# Exploiting the Potential Anomaly Detection in Automobile Safety Data with Multi-type Neural Network

Quanlong Guan[1,3,6], Tian Zhang[1,6], Xiujie Huang[1], Yuansheng Zhong[2], Cuifeng Du[4], Changjiang Liu[2], Zhefu Li[1(✉)], Guanghui Zhang[1,3,6], Xiaofeng Wu[5(✉)], and Zhifei Duan[7]

[1] Jinan University, Jinan 510632, Guangdong, China
{Gql,t_xiujie,lzf}@jnu.edu.cn
[2] Key Laboratory of Safety of Intelligent Robots for State Market Regulation, Guangdong Testing Institute of Product Quality Supervision, Jinan, Guangzhou, China
zhongys@gqi.org.cn
[3] Guangdong Key Laboratory of Data Security and Privacy Preserving, Jinan, Guangdong, China
[4] Cetc Potevio Science and Technology Co., Ltd., Jinan, Guangdong, China
[5] Guangzhou Polytechnic of Sports, Jinan, Guangzhou, China
[6] Guangdong-Macao Advanced Intelligent Computing Joint Laboratory, Jinan, China
[7] Guangzhou XPeng Motors Technology Co., Ltd., Jinan, Guangdong, China
duanzf@xiaopeng.com

**Abstract.** As an internal network widely used in automobiles, the automotive CAN bus network lacks effective security protection mechanisms and is vulnerable to network hackers, posing a serious threat to the safety of vehicles and drivers. The automotive intrusion detection system provides effective protection for the security of the automotive CAN network. To address the shortcomings of current intrusion detection algorithms, such as long application time and incomplete detection types, GIDPS and TIDPS models are proposed to perform supervised multi-classification experiments on vehicle intrusion data. Then, the above model is migrated to the ROAD dataset for verification, and the advantages of the new model in terms of time and accuracy compared with the old model are analysed based on the results. The proposed GIDPS and TIDPS models achieve better results than previous models in terms of synthesis. The new models provides a certain reference value for improving the level of automotive network security. They could be applied to domestic or cross-border automotive markets.

**Keywords:** CAN bus network · Intrusion detection · Network hackers · Multi-classification


A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 593, pp. 489–501, 2024.
https://doi.org/10.1007/978-3-031-63989-0_26

## 1 Introduction

In-vehicle network systems play an important role in connected smart cars [1]. Not only do they need to connect and control each major electronic control unit (ECU) in the car, but they also need to communicate with various external vehicles, traffic systems and cloud platforms [2,3]. The CAN bus system has the advantages of high real-time response, long transmission distance, good interaction effect and good economy. However, the CAN bus protocol does not consider information security issues, and there is no encryption mechanism and identity discrimination mechanism, so hackers can easily obtain information in the car, forge identities to send information, and tamper with bus data, which threatens the security of the car [4,5].

Therefore, it is necessary to improve the detection level of CAN network data in the car and to find illegal intrusion information in time. Intrusion Detection System (IDS) is one of the important security measures to ensure the safe communication of CAN network. The intrusion detection technology it adopts is directly related to the effect of intrusion detection [6]. However, in previous works, the vehicle security intrusion detection model has problems such as slow computation speed and poor detection effect of certain types of attacks.

Based on the problems existing in previous models, we first pre-processed the CAN message data in the car to speed up the model's data processing time. Then, on this basis, we also investigated a new TIDPS model [7,8] based on the self-attention mechanism and a GIDPS model [9] that can be used for time series analysis. Different from the binary classification method of previous models, the model innovatively adopts a multi-classification method to further improve the computational speed of the model and optimise the processing effect of attack types. Finally, the newly proposed model is migrated to a new dataset for comparison, which verifies the effectiveness of our proposed GIDPS and TIDPS models to improve the accuracy and efficiency levels of anomaly detection.

The remaining chapters of this article are organised as follows: Sect. 2 gives an overview of the related work done in the past, analyses the existing problems, and draws out the key points to pay attention to in this article. Section 3 mainly presents two related datasets, including their data structure, type of attack, data volume and preprocessing procedure. Section 4 introduces the structure and computational principles of the three correlation network models. Section 6 is a summary of this thesis, which summarises the work done and the results achieved in this thesis, and introduces the future research direction.

