Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 4222827, 12 pages
https://doi.org/10.1155/2022/4222827

Research Article

Automatic Traffic Anomaly Detection on the Road Network with Spatial-Temporal Graph Neural Network Representation Learning

Hengyuan Zhang,[1] **Suyao Zhao**,[2] **Ruiheng Liu**,[1] **Wenlong Wang**,[2] **Yixin Hong**,[1] **and Runjiu Hu**[3]

[1]*School of Cyber Science and Engineering, ZhengZhou University, Henan Province, China*
[2]*International College of ZhengZhou University, Henan Province, China*
[3]*Wuhan Eleho Co., Ltd., China*

Correspondence should be addressed to Ruiheng Liu; lrhis2019@stu.zzu.edu.cn and Runjiu Hu; hurunjiu@163.com

Received 25 April 2022; Revised 20 May 2022; Accepted 3 June 2022; Published 20 June 2022

Academic Editor: Kuruva Lakshmanna

Traffic anomaly detection is an essential part of an intelligent transportation system. Automatic traffic anomaly detection can provide sufficient decision-support information for road network operators, travelers, and other stakeholders. This research proposes a novel automatic traffic anomaly detection method based on spatial-temporal graph neural network representation learning. We divide traffic anomaly detection into two steps: first is learning the implicit graph feature representation of multivariate time series of traffic flows based on a graph attention model to predict the traffic states. Second, traffic anomalies are detected using graph deviation score calculation to compare the deviation of predicted traffic states with the observed traffic states. Experiments on real network datasets show that with an end-to-end workflow and spatial-temporal representation of traffic states, this method can detect traffic anomalies accurately and automatically and achieves better performance over baselines.

1. Introduction

With the yearly increase in the usage of vehicles and the gradual expansion of the traffic network, there is a growing demand for the detection of abnormal events in the traffic network. We need to monitor each roadway and manage the road which may be affected promptly after an abnormal situation occurs. Through such effective management, traffic congestions and accidents can be handled in time, thus ensuring the normal operation of the traffic. In the real world, we have multiple sites in the city set up as information collection points to collect large amounts of time series data for the status, speed, and other information about the roadway. This data interacts with each other in a complex way. For example, a traffic jam on one road will increase the number of vehicles on other roads. The increase in traffic on that route will further drive drivers whose destination along it to choose other roads, which has a broader impact. The complexity of the data on each section of the corresponding traffic network has increased rapidly as cities have developed, making previous traditional detection methods less effective. If spatial-temporal traffic flow pattern analysis and anomaly detection can be effectively applied, we can detect abnormal road sections while predicting the stations where problems may occur and then prevent and deal with them without delay. Then, the efficiency of transportation will also be significantly improved.

The detection of unexpected situations in road networks is a typical complex anomaly detection problem. This type of research detects the presence of anomalies by analyzing historical data and solves them in time, thus avoiding further undesirable effects. Classical algorithms for anomaly detection problems include constructing linear models, such as principal component analysis (PCA) method; distance-based measures, such as the k-nearest neighbor (KNN) approach; cluster-based detection, such as density-based spatial clustering of applications with noise (DBSCAN) method; and density-based measures, such as local outlier factor (LOF) algorithm. These schemes determine whether anomalies will occur by analyzing the most salient features. Their clear and concise principles lay a good foundation for further research and analysis. However, the situation of traffic in the real world is often influenced by many factors. For example, real-life traffic conditions are very strongly correlated in time and space, so it is essential to build a nonlinear dynamic spatial-temporal model to predict anomalies in road networks. The simplicity of the above models limits them to more profound analysis. They cannot meet the practical needs of prediction. With the continuous efforts of many scholars, some methods based on graph deep learning have been developed in recent years to improve the accuracy of anomaly detection. Anomaly detection based on generative adversarial networks (GAN) is a popular method widely used in anomaly detection. This method generates the hidden space of the network by learning and capturing the features of the data within the hidden space. But most of these methods are more suitable and targeted to specific points for anomaly analysis. During the course of detecting the anomaly event of the traffic network, we not only need to detect the situation of the scattered individual points in each road but also need to pay attention to the overall situation of the traffic and the interaction between the points.

The concept of a graph neural network (GNN) was first introduced by Gori et al. in [1]. In [2], Bruna et al. introduced convolution into graph neural networks, making the effect greatly improved. In the study of graph-related problems, graph data usually contain two parts of information: attribute information and structure information. In traffic network anomalous event prediction, we can easily get the attribute information of road networks, such as the passage time of vehicles and the speed of vehicles, but what we still lack its structural information.

To solve this problem, we proposed a novel traffic anomaly detection method based on graph deviation network (GDN) models. Our method consists of four main parts: (1) traffic information embedding. The road network is modeled as a graph in this method, where nodes denote road links. And the real spatial and temporal attribute information of each node is embedded into the graph, which is the basis for subsequent analysis. (2) Traffic state spatial-temporal graph structure learning. The input data are some discrete temporal data without structural information. To understand the overall situation, we need to learn the implicit information between nodes to determine the general structural properties. (3) Traffic state prediction. The graph attention model is used to predict the traffic state of each node. A node in a graph structure is not affected by other nodes in the same way. The graph attention mechanism makes the prediction results more realistic by dynamically adjusting the attention function. (4) Traffic anomaly detecting. This section is responsible for explaining the deviations to make the results more convincing. Using the learned directed graph with different weights, we could explain which nodes contribute to the anomalies. Graph deviation scoring identifies and interprets deviations in the learned site relationships in the graph. The results show that we can obtain a model with good results by embedding the site information and further learning the relationship between nodes.

2. Literature Review

2.1. Traffic State Prediction. Traffic state prediction is a very critical problem in the field of transportation. When we want to predict the situation of the traffic network, we need to focus on the time series of the input information. Time series data are arranged in chronological order, varying over time and interrelated. Due to the variety of traffic variables and the variability of traffic states on different road segments, the traffic state prediction task can be seen as a multivariate time series forecasting problem.

