Hybrid Transfer and Self-Supervised Learning Approaches in Neural Networks for Intelligent Vehicle Intrusion Detection and Analysis

Tian Zhang, *Student Member, IEEE*, Cuifeng Du, Yuyu Zhou, Quanlong Guan, *Senior Member, IEEE*, Zhiquan Liu, *Member, IEEE*, Xiujie Huang, *Member, IEEE*, Zhiguo Gong, *Senior Member, IEEE*, Lianbing Deng, *Member, IEEE*, and Yang Li

***Abstract*—Intrusion detection is crucial for safeguarding intelligent vehicle systems, aiming to identify abnormal network traffic and operational anomalies. Traditional methods primarily focus on spatial features of attacks, often neglecting temporal dynamics essential for detecting complex, evolving threats. Additionally, the effectiveness of existing techniques is limited by the scope and quality of available datasets, reducing their ability to detect novel, unseen attacks. To address these challenges, this article introduces a Transformer-based transfer learning intrusion detection system (TIDS), designed to capture and analyze spatiotemporal sequence features from vehicle data. TIDS generates high-dimensional feature representations of intricate intrusion patterns, improving the detection of known attack types through instance-based transfer learning, enhancing domain adaptability. Moreover, we proposed a novel self-supervised box classification method that enhances the system's capability to detect previously unknown attacks, thereby increasing the overall robustness of the intrusion detection process. Comparative experiments demonstrate that TIDS outperforms traditional methods in detection speed and accuracy across various intrusion scenarios, effectively responding to emerging threats in intelligent vehicle networks.**

***Index Terms*—Intelligent vehicle intrusion detection, self-supervised box classification, spatiotemporal sequence features, transformer, transfer learning.**

Received 16 November 2024; accepted 11 December 2024. Date of publication 16 December 2024; date of current version 26 March 2025. This work was supported in part by the Science and Technology Planning Project of Guangzhou under Grant 202206030007 and Nansha District under Grant 2023ZD001; in part by the Science and Technology Planning Project of Guangdong under Grant 2023A0505030013 and Grant 2023ZZ03; in part by the Guangdong Key Laboratory of Data Security and Privacy Preserving under Grant 2023B1212060036; in part by the Guangdong–Macau Advanced Intelligent Computing Joint Laboratory under Grant 2020B1212030003; and in part by the Jiangxi Institute of Civil Military Integration under Grant 2024JXRH0Y10 and Grant 2024JMRH0Y09. *(Corresponding authors: Quanlong Guan; Xiujie Huang.)*

Tian Zhang is with the Department of Cyberspace Security, College of Information Science and Technology, Jinan University, Guangzhou 511486, China (e-mail: tianzhang@stu2022.jnu.edu.cn).

Cuifeng Du is with the Operator Business Department, Cetc Potevio Science and Technology Company Ltd., Guangzhou 510310, China (e-mail: ducuifeng@gcidesign.com).

Yuyu Zhou is with the College of Intelligent Systems Science and Engineering, Jinan University, Zhuhai 519070, China (e-mail: zyy@jnu.edu.cn).

Quanlong Guan and Xiujie Huang are with the Department of Computer Science, the College of Information Science and Technology, and the Guangdong Institute of Smart Education, Jinan University, Guangzhou 510632, China (e-mail: gql@jnu.edu.cn; t_xiujie@jnu.edu.cn).

Zhiquan Liu is with the College of Cyber Security, Jinan University, Guangzhou 510632, China (e-mail: zqliu@jnu.edu.cn).

Zhiguo Gong is with the State Key Laboratory of Internet of Things for Smart City, Department of Computer Information Science, University of Macau, Macau, China (e-mail: fstzgg@umac.mo).

Lianbing Deng is with Guangdong Qinzhi Science and Technology Research Institute, Zhuhai 519031, China (e-mail: dlbpaper@outlook.com).

Yang Li is with the Smart Transportation R&D Department, Guangzhou Fundway Smart Transportation Research and Development Company Ltd., Guangzhou 510653, China, and also with the Science and Technology Innovation Business Department, PCI Technology and Service Company Ltd., Guangzhou 510653, China (e-mail: liyang12@pcitech.com).

Digital Object Identifier 10.1109/JIOT.2024.3518636

I. Introduction

In RECENT years, the rapid advancement of communication and network technologies has transformed the automotive industry, driving it toward electrification, connectivity, and intelligence [1], [2]. Intelligent vehicles, equipped with a range of sensors and network systems, not only need to ensure real-time safety but also address critical network security challenges [3]. As these vehicles become increasingly interconnected, robust, real-time intrusion detection systems (IDSs) are essential for securing in-vehicle communication [4].