## 2 Related Work

At present, the more commonly used IDS intrusion detection methods include intrusion detection methods based on statistics and intrusion detection methods based on machine learning [9,10]. During the detection process, if the statistic is found to exceed the threshold, it is judged to be abnormal.

Müter et al. [10] first proposed an entropy-based intrusion detection method that works well for detecting flooding and replay attacks, but has a high false

positive rate for tampering and spoofing attacks, indicating that it can be detected by information theory. Since then, there have been many studies to improve and integrate new variables based on information entropy. For example, in 2021, Zhang et al. [11] proposed a CAN network intrusion detection mechanism based on relative entropy, which replaced the time window in the anomaly detection mechanism based on information entropy with the data window, so it can be detected by analysing non-exception for periodic data frames. However, statistics-based intrusion detection methods may have problems such as modelling difficulties and threshold determination difficulties [4].

The intrusion detection method based on machine learning uses supervised or unsupervised machine learning algorithms to train normal and abnormal data to learn their respective characteristics and identify outliers for intrusion detection. Seo [12] proposed the GIDS (GAN-based vehicle network IDS) model, GAN (Generative Adversarial Network, Generative Adversarial Network), which uses the generation network and the identification network to fight against each other to achieve the goal of co-evolution. Song et al. [13] proposed a DCNN-based IDS to build a simplified ResNet to reduce the complexity of the network. Hu et al. [14] converted the data into two-dimensional images using their own coding methods, and used different convolutional neural networks to extract and train the data features. The deep neural network has the advantages of strong self-learning and low delay. According to tests, it has a high detection accuracy, but it relies on a large amount of data for learning, requires more computational resources and lacks an explanation mechanism.

It can be seen that the previous machine learning detection models have problems such as slow computation speed and poor detection effect for certain types of attacks. Therefore, the work in this paper mainly uses the neural network model based on the self-attention mechanism to pre-process the vehicle intrusion dataset, and then combines appropriate data encoding and multi-classification methods to propose several new models to solve the above problems.

## 3 Data Analysis and Preprocessing

The dataset used in this experiment is the real vehicle network HCRL data set [13,15] collected and published by Korea Hacker and Countermeasures Research Laboratory (HCRL) and the real vehicle ROAD data set [16] recorded by Oak Ridge National Laboratory (ORNL). The following will briefly introduce the two data sets and data preprocessing.

### 3.1 HCRL Dataset

The HCRL dataset [13,15] was collected in a real in-vehicle network environment, including normal network communication behaviour and many different types of attacks. Details are given in Table 1.

**Table 1.** HCRL dataset attack description

| Attack name | Attack method | Attack interval |
|---|---|---|
| DoS attack | Inject a message with CAN ID "0000" once | 0.3 ms |
| Fuzzing attack | Inject a completely random CAN ID and DATA message | 0.5 ms |
| Spoofing attack | Inject messages for specific CAN IDs related to RPM or gear information | 1 ms |

Then by intercepting the CAN network message sequence dataset with a window of step size 10, marking the frame containing the 'T' mark as the corresponding attack frame, extracting the ID of each frame and drawing the corresponding ID curve, see Fig. 1 below. The ordinate in the figure is the number of IDs, and the abscissa is the step size.

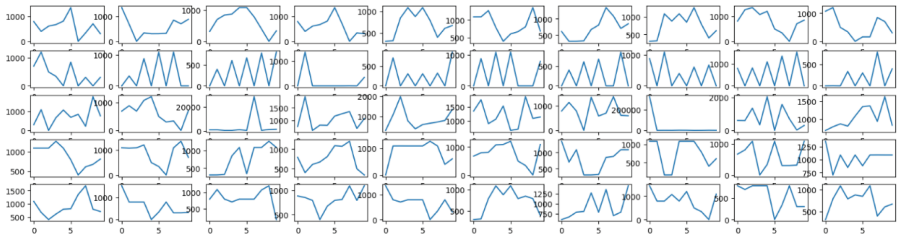

**Fig. 1.** The partial ID curves of the HCRL, from top to bottom: normal data, Dos attack, Fuzzy attack, gear attack, RPM attack

Observing Fig. 1, it can be seen that the ID curve characteristics of DoS attacks are relatively obvious, showing a clear jagged shape, and can be easily classified according to timing and ID. The remaining three features are not obvious and need to be combined with other data extracted from the network for better classification.

