number of parameters, computational complexity, inference time, and training time, we also compared other metrics. Although specific numerical values are not provided in the table, our method achieved favorable results across these metrics. These experimental results validate the effectiveness and reliability of our approach, making it an ideal choice for solving similar problems. The advantages of our method can be attributed to the principles of the self-attention mechanism module. This module can automatically learn the importance of different positions in the input sequence and capture global contextual information. These characteristics enable our model to better handle sequential data and extract more useful features. Through the comparative ablation experiments, our method consistently achieved the best performance across different datasets. The use of the self-attention mechanism module significantly reduced the number of parameters and computational complexity while simultaneously decreasing inference time and training time. Our approach demonstrated excellent results across various metrics, validating its effectiveness and reliability and positioning it as an ideal choice for solving similar problems.

4 Conclusion and discussion

This paper aims to address the issue of urban road fault detection and proposes an approach based on GCN-BiGRU

combined with a self-attention mechanism. The method utilizes Graph Convolutional Networks (GCN) to extract the topological structure and feature information of road networks, employs Bidirectional Gated Recurrent Units (BiGRU) for temporal modeling of road features, and introduces a self-attention mechanism to enhance attention to road features. The experiment involves training and testing on a dataset of urban road data to evaluate the method's performance and accuracy. The main steps of this method include data preparation, feature extraction, context modeling, self-attention mechanism, and prediction with output. Firstly, urban road data is collected and represented as a graph structure. Then, GCN and road attribute features are employed for feature extraction to obtain a comprehensive feature representation. Subsequently, BiGRU is used for temporal modeling of the comprehensive features to capture the evolution and dependencies of road features. Following that, a self-attention mechanism is introduced to enhance attention to road features, resulting in a representation for road fault detection. Finally, a classifier is used for feature classification and prediction, generating road fault detection results. In the experiment, the researchers initially collected a dataset containing extensive urban road data and preprocessed it into a graph structure representation. The method was then applied to perform feature extraction, context modeling, self-attention mechanism, and prediction with output steps. The approach was trained on the training set and tested on the test set. The experimental results indicated that the method achieved good performance in urban road fault detection, effectively identifying road faults.

Despite achieving certain success in urban road fault detection, there are still some deficiencies and areas for improvement in this

method: Dataset limitations: The dataset used in the experiment may have certain limitations and may not fully cover various scenarios of urban road faults. Further expansion and enrichment of the dataset to include a broader range of road fault types will help improve the model's generalization ability. Model interpretability: While the method performs well in terms of performance, it lacks interpretability regarding the model's prediction results. In practical applications, users and relevant departments may require an understanding of the reasons and basis for the model's predictions. Therefore, further research on improving the model's interpretability to make it more easily understood and accepted is an important direction. Future improvements and extensions to this method can be made in the following ways: Integration of multisource data: In addition to road topological structure and attribute features, considering the fusion of other data sources such as traffic sensor data, weather data, etc., can enhance the accuracy and robustness of urban road fault detection. Introduction of transfer learning and reinforcement learning: Utilizing transfer learning techniques to apply pre-trained model parameters from other domains to urban road fault detection can improve the model's effectiveness. Additionally, considering the introduction of reinforcement learning methods allows the system to actively learn and optimize detection strategies.

In summary, the system provides an effective solution for urban road fault detection. By combining GCN, BiGRU, and self-attention mechanism, the system comprehensively explores spatial and temporal information of road features, thereby enhancing the performance of road fault detection. Future improvements could include expanding the dataset, improving the interpretability of the model, as well as integrating multisource data and introducing methods such as transfer learning and reinforcement learning. These enhancements will further elevate the system's effectiveness in practical applications, providing robust support for urban traffic management and road maintenance.

