A Vehicle Abnormal Behavior Detection Model in Single Intelligent Vehicle Scenarios

Wenhui Wang[1], *Qiang Zhu*[2], *Chia-Wei Lee*[3], *Zhenjiang Zhang*[1*]

[1] *Electronics and Information Engineering, Beijing Jiaotong University, China*
[2] *School of Mathematics and Statistics, Xidian University, China*
[3] *Department of Computer and Science, University of Taipei, Taiwan*
23111039@bjtu.edu.cn, zhuqiang@mail.xidian.edu.cn, cwlee@utaipei.edu.tw, zhangzhenjiang@bjtu.edu.cn

Abstract

Connected and automated vehicles (CAVs) play a vital role in transforming human mobility, tackling road congestion and road safety. However, CAVs rely heavily on the security, accuracy, and stability of sensor readings and network data. When there are anomalies in the data, it is necessary to detect them in a timely manner and handle them. However, under single intelligent vehicle scenarios, existing detection methods often struggle to identify unknown types of anomalies and are difficult to deploy on computationally limited vehicle terminals. To address the aforementioned issues, this paper proposes a vehicle anomaly data detection method based on deep learning. First, we modify the discriminator based on the GAN network, so that the network can assign different weights to different sensors, thus improving the generalization performance of the model. Afterwards, we assign weights to each parameter of the model during the training process, and then prune the model according to the weights to improve its computational speed. We verify the reliability of our method on the Safe Pilot Model Deployment (SPMD) data set. It is shown that the proposed model has good detection performance for various anomaly data, especially when facing data that were not encountered during the training process, and the proposed model effectively reduces the computational time of the detection process.

Keywords: Connected and automated vehicles, Anomaly detection, GAN, Single intelligent vehicle, Pruning

1 Introduction

Our existing transportation infrastructure stands at the threshold of a profound evolution towards a seamlessly interconnected, automated, and intelligent network, propelled by the swift rise of connected and automated vehicles (CAVs) [1]. CAVs, spanning a spectrum of connectivity and automation levels, are anticipated to be central to the upcoming stage of the transportation revolution. This transition promises enhanced accessibility, efficiency, safety, environmental friendliness, and ultimately, sustainable transportation alternatives [2-3].

The advancement of CAVs technology stands as a pivotal focus within Intelligent Transportation Systems (ITS) development. It holds significant sway in reshaping human travel patterns, alleviating traffic congestion, and bolstering road safety. As a result, researchers and numerous high-tech enterprises place considerable importance on CAVs [4-6]. The maturation of CAVs technology promises to substantially enhance travel convenience and quality of life. Moreover, as cities continue to evolve, there is a pressing demand for safe and efficient modes of transportation [7-8].

CAVs utilize cutting-edge communication technology to establish seamless connections among vehicles and relevant elements via the Internet of Vehicles (IoVs). Originating from the broader Internet of Things (IoT), IoVs represent a specialized application tailored for vehicles, with a focus on dynamic vehicles as information perception nodes. Leveraging advancements in information and communication technology, IoVs facilitate comprehensive network integration, encompassing Vehicle-to-Vehicle (V2V), Vehicle-to-People (V2P), Vehicle-to-Infrastructure (V2I), and Vehicle-to-Cloud (V2C) communications [9-10]. IoVs leverage perception technology to gather real-time vehicle state information, which is then disseminated through wireless communication networks and modern intelligent information processing systems. This exchange includes critical data such as speed, position, acceleration, and braking, enabling CAVs to proactively warn nearby vehicles of potential safety hazards. As CAV technology continues to evolve, the realization of advanced traffic information services for intelligent vehicle control, traffic management, and decision-making becomes increasingly achievable [11].

Despite the anticipated benefits CAVs bring to Intelligent Transportation Systems (ITS), their effectiveness hinges on the security, precision, and reliability of sensor readings and network data [12-13]. Anomalies in sensor readings caused by malicious cyberattacks or faulty vehicle sensors can have severe consequences, potentially resulting in fatal accidents. Hence, it is imperative for CAVs to develop accurate real-time anomaly detection methods to mitigate such risks effectively.

Over recent years, there has been a notable rise in cybersecurity incidents across various sectors worldwide, often resulting in significant repercussions [14-16]. Consequently, there has been a heightened focus on the

*Corresponding Author: Zhenjiang Zhang; E-mail: zhangzhenjiang@bjtu.edu.cn

DOI: https://doi.org/10.70003/160792642024092505012

development of attack anomaly detection technologies [17]. For instance, Lee et al. [18] employed an extended Kalman filter to generate robust residuals amidst noisy conditions. These residuals, constructed from historical measurements, were then subjected to a parameter statistical method to identify network attacks within autonomous vehicle navigation systems, demonstrating exceptional performance in Integrated Navigation System/Global Navigation Satellite System (INS/GNSS) integration. Wang et al. [19] introduced a novel approach by integrating the adaptive extended Kalman filter (AEKF) with a car-following model. Their method aimed to enhance sensor reading stability within the nonlinear car-following model through time-delay mechanisms, thereby bolstering the safety of CAVs and enhancing their applicability in real-world scenarios. Similarly, Basiri et al. [20] devised two innovative detection methodologies leveraging Kalman state estimation: the rolling window detector (RWD) and the novel residue detector (NRD). These approaches, combined with enhanced Kalman filtering techniques, yielded reduced estimation errors, thereby advancing anomaly detection capabilities.