**Preprocessing the HCRL Dataset.** First, fill in the NAN value of the data. Because there is no data in some bits of some messages, there are NAN values, so the NAN bits are filled as 0. Then, we performed data type conversion and a number system conversion. In this paper, the data is converted directly to decimal and each byte of the ID and DATA fields is used as a feature, making a total of 10 features. Next, we calculated the frame interval. Messages have time stamps and there is a time difference between messages which is not uniform. Therefore, the difference between two adjacent messages is calculated as a feature. We then normalized the data to eliminate the effects of anomalies. In particular, the frame interval, because the gap is almost milliseconds, the order of magnitude is small and needs to be increased. Finally, we label the data. In this dataset, the time slice containing the 'T' tag is marked as the sequence number of the corresponding attack.

**Table 2.** HCRL data step division result

| Step size | Normal | Dos | Fuzzy | Gear | RPM |
|---|---|---|---|---|---|
| 1 | 3404896(200000) | 233004 | 196680 | 180909 | 178914 |
| 5 | 466153(100000) | 84966 | 89427 | 96038 | 102299 |
| 10 | 213629(50000) | 42632 | 51247 | 55838 | 56096 |

From Table 2 we can see that the number of attacks decreases as the step size increases, indicating that the attacks are not sparse.

### 3.2 ROAD Dataset

The ROAD dataset [16] provides the following realistic attacks: one obfuscation attack, many maximally stealthy target fabrication attacks, and two that do not contain faked advanced attacks on messages.

From Table 3, the message with ID 6E0 on the left has two lines, but the underline is the attack message modified by the attacker, and the original message is removed from the dataset to disguise it. With such a spoofed dataset, frequency-based methods will almost certainly fail to provide accurate detection (Table 4).

**Table 3.** ROAD dataset simulates masquerading attack

| Message conflict | Remove conflict |
|---|---|
| 9.196895) FFF#0000000000000000 | 9.196895) FFF#0000000000000000 |
| 9.196896) 6E0#03AF03A603A403A6 | 9.197928) 6E0#595945450000FFFF |
| 9.197928) 6E0#595945450000FFFF | 9.199034) 354#2016800000012080 |

**Table 4.** HCRL data step division result

| Step size | Normal | Correlated | Max_engine | Speedometer | Light_off | Light_on |
|---|---|---|---|---|---|---|
| 1 | 10000 | 5488 | 43 | 11689 | 5476 | 8032 |
| 5 | 10000 | 5488 | 43 | 11689 | 5476 | 8032 |
| **10** | 10000 | 5400 | 43 | 11680 | 5476 | 8032 |

**Preprocessing the ROAD Dataset.** After our data pre-processing operation, we can know that the number of attacks in this data set is small and sparse.

## 4 Building the Intrusion Detection Model

The experimental data set and data pre-processing operations have been described in the third section, and in this section we mainly establish the vehicle intrusion detection model. Therefore, according to the time characteristics of the

above data, we established the time series models GIDPS and TIDPS based on GRU [17] and Transforme [18], and compared with the previous LSTM-based model (LIDPS) [19–21]. Therefore, different types of attack data and normal data can be effectively distinguished, and the intrusion attack of the automotive CAN network can be detected.

### 4.1 LIDPS Model

Previous [19–21] mostly used LSTM neural networks for binary classification tasks, but this paper proposes an LSTM-based LIDPS classification model for multi-classification tasks of vehicle CAN messages, which improves the classification accuracy.

$$f_t = \delta * (W_f * [h_{t-1}, x_t] + b_f) \tag{1}$$

Equation (1) combines the output of the previous layer and the input of this layer to multiply the weight, plus the bias, and passes the result through the sigmoid activation function. In this, $\delta$ is the sigmoid activation function, $h_{t-1}$ is the hidden information of the previous layer, $x_t$ is the input of this layer, $W_f$ is the weight factor and $b_f$ is the bias function.