There are usually two categories for traffic state prediction: traditional statistical methods and machine learning-based methods. Traditional statistical approaches include the autoregressive model (AR) [3], the moving average model (MA) [4], and the autoregressive integrated moving average model (ARIMA) [5] which is obtained by fusing AR and MA. As early as [6], Williams and Hoel build a model and forecast vehicular traffic flow as a seasonal ARIMA process. In [7], Ghosh et al. used a random walk model, Holt-Winters' exponential smoothing technique, and a seasonal ARIMA model to predict the traffic flows in Dublin. However, realistic situations often have very complex nonlinear characteristics. With the rapid development of machine learning, especially deep learning, time series forecasting problems have further developed. In [8], Cai et al. built a k-nearest neighbor model for short-term traffic multistep forecasting Lim and Zohren [9], explaining some of the methods in which deep learning supports the decision of time series data. This also gives more inspiration to the development of subsequent time series forecasting problems. A traffic forecast model based on long short-term memory was built by Zhao et al. in 2017. In the same year, Yu et al. formulate the problem on graphs and build the model with complete convolutional structures, which realize a much faster training speed with fewer parameters. What is more, the difficulty is that realistic traffic networks often have strong spatial and temporal dependencies. In [10], Cui et al. combine LSTMs and GNNs to discover the temporal and spatial dependencies. In the same year, Wang et al. combine the grid partition-based of local outlier factor (LOF) algorithm and develop a grid-based LOF algorithm to detect the abnormal area in Beijing. Park et al. proposed an LSTM-VAE model replacing the feed-forward network to measure the anomaly score and achieve better results. In 2019, Li et al. train a GAN model with an LSTM-RNN discriminator to compute the anomaly scores for each node. In [11], Geiger et al. use the TadGAN model which is trained with cycle consistency loss to allow for effective time series data reconstruction. A computational data science approach (CDS) is proposed in 2022. Carrera et al. also propose a

Table 1: Results of comparison with previous work.

Methodology	Research contents	Multiple variables		Data relationship		
		Yes	No	Temporal	Spatial	Temporal-spatial
Statistics-based method	The autoregressive integrated moving average model (ARIMA) [6]		√	√		
	The local outlier factor (LOF) [19]	√				√
Machine learning-based method	k-nearest neighbor (KNN) [8]	√			√	
	Long short-term memory (LSTM) (Zhao et al.,2017)	√		√		
	Graph convolutional network (GCN) [20]	√			√	
	This study	√				√

spatiotemporal graph convolutional adversarial network (STGAN) to model traffic dynamics in [12].

2.2. Anomaly Detection. The purpose of anomaly detection is to identify data different from normal data and detect data that differs from the expected pattern. Anomaly detection also has a wide range of applications in the real world, such as device failure detection, medical data detection, network intrusion detection, fraud detection, and time series anomaly detection. We usually classify anomalies into point anomalies, conditional anomalies, and population anomalies. Anomaly detection methods are also commonly categorized into the following three types: supervised anomaly detection, unsupervised anomaly detection, and semisupervised anomaly detection. The basic methods for anomaly detection include statistics-based methods [13], linear model-based methods [14], cluster-based detection DBSCAN [15], distance-based methods KNN [16], density-based proximity detection LOF [17], and integrated methods such as isolated forests [18]. With the continuous efforts of many scholars, some deep learning-based methods have developed in recent years, and the accuracy of anomaly prediction has greatly improved. A generative adversarial network (GAN) is a very representative one. A GAN generator is used to learn the data distribution to find the image it should correspond to and then compare it to determine whether anomalies occur.

Road network anomaly detection can be considered as a supervised anomaly detection task in the case of unbalanced sample classes. We want to determine whether anomalous conditions occur by detecting a time series of operational states at a location at a given time. Here, anomaly detection usually refers to a specific point, but in the anomaly detection of the road network, we need to pay attention to multiple points in a whole network. Our input data do not possess a graph structure in the traffic network prediction problem, but only a string of discrete-time series data. The model needs to learn implicitly and discover the relationship between nodes by itself then use the graph attention network to discover anomalies in the road network.

Compared with the methods and detection scenarios used in previous traffic anomaly detection, the characteristics of the nodes are often homogeneous. While in the actual situation, the road network nodes may choose different indicators according to the specific condition. These results in the structure of road networks are often heterogeneous. At the same time, to form the road network structure, the data need to have structural relationships. It is a difficult task to obtain such data. This paper solves the problem of heterogeneity by mapping nodes into a uniform high-dimensional space. Also, we learn the implicit relationship of the nodes through the attention mechanism and graph deviation score calculation.

In order to compare more clearly the differences between the previous work and our model, we have selected some representative work for comparison; the results are shown in Table 1.

3. Methodology

3.1. Description of the Problem. This study formulates the traffic anomaly detection problem as the supervised anomaly detection task. And two-step detection method is proposed based on the GDL model. The detection nodes distributed in different locations of the road network will predict whether anomalies will occur or not. However, it is usually a difficult task to detect anomalies on roads. The traffic situation at a certain point will be correlated with other nodes in the spatial and temporal dimensions.

(1) *Spatial Dimension.* Due to the nature of traffic road connections, some traffic indicators such as passing speed, travel time, and other important indicators for judging traffic incidents at a station are affected by the nodes connected. For example, when a traffic accident occurs at a certain place, the traffic flow will spread towards the surrounding nodes due to the road closure or different, resulting in changes in the situation of the surrounding associated nodes

(2) *Temporal Dimension.* When an anomaly occurs at a node, the result is reflected in the adjacent time dimension, both in the short-term and long-term time frames. For example, if an actual anomaly was detected at 10 : 00 am, the anomaly's impact may continue to be felt until 11 : 00 am. Sometimes the effect may even manifest itself before the anomaly has fully occurred. For example, an anomaly can be detected at 9 : 55 am

Table 2 shows the notations used in this paper, and Table 3 shows the abbreviations we used in this paper.

Table 2: Notations used in this paper.