In recent years, deep learning technology has experienced rapid growth, spurred by the emergence of the Big Data era [21]. Numerous researchers have incorporated deep learning techniques into anomaly detection challenges within the realm of CAVs, yielding promising outcomes. For instance, Wyk et al. [22] integrated convolutional neural networks (CNNs) with Kalman filters and X2 detectors, achieving notable success in detecting anomalies within independently designed vehicle anomaly datasets. Khan et al. [13] proposed a multi-level intrusion detection framework featuring a bidirectional Long Short-Term Memory (Bi-LSTM) architecture centered on deep learning, effectively identifying intrusions within Intelligent Transportation Systems (ITS) while maintaining low false positive rates, thus enabling real-time intrusion differentiation. Almutlaq et al. [23] extracted rules from deep neural networks to develop an integrated intrusion detection system tailored for ITS. This system not only detects abnormal data but also categorizes specific types of anomalies upon detection. Javed et al. [24] employed a multi-level attention mechanism in combination with CNNs based on Long Short-Term Memory (LSTM), significantly enhancing the detection rate of anomalous instances across various magnitudes within the dataset.

However, existing deep learning methods still have areas for improvement. Firstly, deep learning methods generally use historical data for training, which makes it difficult for these methods to pay attention to unexpected situations in historical data. When there is a problem with the source of data (such as the vehicle entering the next roadside unit or experiencing temporary communication interruption with the current roadside unit), these methods may cause misjudgment because the input data of the deep learning model is missing. In addition, the vehicle may encounter restricted communication with the roadside during driving. In this case, the vehicle needs to perform anomaly detection of its own vehicle in the onboard equipment, which usually has lower computing power. Most anomaly detection models which use deep learning methods have a large number of parameters, and deploying them in onboard devices will prolong the time for vehicle data processing, thereby affecting the real-time performance of anomaly detection.

To address the aforementioned issues, we propose a vehicle anomaly detection model based on GAN networks, and use a pruning algorithm based on KL divergence theory to lightweight the model, enabling it to be deployed on vehicle terminal devices. Our main contributions are as follows:

- We propose a GAN based anomaly behavior recognition method that purposefully assigns weights to data of different dimensions during the generation of random noise by the generator. Specifically, for roadside data that vehicles cannot obtain stably, the discriminator should reduce the weight of these features to focus on the sensor data that vehicles can obtain stably. This method can accelerate the fitting speed of the network and improve accuracy during the network training process.
- We propose a model pruning algorithm based on KL divergence distribution. The algorithm determines the importance of each parameter during network training by calculating its distribution and variation process. For parameters with small distribution changes, random perturbations are added to give them a certain desire to explore, preventing them from falling into local optima and better determining the importance of each parameter in the detection process.
- We validated the algorithm proposed in this paper on the Safe Pilot Model Deployment (SPMD) dataset. Compared with existing algorithms, the algorithm proposed in this paper can better identify anomalies exhibited by vehicle data. In addition, experiments have shown that the pruning algorithm proposed in this paper can effectively reduce the number of model parameters while ensuring detection accuracy, enabling it to be deployed in resource limited bicycle intelligent systems.

The remainder of this paper is detailed as follows. Section II illustrates our proposed anomaly detection method. Section III introduces the data set used in the experiment and gives the experimental results. Finally, Section IV summarizes the work of this paper and highlights future directions.

2 Method

This section explains the basic principles of GAN and the basic concept of KL divergence. On this basis, we propose an improved GAN network and model pruning method.

2.1 GAN

The GAN network consists of two basic networks: a generator (often represented as G) and a discriminator (often represented as D). The network structure of the GAN is shown in Figure 1. The generator is used to generate new data, and the basis for generating the data is often a set of noise or random numbers. The discriminator is used to determine which is true between the generated data and the real data. The generator has no labels, which means that the

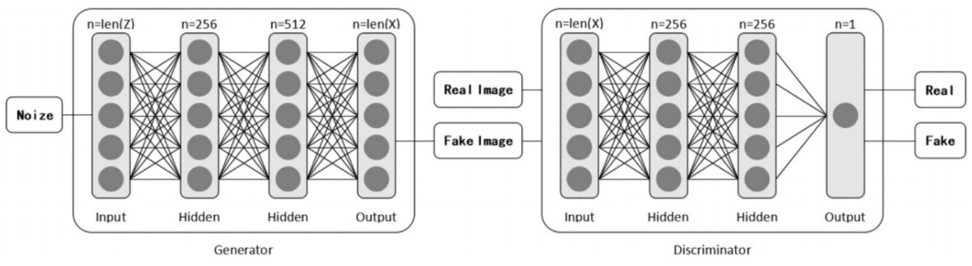


Figure 1. The constructure of GAN

generator is an unsupervised network. In the contrary, the discriminator has labels, which means the discriminator is a supervised network, with labels of false and true (0 and 1). During the training process, the goals of the generator and discriminator are contradictory, and this contradiction can be reflected in the accuracy of the discriminator's judgment. Generally speaking, the idea of GAN can be represented by formula 1.

$$\min_G \max_D V(D,G) = \min_G \max_D [E_{x\sim p_{data}(x)} \log D(x) + E_{z\sim p(z)} \log\left(1-D\left(G(z)\right)\right)] \tag{1}$$

Among them, V represents cross entropy loss, which is the function to be optimized; X represents real data, z represents data with added noise, $G(z)$ represents fake data generated in the generator, $D(x)$ and $D(G(z))$ represent the discrimination results of real and fake data, respectively.

Generators and discriminators compete and collaborate with each other through adversarial training. The goal of the generator is to deceive the discriminator, causing the generated samples to become closer to the real samples, so that the discriminator cannot accurately distinguish them. The goal of the discriminator is to classify samples as accurately as possible, making the difference between real samples and generated samples more apparent.

The adversarial training mechanism of the generator and discriminator enables GAN to learn the distribution of real data and generate diverse and creative samples. The game process between the generator and discriminator drives the learning and improvement of the model, making the generated samples more realistic.