$$\begin{cases} i_t = \delta * (W_i * [h_{t-1}, x_t] + b_i) \\ C_t^{\sim} = tanh * (W_f * [h_{t-1}, x_t] + b_f) \\ C_t = f_t * C_{t-1} + i_t * C_t^{\sim} \end{cases} \tag{2}$$

As shown in Eq. (2), after passing through the tanh activation function, the state value is compressed between –1 and 1 and then multiplied by the output through the sigmoid to obtain the output. Among them, the tanh function is used to help adjust the value flowing through the network and play a role in sorting out. $C_t^{\sim}$ is the memory information of the previous layer.

$$\begin{cases} o_t = \delta * (W_o * [h_{t-1}, x_t] + b_o) \\ h_t = o_t * tanh(C_t) \end{cases} \tag{3}$$

In the output gate, the hidden information of the upper layer in Eq. (3) and the input of this layer first pass through the sigmoid function to determine the output content, and then multiply the memory information passed through the tanh function to obtain the hidden part of the next layer.

### 4.2 GIDPS Model

Gated recurrent unit (GRU) [22] is a recurrent neural network structure. Compared with LSTM [19], GRU [22] only has two gating units, the reset gate and the update gate. Therefore, based on the previous intrusion detection model based on GRU [22], in order to better compare and highlight the TIDPS model, this paper proposes a GIDPS model that can perform multiple classification tasks.

$$\begin{cases} r_t = \delta * (W_r * [h_{t-1}, x_t] + b_r) \\ z_t = \delta * (W_z * [h_{t-1}, x_t] + b_z) \end{cases} \tag{4}$$

In Eq. (4), the input at time t and the state of the hidden layer at time t-1 contain information about previous nodes. After passing through the sigmoid function, the two inputs become the control signal for remembering or forgetting.

$$h_t^{\sim} = tanh * (W_h * [r_t * h_{t-1}, x_t] + b_h) \tag{5}$$

The calculation method of the reset gate is shown in Eq. (5). When $r_t$ tends to zero, the previous hidden information is discarded, leaving only the input, and when $r_t$ tends to 1, the previous information is added to the current information.

$$h_t = (1 - z_t) * h_{t-1} + z_t * h_t^{\sim} \tag{6}$$

The calculation method of the update gate is shown in the above Eq. (6). By selectively forgetting the hidden state of the upper layer and then selectively remembering the hidden state of the current node.

### 4.3 TIDPS Model

The TIDPS model proposed in this paper is based on Transformer [18], using a self-attention mechanism (self-attention) to dynamically generate weights for different connections between the input and output of the same layer network.

$$\begin{cases} Q(W_Q * X), K(W_K * X), V(W_V * X) \\ Attention(Q, K, V) = softmax(\frac{Q * K^T}{\sqrt{d_k}}) * V \end{cases} \tag{7}$$

Among them, Q, K and V are query vector sequence, key vector sequence and value vector sequence, respectively. The input X is projected into three different spaces by the linear transformation of Eq. (7), and Q, K and V can be obtained. $W_Q$, $W_K$ and $W_V$ are three learnable parameter matrices.

$$MultiHead(Q, K, V) = Concat(Attention(Q_i, K, V)...Attention(Q_k, K, V)) \tag{8}$$

The multi-head attention mechanism is to repeat the same operation multiple times, and the results are finally concatenated. Different Q pay attention to different features, evaluate the input information from different angles and then integrate them as shown in Eq. (8).

As shown in Fig. 2, the complete Transformer consists of two parts, the encoder and the decoder, but in the classification task, only the encoder part is needed. The position encoding is to add a position information to the input sequence to enhance the timing characteristics of the sequence.

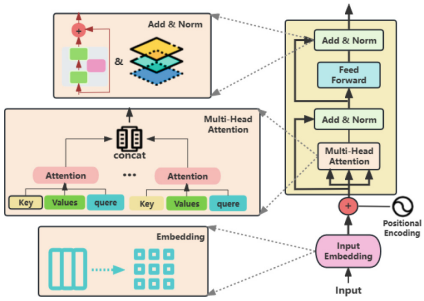


**Fig. 2.** TIDPS model encoder

## 5 Experimental Results and Analysis

The CPU used in this experiment is AMD Ryzen 5 5600H with Radeon Graphis, no GPU is used, the operating system is Windows 10, the IDE is jupyter, the Python version is 3.9, and the pytorch deep learning framework is used.