In-vehicle networks, such as controller area network (CAN) buses, are vital in connecting and controlling electronic control units (ECUs) and facilitating communication with external systems [5], [6]. While the CAN bus offers advantages in terms of real-time response and cost efficiency [7], [8], it lacks fundamental security mechanisms, such as encryption, making it vulnerable to malicious attacks that can compromise vehicle safety and functionality [9], [10].

Thus, enhancing the detection of network anomalies and intrusions is critical for intelligent vehicle security [11], [12], [13], [14]. Traditional IDS often focus on spatial features, neglecting temporal dynamics like attack frequency, duration, and sequence, which are crucial for detecting sophisticated or prolonged attacks [15], [16]. The absence of temporal analysis leads to missed detections or delayed responses, as many threats evolve over time and cannot be captured solely by static analysis [17], [18].

Moreover, existing IDS datasets often cover a narrow range of attacks, limiting the model's generalization to new environments or novel attack patterns [11], [17], [19], [20], [21]. As new attack vectors emerge, models trained on outdated or limited datasets may fail to detect these evolving threats [22], [23]. Generating comprehensive labeled datasets

is also costly and time consuming, further exacerbating the problem [24], [25]. Thus, there is a pressing need for models capable of adapting to new datasets and environments while maintaining high detection accuracy.

In response to these challenges, we proposed a Transformer-based TIDS that integrates both spatial and temporal features to address the shortcomings of existing methods. Unlike traditional approaches that focus mainly on spatial features, TIDS uses Transformer architectures to capture complex spatiotemporal patterns, providing a more comprehensive analysis of attack behaviors. Transformers are particularly effective at modeling sequential data, which is essential for understanding the temporal progression of attacks. By capturing high-dimensional spatiotemporal features, TIDS improves detection accuracy for both rapid and slow-developing attacks.

TIDS also employs transfer learning [24], [26], allowing it to generalize across different datasets and adapt to new environments. This is achieved by mapping the feature space of one dataset to another, aligning the data distributions of both source and target domains, thereby enhancing the model's ability to detect known attacks in new contexts [26]. Additionally, TIDS incorporates a self-supervised learning component, which enables the model to learn from unlabeled data, enhancing its ability to detect new instances of known attacks. This is particularly beneficial in intelligent vehicle networks, where labeled data is scarce, and the attack landscape is constantly evolving. The primary contributions of this article are as follows.

1) We proposed a Transformer-based TIDS that integrates both spatial and temporal features of vehicle network data. Unlike traditional systems that focus primarily on spatial features, TIDS captures spatiotemporal dependencies, offering a more comprehensive detection framework. This results in improved detection accuracy and timeliness, especially for complex attack patterns that evolve over time.
2) We introduced an instance-based transfer learning method to enhance the domain adaptability of the TIDS model. This approach enables the system to detect known attack types across diverse datasets, improving generalization and ensuring high performance even in previously unseen environments. The key difference is the model's ability to adapt to new contexts without requiring extensive retraining, enhancing its robustness.
3) We integrated a self-supervised learning mechanism within TIDS to reduce reliance on labeled data, allowing the model to learn from unlabeled data and improve detection of novel attack manifestations. This approach strengthens the system's ability to identify emerging threats, making it more robust against evolving security risks without the need for extensive labeled datasets.

This article addresses several critical challenges in intelligent vehicle intrusion detection by proposing a system that integrates spatiotemporal analysis, leverages transfer learning for better generalization, and incorporates self-supervised learning for enhanced adaptability to new attack scenarios. The remainder of this article is organized as follows. Section II reviews the prior research on vehicle intrusion detection, highlighting current challenges and providing context for our contributions. Section III introduces the essential concepts about automotive CAN networks, including message formats, attack types, and relevant datasets. Section IV describes the classification of CAN messages using neural networks and the self-supervised box classification method. Section V presents the experimental results, including those from the spatiotemporal attributes-based model and the self-supervised box classification method. Finally, Section VI summarizes the findings and suggests future research directions.

II. Related Work

Recent advancements in intelligent vehicle security have drawn considerable attention to intrusion detection in the CAN bus, a crucial component of intelligent vehicle systems [27]. Existing intrusion detection methods can be broadly categorized into statistical and machine learning-based approaches [28].

Statistical Methods: Statistical techniques, such as entropy-based detection, have been widely employed to identify attacks like flooding and replay [27]. However, these methods often suffer from high false positive rates, particularly when dealing with tampering and spoofing attacks [9]. To address these issues, He et al. [29] proposed an approach integrating message relative distance, which improves detection for certain attacks but remains limited when confronted with complex or unknown attack patterns.