2.1.1 Generator

The main function of a generator is to convert random noise vectors into realistic data samples. During the training process, it attempts to generate data which is similar to real data samples to deceive the discriminator. Through repeated iterative training, the generator learns to generate increasingly realistic samples, to the point where the discriminator ultimately cannot distinguish between generated data and real data.

Specifically, the goal of the generator is to maximize the error rate of the discriminator, which means that the discriminator cannot effectively distinguish between generated fake samples and real samples. This adversarial training process enables the generator to continuously improve the quality of its generated samples until it reaches the desired level.

2.1.2 Discriminator

The function of a discriminator is to receive samples (which can be real samples or samples generated by the generator) as input and predict the authenticity of the samples. The goal of a discriminator is to classify samples and determine whether they are real or generated.

We can fix the generator G and simplify Formula 1 to Formula 3.

$$\max_D V(D,G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{x\sim PG(x)}[\log\left(1-D(x)\right)] \tag{2}$$

In generally, discriminator should try to give higher scores to real data and lower scores to fake data, as shown in formula 2. False data resulted in lower scores, therefore the value of *log* (1 − $D(x)$) is higher. Wang et al. pointed out in the paper that the value of $V(D)$ can be given by formula 3 [25].

$$V(D^*) = -2\log 2 + 2JSD\left(P_{data}(x) \| P_G(x)\right) \tag{3}$$

Formula 3 provides the calculation process of $V(D)$, which allows the gradient propagation calculation of its parameters. However, GAN assumes that the parameters of each dimension are equally important. In practical scenarios of connected vehicles, a lot of vehicle data should be assigned different weights. For example, in V2I restricted scenarios, vehicles cannot stably obtain data from the roadside.

2.2 Assign Weights to Features

To address the aforementioned issues, we propose a dynamic weight allocation method that assigns different weights to each dimension according to its distributed features during the training process of the discriminator. The calculation process of weights is shown in formula 4.

$$\alpha_i = \frac{1}{p_i^2\sqrt{\sigma}} \tag{4}$$

Among them, p_i represents the proportion that 0 occupied in the current batch of the i-th feature, and σ represents the standard deviation of the distribution of the i-th feature. After distributing the weight, formula 3 can be rewritten as formula 5.

$$V(D^*) = -2\log 2 + 2\sum_{i=1}^{n} \alpha_i * JSD\left(P_{data}(x) \| P_G(x)\right) \quad \textbf{(5)}$$

Where n represents the number of features. We can optimize the discriminator to focus on stable data acquisition by using formula 6. The training process of the method proposed in this paper is shown in Table 1.

Table 1. The training and pruning process of our method

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k is a hyperparameter. We used k = 1, the least expensive option, in our experiments.

```
1:Initialize the distribution of each parameter in D
2:Initialize the weight w of each parameter in D
3: for number of training iterations do
4:     for $k$ steps do
5:         Sample minibatch of m noise samples
           $\{z_1, \dots \dots, z_m\}$ from noise prior $p_g(z)$
6:         Sample minibatch of m examples
           $\{x_1, \dots \dots, x_m\}$ from dataset where $x \sim p_{data}(x)$
7:         Calculate the weight, $\alpha = \{\alpha_1, \dots, \alpha_n\}$
8:         Train the discriminator,
           $\nabla_{\theta_d} \frac{1}{m}\sum_{i=1}^{m}[\log D(\alpha x_i) + \log\left(1 - D\left(G\left(\alpha z_i\right)\right)\right)]$
9:         if k % 100 == 0 do
10:            Fit the distribution of each parameter by their values in
               last 100 trainings, $\{p_i \sim D(\mu_i, \sigma_i), i = 1, 2, \dots \dots, n\}$
11:            Calculate the disturbance of each parameter,
               $\{d_i = \frac{1}{iterations\left(\left|\mu_0^i - u_t^i\right| + 1\right)}, i = 1, 2, \dots\dots, n\}$
12:            Calculate the weight of each parameter,
               $\{w_i' = \frac{1}{e^{-KL\left[D\left(p_0^i\right)\|D\left(p_t^i\right)\right]} + 1}, i = 1, 2, \dots\dots, n\}$
13:            Update the weight of each parameter, $w_i := w_i + w_i'$
14:        end if
15:    end for
16:    Sample minibatch of m noise samples {z1, zm} from moise
       prior $p_g(z)$
17:    Train the generator, $\nabla_{\theta_d} \frac{1}{m}\sum_{i=1}^{m}\left(1 - D\left(G\left(z_i\right)\right)\right)$
18:end for
19:obtain the detection model, that is the discriminator $D$
20:Sort the weight of each parameter in $D$
21:The gradient-based updates can use any standard gradient-
   baesd learning rule. We used Adam in our experiments.
```

2.3 KL- Divergence

KL-Divergence (Kullback-Leibler Divergence) is generally used to measure the distance between two probability distribution functions. The calculation process is shown in formula 6.

$$\begin{aligned} KL\left[P(X)\|Q(X)\right] &= \sum_{x \in X} p(x)\log\frac{P(X)}{Q(X)} \\ &= E_{x \sim P(x)}[\log\frac{P(X)}{Q(X)}] \end{aligned} \quad \textbf{(6)}$$

Due to the integration (continuous random variable) or summation (discrete random variable) of X in the calculation formula, KL-divergence is independent of a specific value of X and can also be denoted as $KL[P||Q]$. Note that the definition of KL-divergence is asymmetric with respect to $P(X)$ and $Q(X)$. According to the formula, KL-divergence does not satisfy symmetry, that is $KL[P(X)||Q(X)] \neq KL[Q(X)||P(X)]$. Thus, KL-divergence is clearly not a mathematical measure. The typical application scenario of KL-divergence is as follows: Suppose in a certain optimization problem, $P(x)$ is the true distribution, and $Q(X)$ is an approximate distribution used to fit $P(X)$. The algorithm attempts to modify $Q(X)$ to minimize the $KL[P(X)||Q(X)]$ between the two to achieve fitting $P(X)$ with $Q(X)$.