Symbol	Description
$I_{\text{test}}^{(T_{\text{test}})}$	Data of different stations at the moment of T_{test}
v_i	The station's embedding vector i_{th}
M_{ij}	Adjacency matrix donates the relationship between node i and node j
R_i	The candidate relation node set of i
e_{ji}	The similarity between the embedding vector of the node i and the embedding vector of its candidate node j
$g_i(t)$	The score of the nodes through the graph attention mechanism
$a_{i,j}$	The attention coefficient
$\text{Err}_i(t)$	Error value of the site i at the time t
$A_s(t)$	A smoothed score will be used to decide on an exception has happened or not

Table 3: Abbreviation used in this paper.

Abbreviation	Full name
PCA	Principal component analysis
KNN	k-nearest neighbor
DBSCAN	Density-based spatial clustering of applications with noise
LOF	Local outlier factor algorithm
GNN	Graph neural network
GDN	Graph deviation network
AR	Autoregressive model
MA	The moving average model
ARIMA	The autoregressive integrated moving average model
LSTMs	Long short-term memory
GDL	Geometric deep learning
GCNs	Graph convolutional network
GAT	Graph attention networks
IQR	Interquartile range
SMA	Simple moving average
OCSVM	The one-class support vector machine
HBOS	The histogram-based outlier detection
COPOD	Copula-based outlier detector
IFOREST	Isolation Forest

3.2. Overview. We proposed a novel method for traffic anomaly detection based on the graph deviation network (GDN) method. As Figure 1 shows, our method consists of four main parts:

(1) *Traffic Information Embedding.* The actual attribute information of each node is embedded into the graph, which lays the foundation for subsequent analysis

(2) *Traffic State Spatial-Temporal Graph Structure Learning.* The input data are some discrete temporal data that do not have structural information. We need to learn the implicit information between nodes to determine the general structural properties for understanding the overall situation

(3) *Traffic State Prediction.* A point in a graph structure is affected by other nodes differently. The graph attention mechanism makes the prediction results more realistic by dynamically adjusting the attention function

(4) *Traffic Anomaly Detection.* This section explains the deviations to make the results more accurate. The results show that by embedding the site information and further learning the relationship between nodes, we can obtain a model with good results

In this paper, we use an unsupervised learning method. According to its general principles, our training set is assumed to consist entirely of normal data. Each time, the data that we feed into the model contains data for certain stations within a certain time window. To detect traffic anomalies at a single station, we choose to use the number of stations as a feature of the data. Besides, in order to input and organize the information of all nodes simultaneously, we extract the original timestamp information discrete-time features and add them to the input vector.

Our data were collected from different stations in the city during the T_{test} period: they can be expressed as $= [I_{test}^{(1)}, \ldots, I_{test}^{(T_{test})}]$. We determine if an exception has occurred through the value of output: $o(t) \in \{0, 1\}$. If at the time of t we observe that $o(t) = 1$, this indicates that an anomaly is occurring at this moment. Figure 2 shows the procession of our detection network.

3.3. Traffic Information Embedding. Because our network is heterogeneous, different nodes have different properties and features. Therefore, if we want to reflect the information of different nodes under the same graph structure, we need to map the features to the same space. We use an embedding vector to represent the features of each station:

$$I_{\text{test}}^{(T\text{test})} \longrightarrow v_i, \quad v_i \in R^d, \text{ for } i \in \{1, 2, \cdots, N\}. \tag{1}$$

N represents the number of stations, d for the embedding vector dimension, and v_i for the i_{th} station's embedding vector.

The significance of the embedding layers is reflected in which it turns our sparse matrix into a dense matrix by some linear transformations, and this dense matrix characterizes all the sites with some typical features. This dense matrix not only represents the one-to-one correspondence between the dense matrix and individual features; in fact, it also contains a large number of intrinsic relationships between nodes and nodes. Meanwhile, the result of variant 2 in the part of ablation experiments shows that the module of embedding layer is necessary. When remove this module, the accuracy is decreased by 36.99%. The detection rate is decreased by 8.45% while the false alarm rate is increased to 39.14%. By

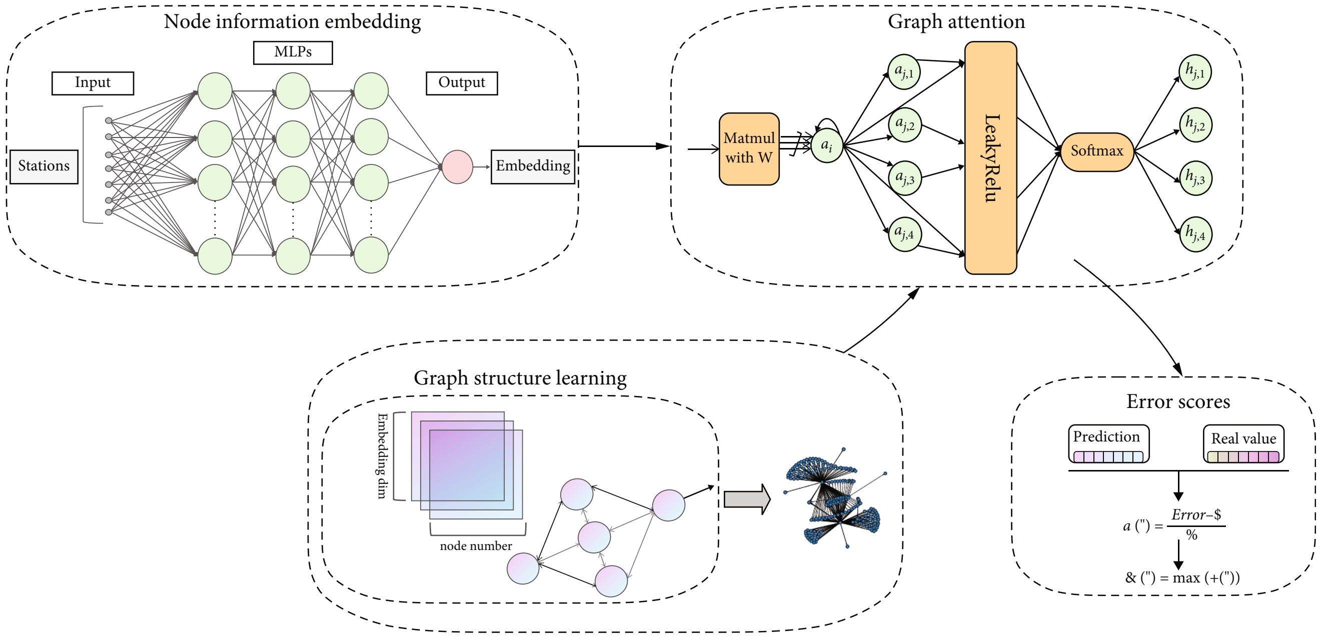

Figure 1: Methodology of traffic network anomaly detection.