Machine Learning-Based Methods: Machine learning models, including supervised and unsupervised learning, offer greater flexibility by identifying abnormal patterns based on labeled data [30], [31]. Despite their potential, these models face challenges related to feature design and high false positive rates. Recent advances have focused on deep learning methods like long short-term memory (LSTM) networks, which automatically extract features to identify anomalies [15], [32]. For example, Seo et al. proposed a GAN-based intrusion detection system [33], while Song et al. [16] utilized deep convolutional neural networks (DCNNs) to optimize network complexity and enhance detection capabilities. Nevertheless, many of these methods either overlook temporal dependencies or struggle to distinguish between genuine attacks and regular network anomalies.

Temporal Feature Integration and Transfer Learning: Recent research has highlighted the importance of temporal dynamics in intrusion detection. LSTM-based methods, like those proposed by Pawelec et al. [34], focus on the sequence prediction of CAN messages at the bit level. Despite notable improvements, these methods often rely on manually set thresholds, which can lead to inaccuracies in detection. To overcome this, Javed et al.'s CANintelliIDS model integrates CNN and attention mechanisms to effectively detect mixed attacks [35]. However, practical deployment of such models is hindered by insufficient information on training time and detection latency.

In light of these challenges, transfer learning has emerged as a promising approach to improve model adaptability and generalization. By leveraging pretrained models from related

tasks, transfer learning methods can enhance detection accuracy, particularly when labeled data is scarce. Our proposed Transformer-based TIDS builds upon this concept by integrating spatiotemporal features, thus improving the detection of both known and novel attacks.

Incorporating Multifactor Authentication for Enhanced Security: Intelligent vehicle systems require robust authentication alongside intrusion detection. Quantum2FA [36] proposes quantum-resistant two-factor authentication using lattice-based cryptography, which enhances security but may introduce computational delays in resource-constrained environments. Additionally, lattice-based cryptography faces challenges in performance optimization and standardization. AB-PAKE [37] offers an attribute-based password-authenticated key exchange protocol for fine-grained access control in vehicle networks, though it encounters issues in key management, attribute revocation, and performance degradation under high authentication volumes due to cryptographic demands. Javaheri et al. [38] highlighted the increasing use of fuzzy logic-based anomaly detection for DDoS and traffic anomaly detection, noting that while these adaptive algorithms are flexible, they struggle with real-time accuracy and require significant computational resources, limiting their use in high-speed vehicle systems.

Proposed Approach: Despite significant advancements, existing models still face limitations in handling false positives, temporal dynamics, and novel attack types. To address these gaps, we proposed a Transformer-based TIDS that integrates spatiotemporal features, improving the detection of both known and unknown attack variants. Additionally, we introduced a self-supervised box classification method, enhancing the model's ability to identify previously unseen attacks without extensive reliance on labeled data. While security mechanisms, such as multifactor authentication are important, the focus of this work is primarily on the detection aspect. Given the resource constraints in intelligent vehicle systems, we assumed that standard security mechanisms are in place, providing a baseline for secure operations. Experimental results on real-world intelligent vehicle datasets validate the effectiveness of our approach, demonstrating enhanced generalization and robustness in detecting in-vehicle network intrusions.

III. Dataset Description and Preprocessing

This section introduces the datasets used in this study: the Car Hacking dataset [16], [33], collected by the Korea Hacker and Countermeasures Research Laboratory, and the ROAD dataset [39], recorded by Oak Ridge National Laboratory. These datasets represent real-world intelligent vehicle network data and are used to test various attack detection methods. Below, we provided an overview of each dataset and describe the preprocessing steps applied, including considerations for data privacy and security.

A. Car Hacking Dataset Overview

The Car Hacking dataset [16], [33] was collected in a real in-vehicle network environment and includes both normal communication and various attack types, such as Denial of Service (DoS), fuzzing, spoofing, and Tachometer spoofing attacks. Each attack lasts between 3 to 5 s, with 300 such intrusions in each attack dataset. A typical attack dataset includes 30 to 40 min of CAN bus communication.

The dataset includes the following fields:

1) Timestamp: the timestamp of each record;
2) CAN ID: the CAN message identifier in hexadecimal;
3) DLC: the data length code (0 to 8 bytes);
4) DATA[0–7]: The data values; and
5) Flag: an indicator of whether the message is normal (R) or an attack (T). Data from this dataset is intercepted and processed to analyze attack patterns, where each frame with the attack label "T" is identified as part of the attack.

1) Data Preprocessing of the Car Hacking Dataset: The dataset contains approximately four million records. Due to memory constraints, a random sample of one million records was selected for this study. The following preprocessing steps were applied, with particular focus on ensuring the privacy and security of sensitive data.