2.4 Model Pruning

The discriminator in GAN networks needs to constantly compete with the generator, which leads to the fact that it has a lot of parameters. In order to satisfy the lightweight requirements of vehicle terminals, it is necessary to perform lightweight processing on the discriminator. This paper proposes a model pruning method based on the KL divergence correlation principle, which evaluates the importance of each parameter of the model. During the model deployment process, different degrees of pruning can be performed on the model based on the actual computing power of the onboard equipment to meet the requirements of real-time detection.

Firstly, during the training process of the model, it is assumed that each parameter follows a certain distribution and gradually approaches its true distribution. Assuming $p \sim D(\theta ; x)$ is used to represent the distribution of a parameter, where θ represents the parameters that describe the distribution. In the optimization process of the model, each parameter approximates its corresponding true distribution. At this point, the importance of the distribution can be calculated by its degree of change. If the degree of variation of a parameter is very small, there may be two situations, 1) This parameter has no effect on the detection process; or 2) The initialized value of this parameter is already its optimal solution. To rule out the situation of 1, we also added random perturbations to this parameter during the training process to make it more exploratory. If this parameter plays an important role in the detection process and has a good initial value, it will return to its optimal position during the optimization process. On the contrary, if a parameter has no impact on the detection process, it will not return to the initial point after being disturbed. In addition, if the parameter distribution changes significantly during the training process, it indicates that the parameter has been effectively optimized, indicating that it plays an important role in the detection process. In this case, we only add a small perturbation to it so that it can escape local optima in the early stages of training.

For specific disturbances, we have also made the following assumptions. Firstly, the parameter should be closer to its optimal solution after training. Therefore, we use the trained distribution as the target distribution in KL divergence, while the pre training distribution is used as the distribution to be fitted. Secondly, considering the learning rate, we use the data of each parameter in 100 rounds of training to fit its corresponding distribution. Finally, according to the laws of large numbers, we use gaussian distribution as a benchmark to fit the distribution of parameters. In summary, the disturbance we propose can be represented by formula 7, and we use formula 8 to calculate the weights of parameters during pruning.

$$d = \frac{1}{epoch\left(\left|\mu_0 - \mu_t\right| + 1\right)} \tag{7}$$

$$w = \frac{2}{e^{-KL[D(p_0)\|D(p)]} + 1} + 1 \tag{8}$$

Among them, p_0 and p represent the distribution of the parameter before and after training, respectively, ***epoch*** represents the rounds of current epoch, μ_0 and μ_t represent the mean of the distribution of the parameter before and after training, respectively.

After the above process, each parameter is assigned a corresponding weight. When the model is migrated from the server to the vehicle terminal, the parameters are pruned from small to large weights, according to the actual computing power of the vehicle, until the complexity of the model meets the requirements of the vehicle. The training and pruning process of the method proposed in this paper is shown in Table 1.

3 Experiments

3.1 Dataset Description

In this paper, the proposed method was validated by Python version 3.7 using the Pytorch packages. The specification of the computer includes an Intel(R) Core(TM) i9-10900K CPU @ 3.70GHz, a 64-bit operating system, and a 64 GB memory. The GPU we used is RTX 3090.

In our experiment, we used the Research Data Exchange (RDE) database in the SPMD program. This dataset records driving data of over 2500 cars over a period of two years, with a huge amount of data widely used in various fields of CAV technology.

It is worth noting that the SPMD dataset includes both vehicle data and data that can be obtained from the roadside through V2I communication, which can also validate our proposed weight allocation method. The raw data in the SPMD dataset does not contain outliers, so we simulated several abnormal situations according to reference [26], namely instantaneous, constant, progressive drift, and deviation anomalies. Their specific representations are as follows:

1) Instant: a sharp, abrupt change between two consecutive readings;
2) Constant: temporary constant change not related to “normal” sensor reading;
3) Gradual drift: a small but persistent change occurring in a specific period;
4) Bias: a constant offset in a specific time.

3.2 Experiment Under Single Anomaly Type

3.2.1 Instant

Table 2 shows the performance evaluation results of MSALSTM-CNN [24], WKN-OC [26] and our method in different magnitude instant anomalies.

The outliers we add follow a Gaussian distribution, and then select specific coefficients to amplify (or shrink) them. In this paper, we selected a magnification of 25, 100, 500, 1000, 10000. From the table, it can be seen that the smaller the selected magnification, the worse the detection performance of the model. This is also in line with reality, that is, the more obvious the data anomaly, the better the detection effect. When the abnormal disturbance is very large, each algorithm exhibits similar high performance and can effectively detect abnormal data. In extreme cases, both the algorithm proposed in this paper and WKN-OC have extremely high accuracy, reaching 99.9%. As the magnification decreases, the detection accuracy of each algorithm decreases. When the coefficient drops to 25, where the anomaly amplitude achieves to base value + 25 * $N(0, 0.01)$, the performance of MSALSTM-CNN methods significantly decreases to 84.1%, while WKN-OC keeps a high accuracy of 96.5%. In this situation, our method performs slightly better than WKN-OC, which achieves a similar accuracy of 96.8%.