### 5.1 Experiments on the HRCL Dataset

As the dataset is large, the model converges quickly. The specific parameters are given in Table 5 and the ratio of training set to test set is 3:1.

To find a more suitable model and improve the training accuracy of the model, we added some modules for ablation experiments: the 1 at the end indicates the original model, and the data has gone through three rounds of the modules introduced above. After the model is finished, we expand the data into one dimension, perform a full connection operation, and finally map to 5 categories. 2 is to expand the data, then add a fully connected layer with a length of 256 and then map to 5 categories, and 3 is to add a one-dimensional convolutional layer and then go through the fully connected layer.

**Table 5.** Training parameters based on HRCL dataset

| Parameter | Batch Size | Learning Rate | Optimizer | Classes | Loss |
|---|---|---|---|---|---|
| Value | 64 | 0.0005 | Adam | 5 | Cross Entropy |

DES of the Table 6 shows the training results using the early stopping strategy and using decimal data, where sql_len represents the sequence step size. It can be seen from the data in the table that the newly added modules can work when the hidden size is small, but as the hidden size increases, although some will speed up the convergence of the model, the accuracy is lower than when no new modules are added.

**Table 6.** The results of training the model on the HCRL dataset

| Cond[1] | Sqe[2] | Hs[3] | $LIDPS_1$ | $LIDPS_2$ | $LIDPS_3$ | $GIDPS_1$ | $GIDPS_2$ | $GIDPS_3$ | $TIDPS_1$ | $TIDPS_2$ | $TIDPS_3$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DES[4] | **5** | **32** | 0.9815 | 0.9933 | 0.9898 | 0.9859 | 0.9853 | 0.9655 | 0.9924 | 0.9922 | 0.9879 |
| | | | 00:55 | 00:51 | 01:00 | 00:45 | 00:50 | 00:36 | 01:53 | 01:33 | 01:24 |
| | | **64** | 0.9949 | 0.9962 | 0.9891 | 0.9935 | 0.9903 | 0.9900 | 0.9939 | 0.9918 | 0.9723 |
| | | | 01:56 | 01:56 | 00:54 | 01:27 | 00:41 | 00:43 | 01:25 | 01:32 | 00:39 |
| | | **128** | 0.9963 | 0.996 | 0.994 | 0.9966 | 0.9948 | 0.9957 | 0.9972 | 0.9946 | 0.9933 |
| | | | 02:25 | 02:15 | 01:46 | 02:49 | 01:17 | 02:06 | 02:09 | 01:44 | 01:27 |
| | **10** | **32** | 0.9913 | 0.9933 | 0.9897 | 0.9802 | 0.9853 | 0.9577 | 0.9924 | 0.9922 | 0.9879 |
| | | | 01:34 | 00:51 | 00:46 | 00:34 | 00:50 | 00:32 | 01:53 | 01:33 | 01:24 |
| | | **64** | 0.9963 | 0.9962 | 0.994 | 0.9936 | 0.9903 | 0.9933 | 0.9939 | 0.9918 | 0.9723 |
| | | | 02:15 | 01:56 | 01:19 | 01:32 | 00:41 | 01:26 | 01:25 | 01:32 | 00:39 |
| | | **128** | 0.9969 | 0.996 | 0.9961 | 0.9958 | 0.9948 | 0.9958 | 0.9972 | 0.9946 | 0.9933 |
| | | | 03:26 | 02:15 | 02:35 | 02:17 | 01:17 | 02:40 | 02:09 | 01:44 | 01:27 |
| PDM[5] | **10** | **128** | pooling epoch | | | | | | | | |
| | | | 0.9742 | \ | \ | 0.9793 | \ | \ | 0.9193 | \ | \ |
| | | | 06:02 | \ | \ | 04:21 | \ | \ | 02:13 | \ | \ |
| | | | non-pooling epoch | | | | | | | | |
| | | | 0.9973 | \ | \ | 0.9967 | \ | \ | 0.9962 | \ | \ |
| | | | 05:37 | \ | \ | 03:57 | \ | \ | 02:24 | \ | \ |
| TDM[6] | **5** | **32** | 0.9931 | 0.9952 | 0.9907 | 0.9913 | 0.9887 | 0.992 | 0.994 | 0.9946 | 0.9943 |
| | | | 02:31 | 03:11 | 02:46 | 01:36 | 02:04 | 01:55 | 01:58 | 02:07 | 02:04 |
| | | **64** | 0.9970 | 0.9970 | 0.9969 | 0.9944 | 0.9942 | 0.9953 | 0.9952 | 0.9956 | 0.9968 |
| | | | 04:06 | 05:04 | 04:34 | 02:45 | 03:26 | 03:09 | 02:27 | 02:14 | 02:10 |
| | | **128** | 0.9971 | 0.9970 | 0.99768 | 0.997 | 0.9964 | 0.9967 | 0.9972 | 0.9965 | 0.9961 |
| | | | 08:32 | 10:29 | 09:02 | 05:45 | 06:34 | 06:11 | 02:43 | 02:29 | 02:26 |