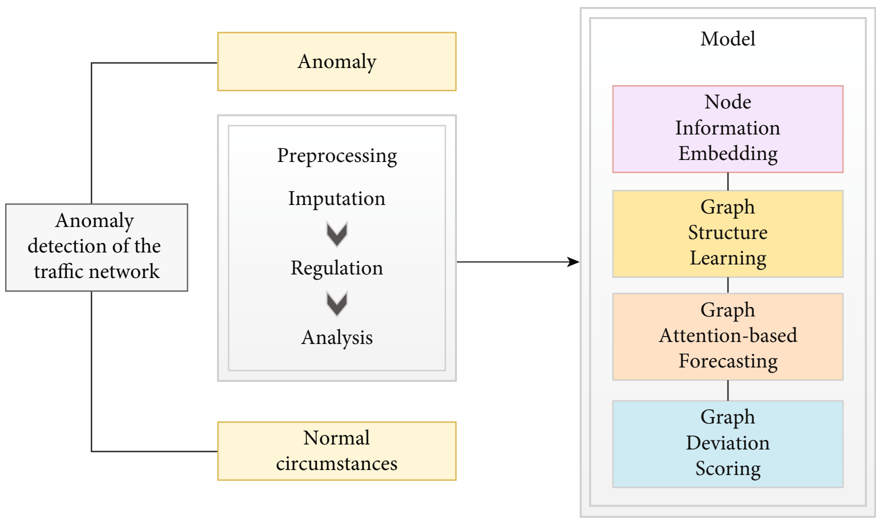

Figure 2: Process of traffic network anomaly detection.

comparing the performance of variant 2 with the proposed model, the importance of the traffic information embedding module is emphasized.

3.4. Traffic State Spatial-Temporal Graph Structure Learning. After we get the node embedding vectors, we will use the v_i to learn the graph structure learning.

The data we use is originally some time series sequences and do not have connections between nodes. However, we want to understand whether there is a relationship between nodes as well as the magnitude of the influence in the model of anomaly detection in the road networks. Therefore, we will learn the implicit relationship between sites during the training process by designing a structural framework. Eventually, we will obtain a directed graph representing the connections between different sites through learning. In this directed graph, the nodes represent the sites where we collected information, and the edges represent the dependencies between them. The connection between the two sites represents the information collected at one site that will be used for training at the second site. We represent this directed graph by the adjacency matrix M, where M_{ij} represents the existence of directed edges from node i to node j, which means that the two sites have a spatial correlation. We use R_i to represent the candidate relation node set of i.

$$R_i \in \{1, 2,. \cdots, N\} \setminus \{i\} \tag{2}$$

To understand the dependencies between the sites, we calculate the similarity between the embedding vector of the node i and the embedding vector of its candidate node $j \in R_i$:

$$e_{ji} = \frac{{v_i}^T v_j}{\|v_i\| \cdot \|v_j\|} \text{ for } j \in R_i, \tag{3}$$

$$M_{ji} = \{ j \in \text{TopK}(\{ e_{ji} : k \in R_i \}) \}. \tag{4}$$

We first compute the normalized dot product with a normalized layer between the embedding vector of the sensor i and the candidate relation $j \in R_i$. Then, we will choose the index of the Topk values of the input (i.e., normalized dot product) and select them as the most important K node related to node i. The user determines the value of k by the desired degree of sparsity. The process of computing the dot product and selecting the Topk nodes will exceed in each batch during every epoch. Finally, we will get the adjacency matrix M which denotes the implicit relationship between the station nodes.

3.5. Traffic State Prediction. Previously, we used the module of Topk, and our model can learn the graph structure of the road node proximity implicit in the data by calculating the dot product of the embedding vector. After that, we get an overall correlation topology. A specific node and the road junctions associated with it do not have precisely the same situation in the traffic road network. The usual situation is that when an accident occurs at a certain location, the impact of that point will spread to the surrounding nodes with different degrees.

Traditional GCNs usually have significant limitations for such task requirements. GCNs cannot adjust the neighbor weights (cannot specify different weights to different nodes in a neighborhood) based on the feature attributes of the nodes. This limits the ability of the model to capture the relevance of spatial information, resulting in poorer predictions in practice. Therefore, we use the mechanism of graph attention by computing the nodes and using the critical nodes $j \in M_{ji} = \{ j \in \text{Topk}(\{ e_{ji} : k \in R_i \}) \}$ which we learned by Topk to compute the similarity function $\pi(i, j)$.

$$g_i(t) = v_i \oplus W{x_i}^{(t)}, \tag{5}$$

$$\pi(i, j) = \text{LeakyReLu}\left(a^T \left({g_i}^{(t)} \oplus {g_j}^{(t)} \right) \right). \tag{6}$$

Here, our formulation differs from the standard formulation of GAT in the paper. In order to use the embedding vector in a batter way, we have chosen a sliding window using $x_i \in R$ that denotes the feature of the node's input.

The sliding window control is used to select a fixed size time range of data as an input to predict the result at the next time point. Here, the time t is defined as x_i:

$${x_i}^t \left[I^{t-\text{window}}, I^{t-\text{window}+1}, \cdots, I^{t-1} \right]. \tag{7}$$

$R(i) = \{ j | M_{ij} > 0 \}$ is the adjacency matrix M_{ij} that represents the topology through what we have learned, which is also the set of nodes that are adjacent to node i. $W \in \mathbb{R}^{d \times w}$ denotes the weight matrix of the linear transformation layer shared by all nodes of our model for feature enhancement of the input nodes. v_i donates the embedding vector of node i. $\oplus$ represents a stitching operation on the transformed features.