3.2.2 Constant

Table 3 presents the comparison results of constant anomalies. Unlike instantaneous anomalies that only vary in amplitude, constant anomaly simulations also have anomaly data with specific durations. This anomaly typically affects not only individual data, but also all data over a period of time. We designed five types of constant anomaly data with different durations and standard deviations [21] to compare the performance of different algorithms on this type of anomaly. From the table, it can be seen that under the same amplitude of anomalies, the longer the duration of anomalies, the better the model performance of various methods. When the duration of the anomaly is the same, the greater the amplitude change of the anomaly, the better the detection performance of the model. Overall, the method proposed in this paper has high accuracy, and as the amplitude decreases, the change in detection accuracy of the proposed method is relatively small. This indicates that this algorithm has good robustness and can detect relatively hidden anomalies.

3.2.3 Gradual Drift

Table 4 shows the comparison results of gradient drift anomalies. By adding a set of linear increments to the basic values of the sensor, gradual drift anomalies can be achieved. By adjusting the duration and amplitude of outliers, we designed four different types of anomalies and added them to the data. Unlike other anomalies, this type of anomaly does not have a significant impact on the data in the initial stage, but gradually becomes more severe over a period of time. This anomaly is covert and slow, making it difficult to detect and distinguish early on. From Table 3, it can be seen that when the amplitude decreases, the detection accuracy of the model also decreases because the difference between abnormal data and normal data is smaller. Both A and B will perform worse in this situation, but the algorithm proposed in this paper has not undergone significant changes, which fully demonstrates that the algorithm has better generalization ability in different abnormal situations.

3.2.4 Bias

Table 5 illustrates the anomaly detection performance of different models on bias anomalies. Deviation anomalies are achieved by increasing or decreasing fixed values based on the original data. As mentioned above, we set different standard deviations and durations to compare the detection performance of each model in different scenarios. From Table 5, it can be seen that when the duration is fixed and the amplitude increases, similar to constant anomalies, the more deviation from normal data, the better the detection performance of the model, which is also in line with the reality. When the amplitude of the anomaly is fixed, the longer the duration of the anomaly, the easier it is for the model to detect it. Compared to MSALSTM-CNN, the algorithm proposed in this paper can maintain its original detection performance even when the amplitude is reduced. Compared to WKN, the algorithm proposed in this paper also maintains a higher detection accuracy in shorter anomalies.

Table 2. Detection performance of instant anomaly type for the msalstm-cnn, wkn-oc and our method

	MSALSTM-CNN (%)				WKN-OC (%)				Our method (%)			
Anomaly magnitude	Acc	Sens	Prec	F1	Acc	Sens	Prec	F1	Acc	Sens	Prec	F1
Base value + 10000 * N(0, 0.01)	99.4	98.9	99.8	99.3	99.9	99.8	99.9	99.9	99.9	99.9	99.9	99.9
Base value + 1000 * N(0, 0.01)	99.0	98.2	99.8	98.7	99.9	99.3	99.9	99.6	99.9	99.5	99.8	99.6
Base value + 500 * N(0, 0.01)	96.0	99.8	97.9	99.0	99.9	99.2	99.8	99.4	99.5	99.0	99.5	99.2
Base value + 100 * N(0, 0.01)	95.8	89.6	98.4	93.8	99.0	90.7	98.8	94.4	98.4	91.3	97.4	94.3
Base value + 25 * N(0, 0.01)	84.1	54.6	98.1	70.2	96.5	65.8	96.1	72.9	97.1	68.2	96.9	80.1

Table 3. Detection performance of constant anomaly type for the msalstm-cnn, wkn-oc and our method

		MSALSTM-CNN (%)				WKN-OC (%)				Our method (%)			
Anomaly magnitude	Duration	Acc	Sens	Prec	F1	Acc	Sens	Prec	F1	Acc	Sens	Prec	F1
Base value + U(0, 5)	3	95.1	90.1	99.5	94.7	98.9	97.1	99.0	98.0	99.1	97.3	99.2	98.2
Base value + U(0, 5)	5	95.4	92.3	99.0	95.5	99.0	97.3	99.0	98.1	99.2	97.5	99.5	98.3
Base value + U(0, 5)	10	96.6	95.6	99.3	97.4	99.7	98.8	99.1	99.0	99.5	97.9	99.7	98.8
Base value + U(0, 3)	10	96.4	95.4	98.9	97.2	99.2	97.8	99.5	98.6	99.3	97.6	99.0	98.3
Base value + U(0, 1)	10	93.0	90.7	98.7	94.6	98.4	94.3	99.0	96.7	99.0	96.0	98.9	97.4

Table 4. Detection performance of gradual drift anomaly type for the msalstm-cnn, wkn-oc and our method

		MSALSTM-CNN (%)				WKN-OC (%)				Our method (%)			
Anomaly magnitude	Duration	Acc	Sens	Prec	F1	Acc	Sens	Prec	F1	Acc	Sens	Prec	F1
Base value + linespace(0, 4)	10	96.0	95.9	99.1	97.5	98.3	97.3	98.7	98.0	98.9	97.1	99.0	98.0
Base value + linespace(0, 4)	20	96.2	96.0	99.3	97.6	98.5	97.7	98.8	98.3	99.1	97.8	99.3	98.5
Base value + linespace(0, 2)	10	94.4	93.1	99.1	95.6	98.0	97.0	98.5	97.7	98.4	97.0	98.6	97.8
Base value + linespace(0, 2)	20	94.1	92.8	99.5	96.0	97.7	96.3	98.2	97.2	98.6	97.3	98.9	98.1

Table 5. Detection performance of bias anomaly type for the msalstm-cnn, wkn-oc and our method