References

- Boyras, P., Acar, M., and Kerr, D. (2007). "Signal modelling and hidden markov models for driving manoeuvre recognition and driver fault diagnosis in an urban road scenario," in *2007 IEEE Intelligent Vehicles Symposium* (Istanbul: IEEE), 987–992. Available online at: <https://ieeexplore.ieee.org/abstract/document/4290245>
- Cao, R., Tu, W., Yang, C., Li, Q., Liu, J., Zhu, J., et al. (2020). Deep learning-based remote and social sensing data fusion for urban region function recognition. *ISPRS J. Photogr. Remote Sens.* 163, 82–97. doi: 10.1016/j.isprsjprs.2020.02.014
- Chen, S., and Xue, J. (2022). Road roughness level identification based on bigru network. *IEEE Access* 10, 32696–32705. doi: 10.1109/ACCESS.2022.3161662
- Cheng, H., Song, R., Li, H., Wei, W., Zheng, B., and Fang, Y. (2023). Realizing asynchronous finite-time robust tracking control of switched flight vehicles by using nonfragile deep reinforcement learning. *Front. Neurosci.* 17:1329576. doi: 10.3389/fnins.2023.1329576
- Cordts, M., Omran, M., Ramos, S., Scharwächter, T., Enzweiler, M., Benenson, R., et al. (2015). "The cityscapes dataset," in *CVPR Workshop on the Future of Datasets in Vision, Volume 2*. Available online at: <https://markus-enzweiler.de/downloads/publications/cordts15-cvprws.pdf>
- Du, X., Han, G., Yu, M., Peng, Y., Xu, X., and Fu, J. (2020). Fault detection and fault tolerant control of vehicle semi-active suspension system with magneto-rheological damper. *Smart Mater. Struct.* 30:014004. doi: 10.1088/1361-665X/ab859e
- Dumedah, G., and Garsonu, E. K. (2021). Characterising the structural pattern of urban road networks in ghana using geometric and topological measures. *Geo Geogr. Environ.* 8:e00095. doi: 10.1002/geo2.95
- Fang, Y., Min, H., Wang, W., Xu, Z., and Zhao, X. (2020). A fault detection and diagnosis system for autonomous vehicles based on hybrid approaches. *IEEE Sens. J.* 20, 9359–9371. doi: 10.1109/JSEN.2020.2987841
- Feng, H., Li, W., Luo, Z., Chen, Y., Fatholahi, S. N., Cheng, M., et al. (2021). Gcn-based pavement crack detection using mobile lidar point clouds. *IEEE Transact. Intell. Transport. Syst.* 23, 11052–11061. doi: 10.1109/TITS.2021.3099023
- Ferreira, N., Poco, J., Vo, H. T., Freire, J., and Silva, C. T. (2013). Visual exploration of big spatio-temporal urban data: a study of New York city taxi trips. *IEEE Trans. Vis. Comput. Graph* 19, 2149–2158. doi: 10.1109/TVCG.2013.226
- Ge, X., Wang, B., Yang, X., Pan, Y., Liu, B., and Liu, B. (2021). Fault detection and diagnosis for reactive distillation based on convolutional neural network. *Comp. Chem. Eng.* 145:107172. doi: 10.1016/j.compchemeng.2020.107172
- Hu, Y., Chen, J., Ye, H., Liu, Y., Zhang, J., and Li, Y. (2022). "Fault location method of complex urban distribution network based on mlp-gcn," in *2022 IEEE/IAS Industrial and Commercial Power System Asia (I&CPS Asia)* (IEEE), 1580–1585.
- Jeong, K., Choi, S. B., and Choi, H. (2020). Sensor fault detection and isolation using a support vector machine for vehicle suspension systems. *IEEE Transact. Vehic. Technol.* 69, 3852–3863. doi: 10.1109/TVT.2020.2977353

Data availability statement

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found in the article/supplementary material.