Thus, the g_i here can combine the transformed features with the embedding vector we learned, which may improve the effectiveness of the attention mechanism. For the final activation function, we chose the general LeakyReLu as the nonlinear activation to compute the attention coefficient.

After calculating the correlation coefficient, we used softmax to normalize the correlation coefficient and obtained the attention coefficient $a_{i,j}$.

$$a_{i,j} = \frac{\exp(\pi(i, j))}{\sum_{k \in N(i) \cup \{i\}} \exp(\pi(i, j))}. \tag{8}$$

The softmax function converts a vector of K real values into a vector of K real values that sum to 1. The K input values to the vector can be positive, negative, zero, or greater than 1, but softmax converts them to be between 0 and 1, so the converted values can be interpreted as probabilities. If one of the inputs is small or negative, softmax turns it into a small probability, and if the input is large, it turns it into a large probability, but always stays between 0 and 1. This property is exactly satisfied by the properties of the weighting calculation.

The GAT obtains a score for each neighboring node by calculating the degree of correlation between them, but this score can have a large variation. Softmax is useful here because it converts the calculated correlation score into a normalized probability distribution, and the converted result is distributed between 0 and 1 depending on the size of the score, which can be considered as the importance of each node, i.e., weighting values. Based on the above considerations, we chose to use the softmax function.

Finally, we aggregate our computed attention coefficients to obtain new features for the output of each node which incorporate the information we have learned from the neighborhood nodes. Here, the activation function we chose to use is ReLu.

$$\text{output}_i = \text{Re Lu}\left(\sum_{j \in R(i)+1} a_{i,j} W x_j \right). \tag{9}$$

3.6. Traffic Anomaly Detection. To make the model more accurate, we hope to understand the anomalies of the relationship and the reasons why the anomalies exist. So we calculate the anomaly scores for each site and combine them to get a single anomaly score for each time tick. This allows the user to specify which sites are uncommon. The anomaly score is calculated by comparing the expected behavior at moment t with the observed behavior. Error value Err of site i at the time t can be expressed as follows:

$$\text{Err}_i(t) = \left| {s_i}^{(t)} - {\hat{s}_i}^{(t)} \right|. \tag{10}$$

Since different stations have different characteristics, their deviation values can vary considerably accordingly. We robustly normalize the error values for each site to avoid one site being overly influenced by other sites.

$$a_i(t) = \frac{\mathrm{Err}_i(t) - \tilde{\mu}_i}{\tilde{\sigma}_i}, \tag{11}$$

where $\tilde{\mu}_i$ and $\tilde{\sigma}_i$ are the median and interquartile range (IQR) across time ticks of the $\mathrm{Err}_i(t)$ values, respectively. IQR is defined as the spread difference between the 75th and 25th percentiles of the data. We chose median and IQR rather than mean and standard deviation. IQR enhances the accuracy of dataset statistics by dropping lower contribution, outlying points. And so as the median will reduce the affection of the outliers which means better robustness. The work [21] shows that they have better robustness in the event of anomalies.

We use the max function to aggregate the sites to calculate the overall anomaly at moment t.

$$A(t) = \max_i a_i(t). \tag{12}$$

To suppress sudden changes in values, we use a simple moving average (SMA) [22] to generate a smoothed score $A_s(t)$. If $A_s(t)$ exceeds a fixed threshold, the time stamp t will be marked as an exception. Here, we set the threshold to the maximum value of $A_s(t)$ on the validation data.

4. Experiments

4.1. Datasets. The incident logs conclude 13759 congestion logs, 1139 accident logs, 74 road construction logs, and 3 lane closure logs. The dataset for this experiment comes from Beijing and is a series of data collected in a natural traffic environment. Figure 3 shows the distribution of the anomalies according to the dataset. Before predicting anomalies in the road network, we have two requirements for the input training data: (1) it is required to be clear which station the data describes, and (2) the data needs to be temporally coherent. We need to perform a preprocessing operation on the raw data to meet these requirements. The time-stamped features of the original data are extracted and added to the original ID features, so that even if the data are completely disrupted, we can ensure that the sequence of the time series is not broken.

4.2. Evaluation Metrics. Here, we use the three metrics of detection rate, false alarm rate, and accuracy to evaluate and compare the baseline model with our modified model.

$$\text{Detection Rate} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{13}$$

$$\text{False Alarm Rate} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}, \tag{14}$$

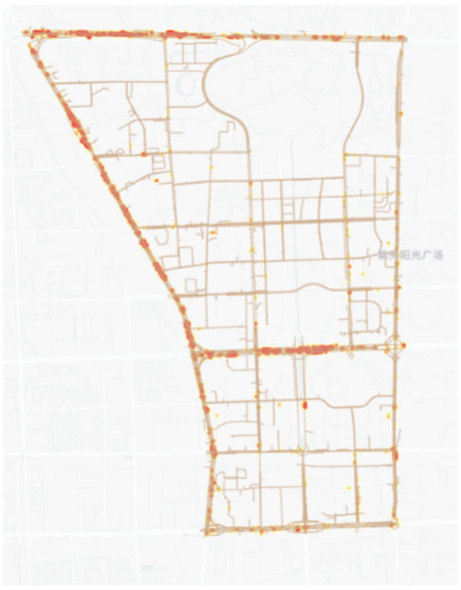

FIGURE 3: Kernel density map of traffic anomaly distribution.

$$\text{Accuracy} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP}} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}. \tag{15}$$

The definitions of FN, FP, TN, and TP are as follows:

(1) FN (false negative): the sample is judged to be negative, but it is a positive sample in fact

(2) FP (false positive): the sample is judged to be positive, but it is negative

(3) TN (true negative): the sample is judged as negative, and in fact, it is also a negative sample

(4) TP (true positive): the sample is judged to be positive and is positive in fact

With these metrics, we can estimate which models have better results.

4.3. Comparison Methods. We compared the model proposed in this paper with the state-of-the-art methods (SOTA) for anomaly detection, together with some previous studies on anomaly detection using the baseline model as a comparison of the results. The models and methods include the following.