		MSALSTM-CNN (%)				WKN-OC (%)				Our method (%)			
Anomaly magnitude	Duration	Acc	Sens	Prec	F1	Acc	Sens	Prec	F1	Acc	Sens	Prec	F1
Base value + U(0, 5)	3	95.1	90.1	99.5	94.7	98.9	97.1	99.0	98.0	99.1	97.5	99.2	98.3
Base value + U(0, 5)	5	95.4	92.3	99.0	95.5	99.0	97.3	99.0	98.1	99.4	97.8	99.2	98.5
Base value + U(0, 5)	10	96.6	95.6	99.3	97.4	99.7	98.8	99.1	99.0	99.8	97.8	99.4	98.6
Base value + U(0, 3)	10	96.4	95.4	98.9	97.2	99.2	97.8	99.5	98.6	99.5	97.4	99.1	98.2
Base value + U(0, 1)	10	93.0	90.7	98.7	94.6	98.4	94.3	99.0	96.7	98.8	96.9	99.0	97.9

Table 6. Detection performance of mixed anomaly types for the msalstm-cnn, wkn-oc and our method

		MSALSTM-CNN (%)		WKN-OC (%)		Our method (%)	
Anomaly magnitude	Sensors	Acc	F1	Acc	F1	Acc	F1
Instant, 1000 * N(1, 0.01)	1	91.3	78.1	97.0	83.4	97.1	86.3
	2	88.9	73.4	95.5	80.0	95.2	83.8
	3	88.9	64.4	91.6	75.8	92.2	80.4
Constant, U(0.5), d=10	1	95.6	90.4	99.2	95.4	99.0	95.5
	2	91.4	81.1	98.0	89.8	98.0	93.4
	3	91.3	77.8	97.7	88.5	97.1	92.0
GD, linespace(0, 4), d=20	1	93.6	84.3	97.6	92.2	97.0	93.4
	2	91.9	81.4	97.5	87.1	96.9	91.6
	3	89.9	76.3	97.4	83.5	96.4	88.0
Bias, U(0.5), d=10	1	96.0	90.5	98.3	94.1	99.5	93.8
	2	93.1	81.4	96.6	90.8	98.3	91.5
	3	90.4	76.2	94.9	84.9	97.0	87.4

3.3 Mixed Anomaly Type

In this section, we assume that multiple sensor data generate anomalies simultaneously, thus creating a mixed anomaly dataset. We compared the detection performance of three algorithms on a mixed anomaly dataset. Unlike individual anomalies, as the model did not use mixed anomaly data for training, each data in this dataset is new to the model. This helps to test the model's generalization ability when facing unknown exceptions.

In order to achieve multi-dimensional mixed anomalies, we selected three sensors, namely vehicle speed sensor (sensor1), vehicle GPS speed sensor (sensor2), and vehicle acceleration sensor (sensor3), and conducted attacks on these data to generate abnormal data.

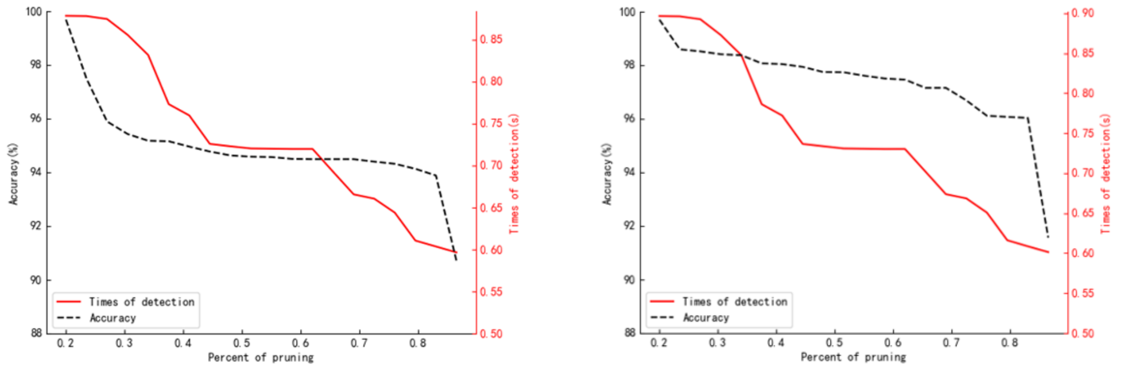
We use data with a single anomaly to train the model, and then test the performance of these models on a mixed anomaly dataset. Table 7 shows the detection performance of a single anomaly model on a mixed model.

From Table 7, it can be seen that the F1 metric of our method is significantly reduced in most cases, because the dataset contains too much data that the model has never seen before during training. It is worth noting that the models obtained using constant anomaly and bias anomaly datasets have similar accuracy and F1 metrics, because these two types of anomalies perform similarly on the data, and therefore the performance of the models is also similar. In addition, the model obtained using the instantaneous anomaly dataset performs the worst on mixed anomalies because the instantaneous anomaly data has a very high amplitude, which reaches 1000, and the instantaneous anomaly does not have a duration. Therefore, this anomaly is significantly different from other anomaly data. Finally, we can see from Table 7 that models obtained from datasets with acceleration anomalies often have the worst detection performance. This is because acceleration is different from the other two types of data, and it can only speculate whether anomalies have occurred based on changes in velocity. Therefore, this model often has poor generalization performance. This is also in line with the actual situation, because in practical scenarios, the value of acceleration has a wider range of values and more significant uncertainty. Therefore, we cannot determine whether there is an anomaly based on the value or trend of acceleration, but rather on the trend of velocity change to determine whether the acceleration is abnormal.

