A Model for Abnormal Detection of In-Vehicle CAN Messages Based on Hyperparameter Optimized CNN

Xiaoyu Zhou

{3159612407@qq.com}

School of Cyberspace Security, Hubei University, No. 368 Youyi Avenue, Wuchang District, Wuhan (Wuchang Main Campus), Wuhan, China

Abstract. Effectively identifying and defending against cyberattacks through intelligent means has become an important research direction for ensuring the safety of intelligent connected vehicles. The paper constructs a novel intrusion detection system framework using CNN, knowledge transfer and model ensemble methods, along with hyperparameter tuning strategies. First, a data transformation model is established to convert CAN message information into images, retaining the key information and features from the original messages while providing good visualization effects and compatibility, thereby facilitating the identification of different network attack patterns. Secondly, a novel intrusion detection system framework is built using CNN, knowledge transfer and model ensemble methods, along with hyperparameter tuning strategies, which can effectively detect various attack features targeting in-vehicle networks. Finally, the effectiveness of the framework is verified using benchmark datasets, and the detection rate data is analyzed alongside other cutting-edge frameworks, showing that this approach delivers outstanding performance and is feasible for practical application.

Keywords: Hyperparameters, Convolutional Neural Networks, CAN, Anomaly Detection.

1 Introduction

As IoT and IoV technologies continue to advance rapidly, modern automotive technology is shifting towards being more intelligent and interconnected. In the automotive industry, smart connected vehicles have emerged as a key area of research in recent years. Models such as the Aito M5 integrate Huawei's HarmonyOS, providing users with seamless device connectivity, allowing control of smart home devices through the vehicle's internal system, and enabling voice control of in-vehicle devices via the voice assistant "Xiaoyi". The XNGP system, developed independently for Xiaopeng's flagship SUV G9, does not rely on high-precision maps and uses a self-evolving AI data system to support autonomous driving functions, having already passed closed testing for autonomous driving in China. Xiaomi's first new energy vehicle, the SU7, launched in 2024, is equipped with the self-developed MIUI Car system, deeply integrated with Xiaomi's smart ecosystem, also supporting linkage with smart home devices, smart voice control, and navigation functions. These examples fully demonstrate the trend of intelligence and connectivity in modern automotive technology. The in-vehicle operating system of intelligent connected vehicles comprises numerous interconnected embedded subsystems. As an in-vehicle information platform, these subsystems achieve efficient communication through the IVN. However, the CAN within the IVN is intended to

CONF-MLA 2024, November 21, Adana, Turkey

DOI 10.4108/eai.21-11-2024.2354634

meet real-time efficient communication needs, leading to strict limitations on packet length and a lack of effective authentication and encryption mechanisms, making it vulnerable to exploitation by attackers. Attackers can implement various attacks by injecting malicious information, including denial-of-service attacks, fuzzing attacks, and deception attacks. Additionally, as in-vehicle networks increasingly integrate with intelligent ecosystems through external interfaces such as cellular networks, Wi-Fi, or Bluetooth, vehicles become more susceptible to novel cyberattacks.