4.3.1. Linear Model

(1) OCSVM. The one-class support vector machine [23] is different from the traditional SVM model. OCSVM is suitable for classification problems where there is only one type of sample or where there are two types of samples, but the number of samples of one type is much less than the number of the other type. The anomaly detection problem is typical of the latter. The samples can be classified as either normal or abnormal. Furthermore, the number of normal samples is much larger than the number of abnormal samples. When we try to make a judgment on a sample, if the features

extracted from that sample do not match the features of the normal case included in the classifier, we can conclude that it is an abnormal case.

4.3.2. Proximity-Based

(1) LOF. The main idea of the local outlier factor method is to assign a degree of outlier to each data example within a certain range [24, 25]. The outlier factor is the average of the ratio of the local reachability density of a point to the local reachability density of its neighboring nodes. If this ratio is closer to 1, it means that the densities between the two nodes are similar, and they may jointly belong to a cluster; if this ratio is much larger than 1, it means that the density of the node is much smaller than the density of its neighboring points. The node is most likely to be an anomaly.

(2) HBOS. The histogram-based outlier detection assumes the independence of features and scores the records in linear time. It is less accurate but has superior computational speed compared to multivariate methods [26]. This method divides the sample into multiple intervals according to the characteristics. The probability of outliers is higher in intervals with a small sample size.

4.3.3. Probabilistic Model

(1) COPOD. The algorithm of copula-based outlier detector not only does not need to calculate the distance between samples but also does not need to adjust the parameters. It has the advantages of high speed as well as low cost of running [27]. The COPOD method calculates the degree of the anomaly in each dimension and then finds the average anomaly in overall dimensions. This method evaluates the anomalies by estimating the tail probability of each point.

(2) IFOREST. The algorithm of isolation forest is a fast anomaly detection algorithm based on integrated learning, which uses a binary tree to slice and dice the data. The depth of the data points in the binary tree reflects the "sparseness" of the data [28]. We determine the degree of the anomaly of the data by estimating the anomaly score. The shorter the average path's length of the data in multiple trees, the more likely the data is anomalous.

(3) LSTM-VAE. LSTM-VAE [29, 30] replaces the feed-forward network in a VAE with LSTM to take advantage of LSTM and VAE. It can measure reconstruction error with the anomaly score.

(4) MAD-GAN. MAD-GAN [31] trains a GAN model with a LSTM-RNN discriminator to compute the anomaly scores for each node.

4.4. Experimental Setup. Our models are built in a Python environment with PyTorch version 1.10.1 and PyTorch Geometric Library version 2.0.3. They are also deployed on Intel(R) Xeon(R) Gold 6278C CPU @ 2.60 GHz and NVIDIA Tesla T4 graphics cards.

4.5. Hyperparameter. We use an Adam optimizer with a learning rate of 0.001 to achieve the optimization. The model is trained for up to 100 epochs, and we use an early stopping technique. We populate each node as a 32-dimensional vector and use a k value of size 6 in the Topk module. The number of hidden cells is set to 128, the batch size is set to 32, and the sliding window size is set to 48. We use 80% of the data as a training dataset, and the remaining 20% is selected as the validation set. After confirming the selection of these crucial hyperparameters, we use the grid search method to perform the search process. The specific range of variations of the hyperparameters is the batch size, slide windows, embedding dim, and Topk.

The best combination of parameters is obtained after the search, as shown in Table 4. The parameter embedding dim represents that we reflect the features of the nodes into a space of 32 dimensions. And we use the LeakyReLu as the activation function when computing the weights. And the best number of the neighbors we select is 6.

4.6. Results. In Table 5, we show the performance of our GDN method and the baselines on detection rate, false alarm rate, and accuracy on the dataset from Beijing. The results show that the GDN method has a value of 0.9749 and 0.9701 on detection rate and accuracy, which is better than other baseline experiments. The value of the false alarm rate is 0.0251, which is significantly lower than other baseline experiments. Meanwhile, we also compare the results with some state-of-the-art methods including LSTM-VAE and MAD-GAN. We can find that our model is equivalent in terms of accuracy and detection rate and has a better performance on FAR. These results show that our method has a superior prediction result.

4.7. Ablation Experiments. Ablation experiments are deployed as sensitivity analysis for the major modules of the proposed model. Three variants are implemented by gradually removing a certain module of the proposed model:

(1) *Variant 1*. This variant is implemented by removing the Topk module of the proposed model to investigate the importance of the learned traffic graph structural representations. The Topk module helps the proposed model to filter out the nodes with high scores of interconnections and figure out which nodes may have a relationship. After removing the Topk module, all nodes in the graph are associated, which means a complete static graph. The data in the table shows a significant decrease in the accuracy of the model, which indicates that selective joins in the graph structure make the model less bloated and thus improve the performance

(2) *Variant 2*. To scrutinize the effectiveness of the traffic information embedding module, this variant is implemented by excluding the embedding module of the proposed model. The embedding mechanism helps the proposed model to find nearby neighbors in the space and reduces the amount of data needed

Table 4: Optimal combination of hyperparameters.

Variable	Value
Learning rate	0.001
Batch size	32
Epochs	100
Optimizer	Admn
Activation function	LeakyReLu
Embedding dim	32
Topk	6
Slide windows	48

Table 5: Results of anomaly detection compared with baseline models.

Method	Accuracy	Detection rate	False alarm rate
LOF	0.80	0.89	0.09
IFOREST	0.83	0.80	0.08
OCSVM	0.82	0.84	0.08
HBOS	0.87	0.31	0.03
COPOD	0.82	0.82	0.08
LSTM-VAE	0.97	0.97	0.03
MAD-GAN	0.91	0.35	0.16
GDN (the selected model)	0.97	0.97	0.02

Table 6: Results of ablation experiments.