Table 7. Performance of our method in the case of mixed anomaly types

Anomaly magnitude	Sensors	Acc	Sens	Prec	F1
Instant, 1000 * N(1, 0.01)	1	97.12	77.00	98.12	86.29
	2	95.23	73.52	97.53	83.84
	3	92.18	70.29	94.01	80.44
Constant, U(0.5), d=10	1	98.97	91.83	99.54	95.53
	2	98.04	88.56	98.87	93.43
	3	97.13	86.40	98.35	91.99
GD, linespace(0, 4), d=20	1	97.02	89.13	98.12	93.41
	2	96.88	86.40	97.35	91.55
	3	96.43	81.53	95.48	87.96
Bias, U(0.5), d=10	1	99.53	91.78	95.86	93.78
	2	98.32	87.53	95.88	91.51
	3	97.01	81.46	94.17	87.36

In addition, in order to better evaluate the anomaly detection performance of our algorithm in mixed anomaly data, we also compared it with other algorithms in terms of accuracy and F1 index, and the results are shown in Table 6. In all cases, the algorithm presented in this paper demonstrated a high F1 score, although the accuracy was not always leading. A higher F1 index indicates that this algorithm has a higher detection rate for abnormal data, therefore it has better stability and generalization. Furthermore, even on the worst performing instantaneous anomaly data model, our algorithm still maintains an F1 score of 86.3%, which is the highest among all methods.



(a) The pruning experience on the model which was trained on instant anomaly dataset

(b) The pruning experience on the model which was trained on mixed anomaly dataset

Figure 2. The relationship between detection accuracy, detection time, and pruning ratio

3.4 Pruning Experience

In this section, we tested the effectiveness of the pruning method proposed in this paper. We set different pruning ratios A, and then reset the parameter of *p* percent in the discriminator to 0 to reduce floating-point operations and improve the speed of model detection. Figure 2 shows the changes in detection accuracy and detection speed of the model under different proportions. We conducted experiments on detecting the worst performing Instant anomalies and mixed anomalies.

From Figure 2, it can be seen that as the pruning ratio increases, the detection accuracy of the model decreases and the detection speed becomes faster. This is also in line with the actual situation, as pruning operations reduce the floating-point operations in the model detection process. It is worth noting that the decreasing trend of accuracy and the decreasing trend of detection time are not the same. This means that we can find a reasonable pruning ratio, so that the network can maintain high detection accuracy while having a faster detection speed. Specifically, we can identify some points where the detection accuracy is rapidly decreasing. From Figure 2(a), when the pruning ratio is greater than 83% percent, the decreasing trend of accuracy significantly increases. On the other hand, when the pruning ratio reaches 85%, the accuracy begins to decline at a faster rate in Figure 2(b). In addition, there is a significant improvement in detection speed throughout the entire pruning process. This is reasonable because the number of operations and pruning ratio are linearly related. Therefore, in practical tasks, we can set the pruning ratio $p \in [0, 0.83]$, and then determine the specific value based on the actual computing power of the onboard equipment. However, the pruning ratio should not exceed 0.83, as this can lead to a rapid decrease in the detection accuracy of the model, which cannot meet the accuracy requirements.

4 Conclusion

In order to better detect abnormal behavior data of vehicles, a new anomaly detection model is proposed. The algorithm is first based on the GAN network, modifying the discriminator structure of the network and assigning weights to each feature, so that the model can effectively identify different types of data anomalies. In addition, this article also proposes a pruning method that enables the model to be deployed on vehicle terminal devices with limited computing power. The results show that the algorithm proposed in this article has high recognition accuracy (whether for individual or mixed anomalies) and can effectively reduce detection time. In this article, we attack specific sensors to generate abnormal data. In the subsequent work, we will attack more sensors and construct more types of attack patterns to increase the generalization of the model.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant 62173026.

References

[1] Y. Lv, Y. Duan, W. Kang, Z. Li, F. Wang, Traffic flow prediction with big data: A deep learning approach, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 16, No. 2, pp. 865–873, April, 2015.

[2] J. Zhang, F. Wang, K. Wang, W. Lin, X. Xu, C. Chen, Data-driven intelligent transportation systems: A survey, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 12, No. 4, pp. 1624–1639, December, 2011.

[3] S. Feng, Z. Song, Z. Li, Y. Zhang, L. Li, Robust platoon control in mixed traffic flow based on tube model predictive control, *IEEE Transactions on Intelligent Vehicles*, Vol. 6, No. 4, pp. 711–722, December, 2021.

[4] H. Hashim, M. Omar, Towards Autonomous Vehicle Implementation: Issues and Opportunities, *Society of Automotive Engineers*, Vol. 1, No. 2, pp. 111–123, May, 2017.

[5] S. Beregi, S. Avedisov, C. He, D. Takacs, G. Orosz, Connectivity-based delay-tolerant control of automated vehicles: Theory and experiments, *IEEE Transactions on Intelligent Vehicles*, Vol. 8, No. 1, pp. 275–289, January, 2023.

[6] J. Zhan, Z. Ma, L. Zhang, Data-driven modeling and distributed predictive control of mixed vehicle platoons, *IEEE Transactions on Intelligent Vehicles*, Vol. 8, No. 1, pp. 572–582, January, 2023.

[7] J. Guanetti, Y. Kim, F. Borrelli, Control of connected and automated vehicles: State of the art and future challenges, *Annual Reviews in Control*, Vol. 45, pp. 18–40, 2018.

[8] E. Ohn-Bar, M. Trivedi, Looking at humans in the age of self-driving and highly automated vehicles, *IEEE Transactions on Intelligent Vehicles*, Vol. 1, No. 1, pp. 90–104, March, 2016.

[9] Y. Wang, N. Masoud, A. Khojandi, Anomaly detection in connected and automated vehicles using an augmented state formulation, *2020 Forum on Integrated and Sustainable Transportation Systems*, Delft, Netherlands, 2020, pp. 156–161.

[10] M. Hasan, S. Mohan, T. Shimizu, H. Lu, Securing vehicle-to-everything (V2X) communication platforms, *IEEE Transactions on Intelligent Vehicles*, Vol. 5, No. 4, pp. 693–713, December, 2020.