At the 2016 Black Hat conference, Miller and Valasek demonstrated how to attack a vehicle's CAN bus via the onboard diagnostic system interface device, successfully interfering with the driving system of a 2014 Jeep Cherokee [1]. Subsequently, in 2017, Tencent's Keen Security Lab successfully infiltrated Tesla's in-vehicle network, achieving a remote, contactless physical attack [2]. By 2018, a British thief even unlocked and stole a Tesla vehicle in less than two seconds using only a tablet. Therefore, protecting IoV systems and intelligent connected vehicles from cyberattack threats has become a critical challenge, making the development of IDS to defend against potential attacks particularly necessary. As ML and DL technologies continue to progress rapidly, IDS based on these techniques have gained widespread attention in cybersecurity and vehicle systems. Through in-depth analysis of traffic data, machine learning and deep learning technologies can effectively distinguish normal network traffic from malicious attack behaviors and are widely applied in the development of classifier-based intelligent intrusion detection systems. Narayanan et al. [3] proposed an anomaly detection method based on vehicle state, which constructs a Hidden Markov Model (HMM) by collecting vehicle state data and judges whether anomalies exist based on observed vehicle states, such as an unexpected door opening during driving being regarded as an anomaly. This method, however, is limited in practicality as it can only identify anomalies once the attacker has already modified the vehicle's state. Taylor et al. [4] developed an anomaly detection technique leveraging Long Short-Term Memory (LSTM) networks. This method trains a neural network with message data to forecast the content of upcoming messages. If the deviation between the predicted and received message surpasses a defined threshold, the system flags the message as anomalous. Kang et al. [5] introduced a deep neural network-based intrusion detection approach, which extracts feature vectors from in-vehicle network data and utilizes a pre-trained Deep Belief Network (DBN) to optimize the deep neural network parameters . Wang et al. [6] proposed a distributed anomaly detection system based on Hierarchical Temporal Memory (HTM), which effectively monitors in-vehicle network data streams by comparing predicted values with actual ones to detect anomalies. Zou et al. [7] designed an intrusion detection method based on a feed-forward neural network, leveraging a Multi-Layer Perceptron (MLP) model to evaluate the CICIDS2017 dataset, demonstrating its detection capability in vehicular microprocessor environments. Yang et al. [8] developed a Multi-Tiered Hybrid Intrusion Detection System (MTH-IDS), employing a tree-based stacking algorithm for network traffic analysis, which achieved outstanding performance on both IoV and CICIDS2017 datasets. Binsaeed et al. [9] proposed an intrusion detection system (IDS) that combines XGBoost feature selection with deep learning techniques, addressing data imbalance issues through the SMOTE method. The system demonstrated high accuracy and effective detection of minority classes on the CIC-IDS2017 dataset. However, the study was limited to three public datasets and did not validate its generalizability in more complex scenarios. Additionally, the computational complexity of the deep learning model may hinder its suitability for real-time detection. Song et al.[10] proposed an in-vehicle network intrusion detection system (IDS) based on a deep

convolutional neural network (DCNN), which leverages a simplified Inception-ResNet architecture to detect malicious activities within the in-vehicle network. The complexity and real-time requirements of IoV systems present more comprehensive demands on existing intrusion detection methods. Therefore, further optimizing models and integrating system frameworks to enhance detection capabilities has become a new solution.

This paper constructs a novel intrusion detection system framework using convolutional neural networks, transfer learning, and ensemble learning. The data transformation model is established to convert CAN message information into images, retaining the key information and features from the original messages while providing good visualization effects and compatibility, thereby facilitating the identification of different network attack patterns. The novel intrusion detection system framework is built using CNN, transfer learning, ensemble learning, and hyperparameter optimization techniques, which can effectively detect various attack features targeting in-vehicle networks.

2 Background Knowledge

The In-Vehicle Network (IVN), also known as the vehicle network, connects various Electronic Control Units (ECUs) using communication protocols, with the CAN bus being the most widely used standard. Developed by Bosch in 1986, the CAN bus supports real-time communication within vehicles, facilitating control of vehicle body systems efficiently [11].

Convolutional Neural Networks (CNNs) play a fundamental role in deep learning, distinguished by their use of convolutional operations in neural network architectures [12]. First conceptualized as the Time Delay Network by Alexander et al. in 1987, CNNs have since evolved with advancements in deep learning methodologies and computational capabilities, now widely applied across fields such as natural language processing and bioinformatics. A standard CNN architecture includes an input layer, convolutional and pooling layers, and a fully connected layer, with the convolutional and pooling layers forming the central framework. The convolutional layer utilizes fixed-sized kernels that move across image regions, extracting features such as edges and textures, while the pooling layer follows to downsample and reduce feature complexity. This study incorporates convolutional, pooling, fully connected layers, and Dropout into an enhanced CNN framework for feature extraction and classification. Five prominent CNN models were employed for comparison, given their proven effectiveness in image classification tasks. Among them, VGG16 includes five convolutional blocks and three fully connected layers, while VGG19 features three convolutional layers. The Inception network employs convolutional filters to capture multi-scale contextual information, effectively minimizing computational demands through dimensionality reduction. Xception, a refined version of Inception, replaces conventional convolutions with depthwise separable convolutions, while InceptionResNet integrates residual connections from ResNet into the Inception framework, offering enhanced performance despite increased memory and computational requirements. Using transfer learning and fine-tuning techniques, the five CNN models were trained on the vehicle network dataset, and the top three performers were selected as base learners to construct an ensemble model. As shown in Figure 1.