[1] Condition
[2] Sql_len
[3] Hidden size
[4] Training results on the HCRL dataset during early stopping
[5] Increase the training results of pooling different models on the HCRL dataset
[6] Training results of different models on the HCRL dataset

It can be seen that the accuracy and time of our proposed TIDPS are quite stable. Among many results, the convergence speed of GIDPS is the fastest, but the accuracy is just the opposite. This is the price of its model being simpler than the previous LIDPS model.

Maximum pooling can reduce the dimensionality of the data and reduce the amount of computation. There are such practices in previous work, but from the pooling experimental data PDM in the Table 6, it can be seen that not only the training time increases, but also the accuracy drops a lot, so the pooling layer does not play a positive effect.

The TDM of Table 6 shows that with almost equal accuracy, the accuracy of LIDPS and GIDPS increases as the hidden size increases, but the time also increases as the hidden size doubles, while the time for TIDPS was consistently around 2 min. The ratio of time required for LIDPS, GIDPS, and TIDPS to achieve similar accuracy is approximately 18:12:5. The time for training data

may not be long, but the algorithm is used for real-time detection, i.e. the data generated all the time is fed into the network to perform calculations. Although the computation time is very short, the long accumulation time will cause a large time delay, so that even if an anomaly is detected, it cannot be processed in time. Therefore, the use of TIDPS or GIDPS has a great effect in reducing the algorithm time. It can be seen that our proposed GIDPS and TIDPS models have significantly improved in terms of accuracy, precision, and computation time compared to the previous LSTM-based LIDPS model.

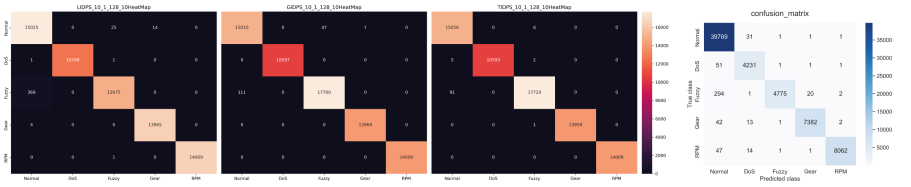


**Fig. 3.** From left to right are the heat comparison charts of LIDPS, GIDPS, and TIDPS in the HCRL dataset, and the far right is the CNN result map

Figure 3 is the heat map generated by several classification results with better results in this experiment and the resulting map of CNN. When the number of experimental test sets corresponding to the three models we proposed is much larger than that of CNN, each model adopted is more accurate than CNN. Although there is still the problem of detection errors in the detection of fuzzy attacks, the detection success rate of gear and RPM spoofing attacks is much better than the CNN method.

### 5.2 Experiments on the ROAD Dataset

This data set is different from the HCRL data set [13,15]. Since there is less attack data and training is faster, 20 epochs are used. See Table 7 for details. Since the amount of data is small, the number of test sets is slightly increased to avoid randomness, and the ratio of training set to test set is 7:3.

**Table 7.** Training parameters based on ROAD dataset

| Parameter | BatchSize | LearningRate | Optimizer | Classes |
|---|---|---|---|---|
| Value | 64 | 0.0005 | Adam | 6 |

Table 8 below shows the model with the best GIDPS and LIDPS results in the experiments and the precision, recall, and F1-score of the 3 layers in the TIDPS model.