[11] D. Cao, X, Wang, L. Li, C. Lv, X. Na, Y. Xing, X. Li, Y. Li, Y. Chen, F. Wang, Future directions of intelligent vehicles: Potentials, possibilities, and perspectives, *IEEE Transactions on Intelligent Vehicles*, Vol. 7, No. 1, pp. 7–10, March, 2022.

[12] P. Laso, D. Brosset, J. Puentes, Analysis of quality measurements to categorize anomalies in sensor systems, *2017 Computing Conference*, London, UK, 2017, pp. 1330–1338.

[13] I. Khan, N. Moustafa, D. Pi, W. Haider, B. Li, A. Jolfaei, An enhanced multi-stage deep learning framework for detecting malicious activities from autonomous vehicles, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 23, No. 12, pp. 25469–25478, December, 2022.

[14] D. Zhang, Q. Wang, G. Feng, Y. Shi, A. Vasilakos, A survey on attack detection, estimation and control of industrial cyber–physical systems, *ISA Transactions*, Vol. 116, pp. 1–16, October, 2021.

[15] D. Zhang, Z. Ye, G. Feng, H. Li, Intelligent event-based fuzzy dynamic positioning control of nonlinear unmanned marine vehicles under DoS attack, *IEEE Transactions on Cybernetics*, Vol. 52, No. 12, pp. 13486–13499, December, 2022.

[16] S. Nifakos, K. Chandramouli, C. Nikolaou, P. Papachristou, S. Koch, E. Panaousis, S. Bonacina, Influence of human factors on cyber security within healthcare organisations: A systematic review, *Sensors*, Vol. 21, No. 15, Article No. 5119, August, 2021.

[17] Z. Ju, H. Zhang, X. Li, X. Chen, J. Han, M. Yang, A survey on attack detection and resilience for connected and automated vehicles: From vehicle dynamics and control perspective, *IEEE Transactions on Intelligent Vehicles*, Vol. 7, No. 4, pp. 815–837, December, 2022.

[18] S. Lee, Y. Cho, B. Min, Attack-aware multi-sensor integration algorithm for autonomous vehicle navigation systems, *2017 IEEE International Conference on Systems, Man, and Cybernetics*, Banff, Canada, 2017, pp. 3739–3744.

[19] Y. Wang, N. Masoud, A. Khojandi, Real-time sensor anomaly detection and recovery in connected automated vehicle sensors, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 22, No. 3, pp. 1411–1421, March, 2021.

[20] M. Basiri, J. Thistle, J. Simpson-Porco, S. Fischmeister, Kalman filter based secure state estimation and individual attacked sensor detection in cyber-physical systems, *2019 American Control Conference*, Philadelphia, USA, 2019, pp. 3841–3848.

[21] L. Alzubaidi, J. Zhang, A. Humaidi, A. Dujaili, Y. Duan, O. Shamma, J. Santamaría, M. Fadhel, M. Amidie, L. Farhan, Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions, *Journal of Big Data*, Vol. 8, No. 1, pp. 1–74, March, 2021.

[22] F. Wyk, Y. Wang, A. Khojandi, N. Masoud, Real-time sensor anomaly detection and identification in automated vehicles, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 21, No. 3, pp. 1264–1276, March, 2020.

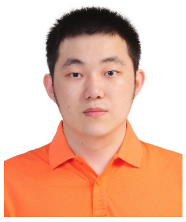
[23] S. Almutlaq, A. Derhab, M. Hassan, K. Kaur, Two-stage intrusion detection system in intelligent transportation systems using rule extraction methods from deep neural networks, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 24, No. 12, pp. 15687-15701, December, 2023.

[24] A. Javed, M. Usman, S. Rehman, M. Khan, M. Haghighi, Anomaly detection in automated vehicles using multistage attention-based convolutional neural network, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 22, No. 7, pp. 4291–4300, July, 2021

[25] H. Wang, T. Pu, X. Li, C. Liu, J. Wu, J. Yang, Z, Zhang, Y. Lu, Q. Wang, L. Song, H. Chiu, J. Ao, X. Liu, High-Performance Normally-Off Operation p-GaN Gate HEMT on Free-Standing GaN Substrate, *IEEE Transactions on Electron Devices*, Vol. 69, No. 9, pp. 4859–4863, September, 2022.

[26] Z. He, Y. Chen, H. Zhang, D. Zhang, WKN-OC: A New Deep Learning Method for Anomaly Detection in Intelligent Vehicles, *IEEE Transactions on Intelligent Vehicles*, Vol. 8, No. 3, pp. 2162–2172, March, 2023.

Biographies



Wenhui Wang has received his Master degree in Beijing Institute of Petrochemical Technology, and he is now pursuing Ph. D degree in Beijing Jiaotong University. His current research interests include, but are not limited to, edge computing, internet of vehicles and federated learning.

Qiang Zhu received the PhD degree from University of Science and Technology of China in July 2005. He is currently a Professor in the School of Mathematics and Statistics at the Xidian University of Xi'an, China. His research interests include graph theory and fault tolerant computing including system-level diagnosis and interconnection networks.

Chia-Wei Lee received the PhD degree from National Cheng Kung University, Tainan, Taiwan, in November 2009. He is currently an Associate Professor in the Department of Computer Science at the University of Taipei, Taiwan. His research focuses include graph theory and algorithms, as well as system-level diagnosis and interconnection networks.

Zhenjiang Zhang received the Ph.D. degree in communication and information systems from Beijing Jiaotong University, where he was an Associate Professor from 2008 to 2013. His research interests include cognitive radio, wireless sensor networks, and edge computing. He has been the guest editor of IET Communications, Sensors, and International Journal of Distributed Sensor Networks.