Model	Detection rate	False alarm rate	Accuracy
GDN	97.49%	2.51%	97.01%
Variant 1	99.96%	49.97%	50.46%
Variant 2	89.25%	39.14%	61.13%
Variant 3	81.80%	18.36%	81.63%

for training. After we remove the embedding layer and substituted with $g(t) = Wx_i$, the detection rate and the accuracy become worse. This indicates that the presence of embedding layer helps to learn the relationship between nodes in the graph attention mechanism

(3) *Variant 3*. To explore the importance of the attention module, this variant is implemented by removing the attention mechanism of the proposed model. The attention mechanism helps the proposed model to show the closeness of the relationships between nodes and how they affect each other. When we set the weight of all nodes to 1 (all nodes have equal weight), the new model does not work as well as the original model. The result shows that it is crucial to use the attention mechanism to assign weights according to the closeness or detachment of the nodes

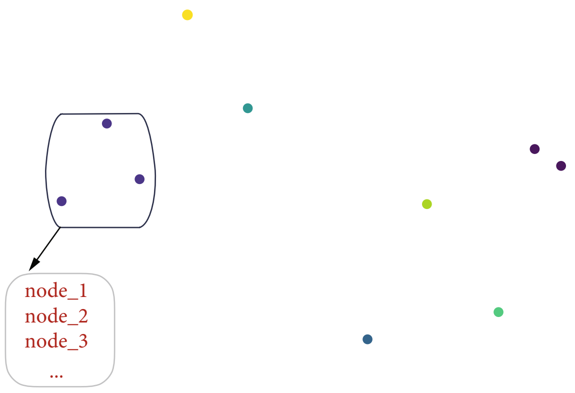

Figure 4: A plot of the node embedding vector encoded by the model. Different colors denote each class. We used the t-SNE and DBSCAN to achieve the dimensionality reduction and clustering.

All variants of the proposed model are tested on Beijing dataset to investigate the optimal architecture of the proposed model. The results are shown in Table 6.

The model we used is obtained by fusing several modules. In order to verify what contribution each module makes to the proposed model, we dynamically adapt the method we want to confirm while keeping the other methods unchanged. The specifics of the changes in the results give us ideas of the role played by the method. Table 4 shows the results obtained by the ablation experiments. The performance of the variants on all metrics can be discussed in comparison with the proposed model. For variant 1, the accuracy is decreased by 47.98% while the false alarm rate is increased to 49.97%. Although the detection rate is improved, both the false alarm rate and the accuracy are declined at the cost. By comparing the performance of variant 1 with the proposed model, we verify the necessity to learn the traffic graph structural representations. For variant 2, the accuracy is decreased by 36.99%. The detection rate is decreased by 8.45% while the false alarm rate is increased to 39.14%. By comparing the performance of variant 2 with the proposed model, the importance of the traffic information embedding module is emphasized. For variant 3, the accuracy is decreased by 15.85%. The detection rate is decreased by 16.09% while the false alarm rate is increased to 18.36%. By comparing the performance of variant 3 with the proposed model, the effectiveness of the integration of the attention module is demonstrated. Taken together, the Topk module, the embedding layer, and the attention mechanism all make unique contributions to our model. They are fused to help our model achieve better results.

4.8. Interpretability of Model

4.8.1. Embedding Layer. In our modified model for anomaly problem detection in road networks, we used an embedding vector layer to encode all the heterogeneous stations to map the embedding vector into a 2D space, using the distance metric to represent the relationship between different stations. Here, we use the t-SNE to reduce the dimension and

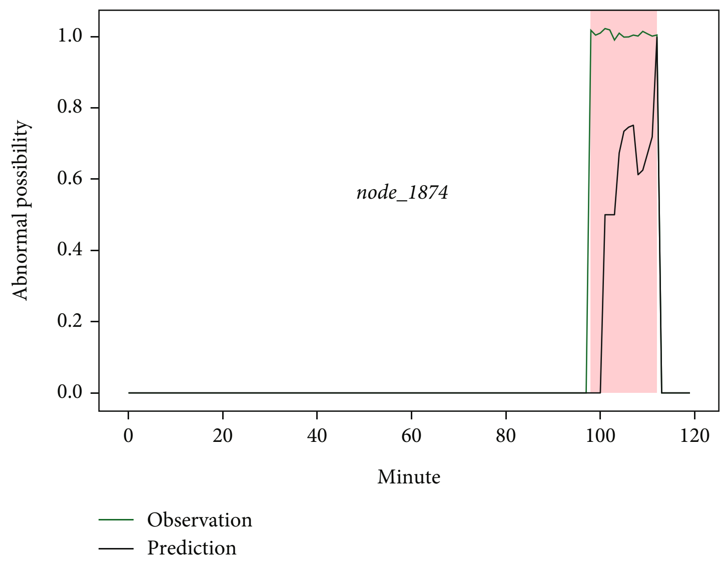

Figure 5: Explain the anomaly by comparing expected and observed data.

use the DBSCAN to cluster all the nodes. We visualize the vectors in Figure 4. We compute the similarity of geographic locations by Euclidean distance to cluster geographically similar sites together. The nodes with a similar embedding vector will be clustered together. Nodes in the same class of clusters may be at connected road junctions or have similar traffic flow conditions in the road network.

4.8.2. Attention Mechanism for Association between Nodes. The model we transformed is based on the graph structure. Our model learns a latent network structure and represents the directed graph. The magnitude of the interaction between associated nodes is not the same. With the different weights in the attention mechanism, we can get a clearer picture of the degree influenced by the surrounding nodes by assigning different degrees of correlation to adjacent nodes based on the graph structure obtained from Topk learning. Figure 5 shows the predicting situation. When an anomaly is detected at the target location, the weight of attention can be seen as a distribution of the diffusion influence of the node to the surrounding area. Through this distribution, we can understand the critical location of the road. When the real anomaly occurs, we can carry out different sparing measures according to this distribution.

5. Conclusion

In this paper, we proposed an automatic traffic anomaly detection method. We can understand whether stations relate to each other or not and the closeness of the relationship by embedding the data collected from stations in the graph and learning from graph structure. Based on this, we predict the traffic road anomalies and correct the model by deviation scores. Experiments on real datasets show that using the GDN model to predict traffic anomalous events outperforms the baseline model. Our model is interpretable, facilitating user understanding and further in-depth research.