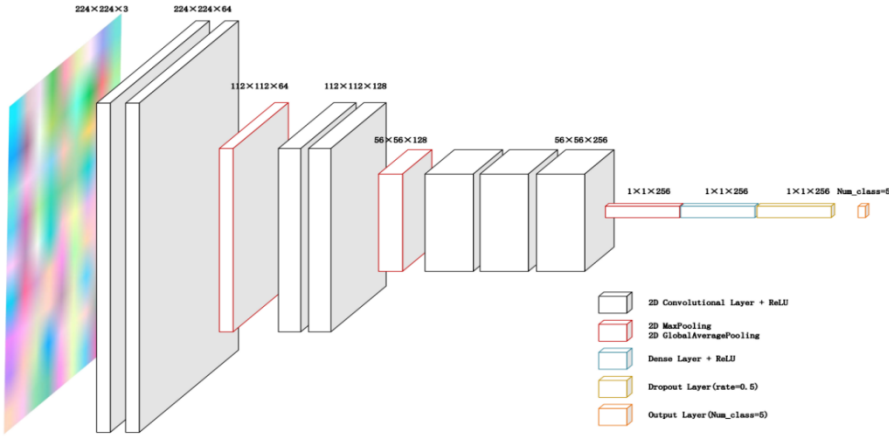

Fig. 1. 2D CNN model

Hyperparameter Optimization is a common method for optimizing machine learning performance. Hyperparameters are parameters set manually before model training, not learned directly from data, and define the model's architecture for controlling the learning process and selecting the optimal set of hyperparameters [13]. Hyperparameters control the learning process. Hyperparameter optimization is typically defined as finding the optimal combination of hyperparameters in a multi-dimensional space to maximize model performance [14]. This problem can be formalized as a black-box optimization problem as follows:

$$\theta^* = \arg\min_{\theta \in \Theta} f(\theta) \tag{1}$$

where θ represents hyperparameters, Θ is the hyperparameter search space, and $f(\theta)$ is the objective function based on the hyperparameter combination θ, usually the loss or error on the validation set. Given the high time cost of model training, hyperparameter optimization algorithms aim to find an optimal hyperparameter combination with minimal evaluations of the objective function. In this study, the VGG16 model was used as an example, and Random Search and Bayesian Optimization methods were employed to adjust hyperparameters such as frozen layers, training epochs, early stopping patience, learning rate, and dropout rate. Random Search samples different hyperparameter combinations within a given space, quickly identifying an optimal combination. Bayesian Optimization, on the other hand, uses a Tree-structured Parzen Estimator (TPE) surrogate model to approximate the relationship between hyperparameters and the objective function. By defining an objective function that receives hyperparameters and returns model accuracy as an evaluation metric and specifying a hyperparameter space, Bayesian Optimization updates the TPE surrogate model based on results in each iteration and selects the next optimal hyperparameters for evaluation. Finally, the model is retrained with optimized hyperparameters to validate performance. Compared to Random Search, Bayesian Optimization quickly converges to the global optimum with fewer evaluations.

3 Methodology

In this context, effectively utilizing intelligent methods to identify and defend against network attacks has become a critical research direction for ensuring the security of intelligent connected vehicles. Based on this, this study uses CNN, Domain Adaptation, and Combined Learning to construct a novel intrusion detection system framework. This framework combines various advanced learning techniques, including hyperparameter optimization, to enhance model performance and improve the detection capability for attack patterns in vehicle network traffic. This study not only employs a custom-designed CNN model to strengthen learning effectiveness but also integrates prediction results from multiple models, further improving detection accuracy and efficiency. The framework's validity is verified using the CICIDS2017 and Car-Hacking benchmark datasets, achieving a detection rate and F1 score of 99.25%. The first step involves creating a data transformation approach that converts CAN messages into images, preserving key information and features from the original messages. This approach provides better visualization and compatibility, facilitating the recognition of various network attack patterns. Next, CNN, Domain Adaptation, Combined Learning, and Optimization of Hyperparameters are used to construct a novel intrusion detection system framework capable of effectively detecting a variety of attack characteristics targeting in-vehicle networks. Finally, the framework is validated using benchmark datasets, and the detection rate data are Compared to other advanced frameworks. The data preprocessing for the algorithm is as follows:

Data: CAN bus intrusion detection dataset
Result: Training and test sets of 9x9x3 color images

- Step 1. Read the raw data from the CSV file;
- Step 2. Quantify features using Quantile Transformer;
- Step 3. Multiply the quantified features by 255 to convert them into pixel values;
- Step 4. For each data category, generate images as follows:
- for each data category & each data sample do Convert the features of the data sample into a 9x9x3 matrix representing a color image; Name the image using the label of the data sample; Save the generated image to a specified folder;
- Step 5. Split all generated images into 80% for the training set and 20% for the test set;
- Step 6. Adjust the resolution of the images to 224x224 pixels for subsequent model processing;
- End