**Table 8.** HCRL data step division result

| Model | Attack name | Precision | Recall | F1-score |
|---|---|---|---|---|
| GIDPS | Normal | 0.9953 | 0.9970 | 0.9961 |
| | Correlated signal | 1.0000 | 1.0000 | 1.0000 |
| | Max engine coolant temp | 0.6000 | 0.1875 | 0.2857 |
| | Max speedometer | 1.0000 | 1.0000 | 1.0000 |
| | Reverse light off | 0.9976 | 1.0000 | 0.9988 |
| | Reverse light on | 0.9988 | 0.9996 | 0.9992 |
| LIDPS | Normal | 0.9839 | 0.9919 | 0.9879 |
| | Correlated signal | 0.9917 | 0.9805 | 0.9861 |
| | Max engine coolant temp | 0.0000 | 0.0000 | 0.0000 |
| | Max speedometer | 1.0000 | 1.0000 | 1.0000 |
| | Reverse light off | 0.9976 | 0.9988 | 0.9982 |
| | Reverse light on | 0.9960 | 0.9992 | 0.9976 |
| TIDPS | Normal | 0.9927 | 0.9960 | 0.9943 |
| | Correlated signal | 0.9993 | 1.0000 | 0.9996 |
| | Max engine coolant temp | 0.0000 | 0.0000 | 0.0000 |
| | Max speedometer | 1.0000 | 0.9997 | 0.9998 |
| | Reverse light off | 0.9971 | 0.9971 | 0.9971 |
| | Reverse light on | 0.9975 | 1.0000 | 0.9987 |

As can be seen from Fig. 4 above, due to the small number of samples for the maximum engine coolant temperature attack, the LIDPS and TIDPS models cannot learn their features, so the recognition results of the two models are 0 for this sample. Therefore, its discriminative power is poor and the whole 6-category task degenerates into a 5-category task. Unlike the other two models, GIDPS is able to detect maximum engine coolant temperature attacks and therefore has a higher accuracy than the other models. Overall, GIDPS performed best, with the highest accuracy and the shortest time. LIDPS performed slightly worse in the related category of signaling attacks. In other cases, the three models performed similarly.

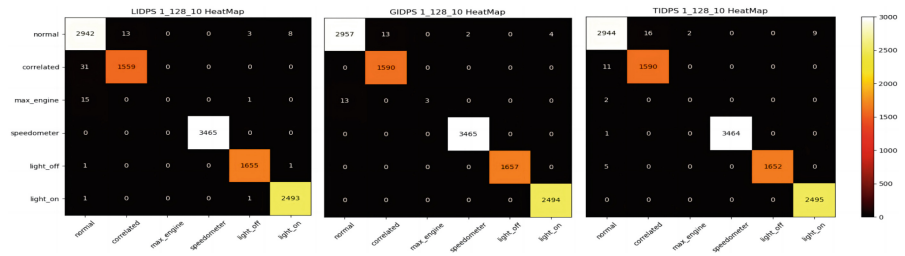


**Fig. 4.** ROAD dataset classification heat map

## 6 Conclusion

GIDPS and TIDPS models are applied for intrusion detection algorithms. We have compared and trained the proposed GIDPS and TIDPS models with previous LSTM-based LIDPS models. The time required by the proposed GIDPS and TIDPS models is much less than the previous LSTM-based LIDPS model, and the accuracy is slightly higher than that of LIDPS. It can be seen that the proposed model is efficient. Furthermore, we also migrated the model to a different dataset and still got this result. We can see that All models still are not perfect in the detection of fuzzy attacks, but under the premise of multi-classification, the effect could be slightly improved compared to binary classification. In the future, we will continue to practice and explore further improvements in algorithm performance, generalization, and resource consumption comparison.

**Acknowledgements.** This work was supported by the Science and Technology Planning Project of Guangdong (2020B0909030005, 2020ZDZX3013, 2023ZZ03), the Science and Technology Planning Project of Guangzhou (202206030007), the Opening Project of Key Laboratory of Safety of Intelligent Robots for State Market Regulation (GQI-KFKT202205), Guangdong Key Laboratory of Data Security and Privacy Preserving (2023B1212060036), Guangdong-Macao Advanced Intelligent Computing Joint Laboratory (2020B1212030003).