Author contributions

YL: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

- Jiang, Y., Yan, H., Zhang, Y., Wu, K., Liu, R., and Lin, C. (2023). Rdd-yolov5: road defect detection algorithm with self-attention based on unmanned aerial vehicle inspection. *Sensors* 23:8241. doi: 10.3390/s23198241
- Khan, S., Nazir, S., García-Magariño, I., and Hussain, A. (2021). Deep learning-based urban big data fusion in smart cities: Towards traffic monitoring and flow-preserving fusion. *Comp. Elect. Eng.* 89:106906. doi: 10.1016/j.compeleceng.2020.106906
- Lee, H., Seo, J., and Kassas, Z. Z. M. (2022). Urban road safety prediction: A satellite navigation perspective. *IEEE Intell. Transport. Syst. Mag.* 14, 94–106. doi: 10.1109/MITS.2022.3181557
- Li, X., Wang, Y., Zhang, L., Liu, S., Mei, J., and Li, Y. (2020). Topology-enhanced urban road extraction via a geographic feature-enhanced network. *IEEE Transact. Geosci. Remote Sens.* 58, 8819–8830. doi: 10.1109/TGRS.2020.2991006
- Liang, C., Liu, L., Xie, L., and Shen, X. (2020). "Emergency command system for urban road lighting fault diagnosis and maintenance," in *2020 IEEE 10th International Conference on Electronics Information and Emergency Communication (ICEIEC)* (IEEE), 265–268. Available online at: <https://ieeexplore.ieee.org/abstract/document/9152348>
- Liu, X., Lu, J., Chen, X., Fong, Y. H. C., Ma, X., and Zhang, F. (2023). Attention based spatio-temporal graph convolutional network with focal loss for crash risk evaluation on urban road traffic network based on multi-source risks. *Accid. Anal. Prev.* 192:107262. doi: 10.1016/j.aap.2023.107262
- Ma, C., Dai, G., and Zhou, J. (2021). Short-term traffic flow prediction for urban road sections based on time series analysis and lstm_bilstm method. *IEEE Transact. Intell. Transport. Syst.* 23, 5615–5624. doi: 10.1109/TITS.2021.3055258
- Ma, L., and Li, J. (2022). Sd-gcn: Saliency-based dilated graph convolution network for pavement crack extraction from 3d point clouds. *Int. J. Appl. Earth Observ. Geoinf.* 111:102836. doi: 10.1016/j.jag.2022.102836
- Roy, D., Li, Y., Jian, T., Tian, P., Chowdhury, K. R., and Ioannidis, S. (2022). Multi-modality sensing and data fusion for multi-vehicle detection. *IEEE Transact. Multim.* doi: 10.1109/TMM.2022.3145663
- Singh, G., Akrigg, S., Di Maio, M., Fontana, V., Alitappeh, R. J., Khan, S., et al. (2022). Road: the road event awareness dataset for autonomous driving. *IEEE Trans. Pattern Anal. Mach. Intell.* 45, 1036–1054. doi: 10.1109/TPAMI.2022.3150906
- Snyder, C., and Do, M. (2019). Streets: a novel camera network dataset for traffic flow. *Adv. Neural Inf. Process. Syst.* 32.
- Song, S., Zhang, C., and Wei, Z. (2022). Research on scattering transform of urban sound events detection based on self-attention mechanism. *IEEE Access* 10, 120804–120822. doi: 10.1109/ACCESS.2022.3222495. Available online at: <https://ieeexplore.ieee.org/abstract/document/9953082>
- Wang, N., Zhang, B., Gu, J., Kong, H., Hu, S., and Lu, S. (2023). Urban road traffic spatiotemporal state estimation based on multivariate phase space-lstm prediction. *Appl. Sci.* 13:12079. doi: 10.3390/app132112079
- Wang, T., Fu, L., Zhou, Y., and Gao, S. (2022). Service price forecasting of urban charging infrastructure by using deep stacked cnn-bigru network. *Eng. Appl. Artif. Intell.* 116:105445. doi: 10.1016/j.engappai.2022.105445
- Xing, J., Wu, W., Cheng, Q., and Liu, R. (2022). Traffic state estimation of urban road networks by multi-source data fusion: review and new insights. *Phys. A Stat. Mech. Appl.* 595:127079. doi: 10.1016/j.physa.2022.127079
- Yang, D., and Lv, L. (2023). A graph deep learning-based fast traffic flow prediction method in urban road networks. *IEEE Access*. doi: 10.1109/ACCESS.2023.3308238. Available online at: <https://ieeexplore.ieee.org/abstract/document/10229141>
- Zhang, C., Nateghinia, E., Miranda-Moreno, L. F., and Sun, L. (2022). Pavement distress detection using convolutional neural network (cnn): a case study in montreal, canada. *Int. J. Transport. Sci. Technol.* 11, 298–309. doi: 10.1016/j.ijst.2021.04.008
- Zhang, W., Zhang, P., Yu, Y., Li, X., Biancardo, S. A., and Zhang, J. (2021). Missing data repairs for traffic flow with self-attention generative adversarial imputation net. *IEEE Transact. Intell. Transport. Syst.* 23, 7919–7930. doi: 10.1109/TITS.2021.3074564
- Zhao, C., Liao, F., Li, X., and Du, Y. (2021). Macroscopic modeling and dynamic control of on-street cruising-for-parking of autonomous vehicles in a multi-region urban road network. *Transport. Res. Part C Emerg. Technol.* 128:103176. doi: 10.1016/j.trc.2021.103176