The model training process for the algorithm in this paper is as follows:
Data: Training set Dtrain, test set Dtest
Result: Optimized CNN model

- Step 1. Initialize dataset Dtrain, test set Dtest;
- Step 2. Data preprocessing: normalization and noise removal;
- Step 3. Select a pre-trained model Mpretrained (such as ResNet or VGG);
- Step 4. Replace the last layer of Mpretrained with a classification layer Mnew;
- Step 5. Freeze the parameters of the convolutional layers in Mpretrained;
- Step 6. Define optimizer O (e.g., Adam, SGD);
- Step 7. Define loss function L (e.g., cross-entropy loss);

- Step 8. for each x ∈ Dtrain do Apply data augmentation: random rotation, scaling, and horizontal flipping; Input x into the model Mnew; Calculate the loss Ltrain = L(model output, true label); Backpropagate to compute gradients; Update model parameters Mnew via optimizer O;
- Step 9. Unfreeze part of the convolutional layers in Mpretrained for fine-tuning;
- Step 10. for x ∈ Dtrain do Calculate the loss Lfine-tune = L(model output, true label); Backpropagate to compute new gradients; Update model parameters Mnew via optimizer O;
- Step 11. Evaluate the model on Dtest;
- Step 12. Calculate and output evaluation metrics: accuracy, recall, F1-score;
- Step 13. Adjust model parameters or hyperparameters based on the evaluation results; if evaluation results are unsatisfactory, then Retrain or adjust the network structure;
- End

4 Experiment

The hardware environment for this algorithm includes a GPU RTX 3090 (24GB) * 1, CPU 12 vCPU Intel(R) Xeon(R) Platinum 8375C CPU @ 2.90GHz, with a system disk of 30 GB, data disk of 50 GB, and 72 GB of memory. The software environment consists of TensorFlow 2.9.0 and Python 3.8 (Ubuntu 20.04).

The study uses two datasets: The Car-Hacking dataset [15], generated from the CAN bus of real vehicles, includes four main attack types: DDoS, input fuzzing, gear tampering, and RPM tampering attacks. Meanwhile, the CICIDS2017 dataset [16], a widely-used benchmark for testing Intrusion Detection System performance, covers a variety of advanced attack types, such as port scanning, brute force, web attacks, and botnet activities. Data preprocessing was conducted to ensure compatibility with the CNN model, transforming tabular in-vehicle network traffic data into image format for better CNN performance. The first step of the image transformation process was normalizing the data within the 0–255 range. To address outliers effectively, quantile normalization was applied to ensure that feature values followed a normal distribution, enhancing model stability against outliers. Data samples were then divided into blocks based on timestamps and feature size; each CAN message from the two datasets contains 9 and 20 features, respectively. These were transformed into color images of 9×9×3 or 20×20×3 dimensions [17]. Finally, based on the attack pattern within each data block, the transformed images were labeled. If all samples in a block were normal, the image was labeled “Normal”; if it contained attack samples, it was labeled with the most frequent attack type within the block, such as “RPM.” The processed image set ultimately served as a rich input source for the CNN model, significantly enhancing its accuracy and reliability in attack detection. Samples from each category are shown in Figure 2.

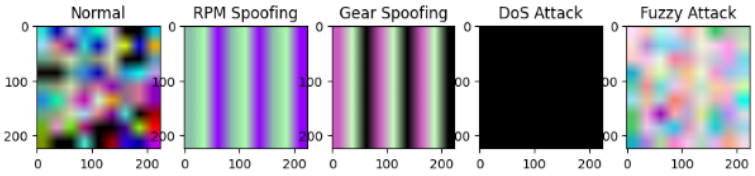

Fig. 2. shows samples from each category.

In training the deep learning models, VGG16, VGG19, Xception, Inception, and Inception ResNet were used as base CNN models [18], alongside a custom CNN model enhanced by hyperparameter optimization (HPO). The final image set was directly fed into these models. Given the generalizability of the low-level feature patterns learned by CNNs, transfer learning was employed by transferring the lower-layer weights of a pre-trained neural network model from one dataset to another [19], while retraining only the upper layers on the new dataset to enhance model performance by updating high-level features. Similar to other deep learning models, the CNN model requires tuning multiple hyperparameters, categorized as either model design or model training hyperparameters. In the proposed framework, model design hyperparameters include the number or percentage of frozen layers, learning rate, and dropout rate, while model training hyperparameters include batch size, epoch number, and early stopping patience. Five-fold cross-validation was applied to the experiments to assess the proposed model, preventing overfitting and biased results. Given that network traffic data is typically highly imbalanced, with attack samples in the minority, four metrics-accuracy, precision, recall, and F1 score-were used for performance evaluation [20]. The loss function (used to monitor each model's training progress) evaluates the gap between the predicted values and true labels; the closer the training loss curve is to the validation loss curve, the better the model performance. Ultimately, the top three models, including the optimized custom CNN model, were selected for the ensemble learning model. The illustrates the accuracy and loss of CNN models are shown in Figure 3.