## References

1. Koscher, K.: Experimental security analysis of a modern automobile. In: IEEE Symposium on Security and Privacy, vol. **2010**, pp. 447–462. IEEE (2010)
2. Kong, Q., et al.: Blockchain-based privacy-preserving driver monitoring for MaaS in the vehicular IoT. IEEE Trans. Veh. Technol. **70**(4), 3788–3799 (2021)
3. Zhang, T., et al.: VSRQ: quantitative assessment method for safety risk of vehicle intelligent connected system (2023). arXiv: 2305.01898 [cs.AI]
4. Tomlinson, A., Bryans, J., Shaikh, S.A.: Towards viable intrusion detection methods for the automotive controller area network. In: 2nd ACM Computer Science in Cars Symposium, pp. 1–9 (2018)
5. Cheng, X., et al.: Influence-aware successive point-of-interest recommendation. In: World Wide Web, vol. 26, no. 2, pp. 615–629 (2023)
6. Qaddoura, R., et al.: A multi-layer classification approach for intrusion detection in IoT networks based on deep learning. Sensors **21**(9), 2987 (2021)
7. Xie, G., et al.: Threat analysis for automotive CAN networks: a GAN model-based intrusion detection technique. IEEE Trans. Intell. Transp. Syst. **22**(7), 4467–4477 (2021)
8. Guo, J., et al.: TROVE: a context-awareness trust model for VANETs using reinforcement learning. IEEE Internet Things J. **7**(7), 6647–6662 (2020)
9. Javed, A.R., et al.: CANintelliIDS: detecting in-vehicle intrusion attacks on a controller area network using CNN and attention-based GRU. IEEE Trans. Netw. Sci. Eng. **8**(2), 1456–1466 (2021)
10. Müter, M., Groll, A., Freiling, F.C.: A structured approach to anomaly detection for in-vehicle networks. In: 2010 Sixth International Conference on Information Assurance and Security, pp. 92–98. IEEE (2010)

11. Haichun, Z., et al.: Research on entropy-based vehicle CAN bus anomaly detection. Autom. Eng. **43**(10), 1543–1548 (2021)
12. Seo, E., Song, H.M., Kim, H.K.: GIDS: GAN based intrusion detection system for in-vehicle network. In: 2018 16th Annual Conference on Privacy, Security and Trust (PST), pp. 1–6. IEEE (2018)
13. Song, H.M., Woo, J., Kim, H.K.: In-vehicle network intrusion detection using deep convolutional neural network. Veh. Commun. **21**, 100198 (2020)
14. Hu, R., et al.: Multi-attack and multi-classification intrusion detection for vehicle-mounted networks based on mosaic-coded convolutional neural network. Sci. Rep. **12**(1), 1–16 (2022)
15. Seo, E., Song, H.M., Kim, H. K.: GIDS: GAN based intrusion detection system for in-vehicle network. In: 2018 16th Annual Conference on Privacy, Security and Trust (PST), pp. 1–6, August 2018. https://doi.org/10.1109/PST.2018.8514157.
16. Verma, M.E., et al.: Road: the real ornl automotive dynamometer controller area network intrusion detection dataset (with a comprehensive can ids dataset survey & guide). arXiv preprint arXiv:2012.14600 (2020)
17. Chung, J., et al.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
18. Cortes, C., et al.: Advances in neural information processing systems 28. In: Proceedings of the 29th Annual Conference on Neural Information Processing Systems (2015)
19. Taylor, A., Leblanc, S., Japkowicz, N.: Anomaly detection in automobile control network data with long short-term memory networks. In: 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA), pp. 130–139. IEEE (2016)
20. Hossain, M.D., et al.: LSTM-based intrusion detection system for invehicle can bus communications. IEEE Access **8**, 185489–185502 (2020)
21. Hanselmann, M., et al.: CANet: an unsupervised intrusion detection system for high dimensional CAN bus data. IEEE Access **8**, 58194–58205 (2020)
22. Chung, J., et al.: Empirical evaluation of gated recurrent neural networks on sequence modeling. English (US). In: NIPS 2014 Workshop on Deep Learning, December 2014