However, there are still some limitations to the application of our model as evidenced by the following: (1) Because of the graph network structure, it has the disadvantage of being computationally intensive. The long training time of the model makes it difficult for applications in light or real-time scenarios. (2) Our model models multivariate temporal sequences and uses the relationships between potential graph structures between learning nodes to make predictions. We ignore the influence of the structure on the road network roads under real conditions, i.e., the influence of known prior knowledge such as road topology and weather on the model. However, this influence may have a huge impact on further improving the model.

For future study, the proposed method can be further developed from the following perspectives: (1) modeling the heterogeneity of road network topology. The current model considers multivariate traffic time series heterogeneity by learning hidden relationships between different time series as a graph. The road network topology contains bountiful auxiliary information, such as road class and geometry. Integrating the heterogeneity regularizer of road network topology into the current model may help improve the performance of traffic anomaly detection. (2) Forecasting the traffic state and anomalies simultaneously. The proposed model is trained by taking anomaly detection as an optimization objective. Collaborative training for traffic state prediction and anomaly detection modules can broaden the model applications in intelligent transportation systems. (3) Considering that we are using an unsupervised learning training method, we can change the architecture of the model or design some online training methods to achieve better performance.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no competing interest.

References

[1] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *2005 IEEE international joint conference on neural networks*, vol. 2no. 2005, pp. 729–734, Montreal, QC, Canada, 2005.

[2] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, https://arxiv.org/abs/1312.6203.

[3] B. S. Chen, S. C. Peng, and K. C. Wang, "Traffic modeling, prediction, and congestion control for high-speed networks: a fuzzy AR approach," *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 5, pp. 491–508, 2000.

[4] R. P. Haining, "The moving average model for spatial interaction," *Transactions of the Institute of British Geographers*, vol. 3, no. 2, pp. 202–225, 1978.

[5] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.

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

[7] B. Ghosh, B. Basu, and M. O'Mahony, "Time-series modelling for forecasting vehicular traffic flow in Dublin," in *84th Annual Meeting of the Transportation Research Board*, Washington, DC, 2005.

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

[9] B. Lim and S. Zohren, "Time-series forecasting with deep learning: a survey," *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2194, p. 20200209, 2021.

[10] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, "Traffic graph convolutional recurrent neural network: a deep learning framework for network-scale traffic learning and forecasting," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 11, pp. 4883–4894, 2019.

[11] A. Geiger, D. Liu, S. Alnegheimish, A. Cuesta-Infante, and K. Veeramachaneni, "Tad GAN: time series anomaly detection using generative adversarial networks," in *2020 IEEE International Conference on Big Data (Big Data)*, pp. 33–43, Atlanta, GA, USA, 2020.

[12] F. Carrera, V. Dentamaro, S. Galantucci, A. Iannacone, D. Impedovo, and G. Pirlo, "Combining unsupervised approaches for near real-time network traffic anomaly detection," *Applied Sciences*, vol. 12, no. 3, p. 1759, 2022.

[13] X. Zhu, *Anomaly Detection through Statistics-Based Machine Learning for Computer Networks*, The University of Arizona, 2006.

[14] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, *A Novel Anomaly Detection Scheme Based on Principal Component Classifier*, Technical Report, Miami Univ Coral Gables Fl Dept Of Electricaland Computer Engineering, 2003.

[15] M. Çelik, F. Dadaşer-Çelik, and A. Ş. Dokuz, "Anomaly detection in temperature data using DBSCAN algorithm," in *2011 international symposium on innovations in intelligent systems and applications*, pp. 91–95, Istanbul, Turkey, 2011.

[16] M. Y. Su, "Real-time anomaly detection systems for denial-of-service attacks by weighted k-nearest-neighbor classifiers," *Expert Systems with Applications*, vol. 38, no. 4, pp. 3492–3498, 2011.

[17] T. Wang, W. Zhang, J. Wei, and H. Zhong, "Workload-aware online anomaly detection in enterprise applications with local outlier factor," in *2012 IEEE 36th Annual Computer Software and Applications Conference*, pp. 25–34, Izmir, Turkey, 2012.

[18] Z. Ding and M. Fei, "An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window," *IFAC Proceedings Volumes*, vol. 46, no. 20, pp. 12–17, 2013.

[19] Q. Wang, W. Lv, and B. Du, "Spatio-temporal anomaly detection in traffic data," in *Proceedings of the 2nd International Symposium on Computer Science and Intelligent Control*, pp. 1–5, Stockholm, Sweden, 2018.

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

[21] C. Park and B. R. Cho, "Development of robust design under contaminated and non-normal data," *Quality Engineering*, vol. 15, no. 3, pp. 463–469, 2003.

[22] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 387–395, London, United Kingdom, 2018.

[23] S. Omar, A. Ngadi, and H. H. Jebur, "Machine learning techniques for anomaly detection: an overview," *International Journal of Computer Applications*, vol. 79, no. 2, pp. 33–41, 2013.

[24] S. Agrawal and J. Agrawal, "Survey on anomaly detection using data mining techniques," *Procedia Computer Science*, vol. 60, pp. 708–713, 2015.

[25] Y. Yan, L. Cao, and E. A. Rundensteiner, "Scalable Top-N local outlier detection," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, Halifax, NS, Canada, 2017.

[26] M. Goldstein and A. Dengel, "Histogram-based outlier score (HBOS): a fast unsupervised anomaly detection algorithm," *KI-2012: poster and demo track*, vol. 9, 2012.

[27] Z. Li, Y. Zhao, N. Botta, C. Ionescu, and X. Hu, "COPOD: copula-based outlier detection," in *2020 IEEE international conference on data mining (ICDM)*, pp. 1118–1123, Sorrento, Italy, 2020.

[28] X. Zhao, Y. Wu, D. L. Lee, and W. Cui, "iForest: interpreting random forests via visual analytics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 407–416, 2019.

[29] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1544–1551, 2018.

[30] S. Lin, R. Clark, R. Birke, S. Schönborn, N. Trigoni, and S. Roberts, "Anomaly detection for time series using VAE-

LSTM hybrid model," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4322–4326, Barcelona, Spain, 2020.

[31] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S. K. Ng, "MAD-GAN: multivariate anomaly detection for time series data with generative adversarial networks," in *International conference on artificial neural networks*, pp. 703–716, Cham, 2019.