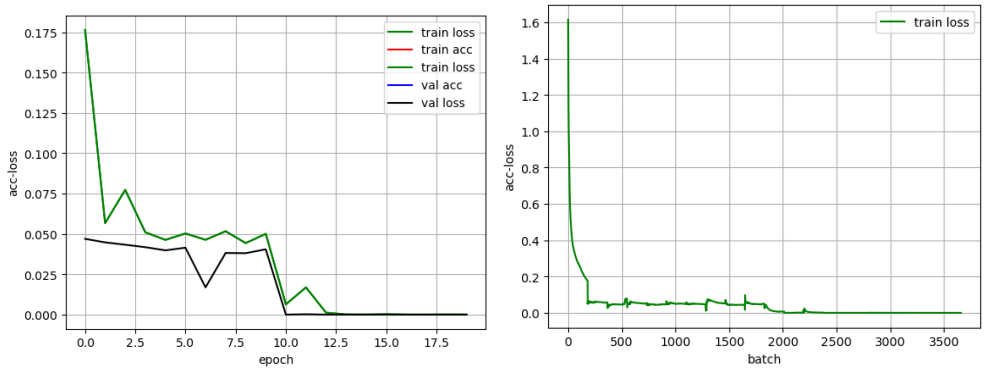

Fig. 3. illustrates the accuracy and loss of CNN models.

Table 1 presents the detailed data for first 10 epoch, while Table 2 provides the specific data for the last 10 epoch. Both tables offer a clear snapshot of the model's performance trends across

different epochs. The training and validation loss curves demonstrate smooth convergence, with the losses significantly decreasing as epochs progress. This suggests the model effectively minimizes error and aligns predictions with true labels.

Table 1. the detailed data for first 10 epoch

Epoch	Train Accuracy	Train Loss	Valid Accuracy	Valid Loss
1	0.9511	0.1086	0.9783	0.0436
2	0.9707	0.0665	0.9756	0.0473
3	0.9681	0.0524	0.9573	0.0502
4	0.9701	0.0518	0.9706	0.0493
5	0.9693	0.0506	0.9729	0.0479
6	0.9686	0.0580	0.9706	0.0487
7	0.9704	0.0552	0.9705	0.0501
8	0.9693	0.0610	0.9693	0.0532
9	0.9653	0.2308	0.9693	0.0523

Table 2. the detailed data for last 10 epoch

Epoch	Train Accuracy	Train Loss	Valid Accuracy	Valid Loss
1	0.9511	0.1086	0.9783	0.0436
2	0.9707	0.0665	0.9756	0.0473
3	0.9681	0.0524	0.9573	0.0502
4	0.9701	0.0518	0.9706	0.0493
5	0.9693	0.0506	0.9729	0.0479
6	0.9686	0.0580	0.9706	0.0487
7	0.9704	0.0552	0.9705	0.0501
8	0.9693	0.0610	0.9693	0.0532
9	0.9653	0.2308	0.9693	0.0523
10	0.9675	0.1535	0.9560	0.0560

5 Conclusion

The study constructed a novel intrusion detection system framework utilizing Convolutional Neural Networks (CNN), Domain Adaptation, and Combined Learning techniques to address the growing security challenges in modern vehicular networks. Through comprehensive experimentation and analysis, the framework effectively detects various in-vehicle network attack patterns with high accuracy and reliability. The integration of CNN architecture enables deep feature extraction from network traffic data, while Domain Adaptation mechanisms ensure the model's robustness across different operational environments and vehicle types. The Combined Learning approach synthesizes multiple learning paradigms to enhance the system's overall performance and adaptability. Benchmark datasets, including both normal vehicle

communication patterns and simulated attack scenarios, were used to validate the framework's efficacy. The experimental results demonstrated superior detection rates compared to traditional intrusion detection methods, with significantly reduced false positive rates. The framework showed particular strength in identifying sophisticated attack patterns that conventional systems often miss. Future work will focus on leveraging intelligent methods to identify and defend against network attacks, an essential research direction for ensuring the security of intelligent connected vehicles. This includes exploring advanced deep learning architectures, developing real-time response mechanisms, and incorporating emerging cybersecurity techniques. Additionally, the research team plans to investigate adaptive defense strategies that can evolve with new threat patterns and investigate the framework's scalability across different vehicle manufacturers and models. The integration of federated learning approaches could further enhance privacy preservation while maintaining robust security measures. This ongoing research represents a critical step toward establishing comprehensive security solutions for the rapidly evolving automotive cyber-physical systems.

References

[1] Putnam III W H. Developing a CAN Bus-based Secure System for Automotive Module Connectivity[D]. University of Aizu, 2018.

[2] Nie S, Liu L, Du Y. Free-fall: Hacking tesla from wireless to can bus[J]. Briefing, Black Hat USA, 2017, 25(1): 16.

[3] Narayanan S N, Mittal S, Joshi A. OBD_SecureAlert: An anomaly detection system for vehicles[C]//2016 IEEE International Conference on Smart Computing (SMARTCOMP). IEEE, 2016: 1-6.

[4] Taylor A. Anomaly-based detection of malicious activity in in-vehicle networks[D]. Université d'Ottawa/University of Ottawa, 2017.

[5] Kang, M. J., & Kang, J. W. (2016). Intrusion detection system using deep neural network for in-vehicle network security. PLOS ONE, 11(6), e0155781.

[6] Wang C, Zhao Z, Gong L, et al. A distributed anomaly detection system for in-vehicle network using HTM[J]. IEEE Access, 2018, 6: 9091-9098.

[7] Zou L, Luo X, Zhang Y, et al. HC-DTTSVM: A network intrusion detection method based on decision tree twin support vector machine and hierarchical clustering[J]. IEEE Access, 2023, 11: 21404-21416.

[8] Yang L, Shami A. IDS-ML: An open source code for Intrusion Detection System development using Machine Learning[J]. Software Impacts, 2022, 14: 100446.

[9] Binsaeed K A, Hafez A M. Enhancing intrusion detection systems with XGBoost feature selection and deep learning approaches[J]. International Journal of Advanced Computer Science and Applications, 2023, 14(5).

[10] Song H M, Woo J, Kim H K. In-vehicle network intrusion detection using deep convolutional neural network[J]. Vehicular Communications, 2020, 21: 100198.

[11] Lokman, S. F., Othman, A. T., & Abu-Bakar, M. H. (2019). Intrusion detection system for automotive Controller Area Network (CAN) bus system: A review. EURASIP Journal on Wireless Communications and Networking, 2019(1), 1 – 17.

[12] Shahriar M H, Xiao Y, Moriano P, et al. CANShield: Deep Learning-Based Intrusion Detection Framework for Controller Area Networks at the Signal-Level[J]. IEEE Internet of Things Journal, 2023.
[13] Bischl B, Binder M, Lang M, et al. Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges[J]. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2023, 13(2): e1484.
[14] Monica, Agrawal P. A Survey on Hyperparameter Optimization of Machine Learning Models[C]//Proceedings of the 2024 2nd International Conference on Disruptive Technologies (ICDT). Greater Noida, India: IEEE, 2024: 11-15. DOI: 10.1109/ICDT61202.2024.10489732.
[15] Rajapaksha S, Kalutarage H, Al-Kadri M O, et al. Ai-based intrusion detection systems for in-vehicle networks: A survey[J]. ACM Computing Surveys, 2023, 55(11): 1-40.
[16] Stiawan D, Idris M Y B, Bamhdi A M, et al. CICIDS-2017 dataset feature analysis with information gain for anomaly detection[J]. IEEE Access, 2020, 8: 132911-132921.
[17] Zhu Y, Brettin T, Xia F, et al. Converting tabular data into images for deep learning with convolutional neural networks[J]. Scientific reports, 2021, 11(1): 11325.
[18] Zhao Z, Zhang Q, Yu X, et al. Applications of unsupervised deep transfer learning to intelligent fault diagnosis: A survey and comparative study[J]. IEEE Transactions on Instrumentation and Measurement, 2021, 70: 1-28.
[19] Sun R Y. Optimization for deep learning: An overview[J]. Journal of the Operations Research Society of China, 2020, 8(2): 249-294.
[20] Zhou J, Gandomi A H, Chen F, et al. Evaluating the quality of machine learning explanations: A survey on methods and metrics[J]. Electronics, 2021, 10(5): 593.