1) *Handling Missing Values:* NaN values, which are caused by missing data in certain messages, are filled with zeros. This ensures the continuity of data without introducing any erroneous or personally identifiable information. By substituting missing values with neutral elements (i.e., zeros), we avoided the risk of reconstructing sensitive vehicle data from incomplete messages.
2) *Data Transformation and Anonymization:* CAN message data, originally in hexadecimal format, is converted into decimal for computational ease. In the conversion process, sensitive vehicle data, such as the vehicle's unique identifiers embedded within the CAN ID or DATA fields, are transformed to generic representations. This ensures that no vehicle-specific information, can be traced back from the transformed dataset.
3) *Frame Interval Calculation:* The time difference between consecutive messages is computed and added as an additional feature. While this provides crucial sequential information for attack detection, it does not compromise privacy, as it does not involve any direct vehicle-identifiable features. The sequence of data points is abstracted, ensuring that no private or identifiable patterns can be inferred.
4) *Normalization:* Data normalization ensures that all features are considered equally during model training. Particularly for time intervals, which are typically small and may vary across datasets, normalization helps prevent the system from associating specific temporal patterns with individual vehicles. This removes any potential for private temporal data to be reverse-engineered.
5) *Segmentation:* The dataset is divided into time slices with a predefined step size to capture sequential dependencies. This segmentation process is designed to further anonymize the data by aggregating and abstracting vehicle communications into generalized time windows, rather than retaining raw, potentially identifiable messages.

TABLE I
CAR HACKING DATASET 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

TABLE II
MASQUERADING ATTACK IN ROAD DATASET

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

6) *Labeling:* The dataset includes a tag field to identify attack packets. The time slice containing the attack label is marked with the corresponding attack sequence number. Importantly, the labeling process is entirely based on attack patterns rather than any vehicle-specific identifiers. After preprocessing, the dataset is balanced by selecting a proportionate amount of normal data, ensuring that the private details of vehicle operations are not exposed through imbalance or overrepresentation.

After preprocessing, the dataset is balanced by selecting a proportionate amount of normal data, as shown in Table I.

B. ROAD Dataset Overview

The ROAD dataset [39] contains real-world attacks on intelligent vehicles, including fuzzing attacks, data manipulation attacks (e.g., correlated signal, engine coolant temperature, and speedometer), and advanced attacks (e.g., accelerator drive and reverse attacks). Each attack was physically verified, and the effects on the vehicle were recorded.

The dataset is available in two formats: 1) the conventional message format, which consists of textual CAN frames and 2) the signal format, where the 8-byte DATA field is extracted and analyzed as individual signals, with 22 distinct signals in total. The Label field indicates whether the packet is an attack packet. Some attacks, such as fuzzy and advanced attacks, are not labeled, making them difficult for supervised learning.

The dataset also simulates masquerading attacks by removing the legitimate target ID frame before each injected frame. This ensures that the injected message appears as if it were a normal message, complicating detection and requiring advanced techniques to identify anomalies. An example of a masquerading attack is shown in Table II, where the modified message (with the underlined entry) replaces the original frame in the dataset.

1) Data Preprocessing of ROAD Dataset: The ROAD dataset's message data is composed of textual strings, which complicates processing. Additionally, attack labels are not directly provided, requiring the comparison of multiple files for categorization. The preprocessing steps for this dataset mirror those of the Car Hacking dataset. These include the conversion of textual data into numeric format, normalization of the features, and segmentation into time slices for model training. Table III presents the step division results for different time intervals in the ROAD dataset [39].

TABLE III
ROAD DATASET 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

After completing the data preprocessing steps, it is clear that attacks are sparsely distributed within the dataset. However, the preprocessing procedures ensure proper segmentation and labeling, making the dataset suitable for training and evaluating intrusion detection models.

IV. ESTABLISHMENT OF INTRUSION DETECTION SYSTEM MODEL

This article introduces a deep neural network model for detecting abnormal behaviors in intelligent vehicle CAN messages. Section III covers the experimental dataset and preprocessing steps. This section focuses on the design of the intrusion detection model (IDS) for intelligent vehicles.

We proposed a Transformer-based transfer learning model (TIDS) that extracts time-series features from vehicle data, transforming them into high-dimensional feature maps to capture spatiotemporal information. This model preserves the spatiotemporal attributes of intrusion attacks and leverages instance-based transfer learning, distinguishing it from LSTM [15] and GRU-based models [40], which are typically used for binary classification. To demonstrate the model's effectiveness, we extended TIDS to multiclass classification tasks and introduce the multiclass IDS models based on LSTM (LMIDS) and GRU (GMIDS) to improve detection accuracy in complex attack scenarios. The architecture of the proposed IDS model is shown in Fig. 1. The pseudocode for the intrusion detection process, including data preprocessing, feature extraction, attack classification, anomaly detection, and final decision making for the LMIDS, GMIDS, and TIDS models, is described in Algorithm 1.

A. LMIDS Model Construction

LSTM networks [15], [41], [42] are particularly well-suited for modeling sequential data, such as vehicle CAN messages. The LMIDS model uses the gating mechanism of LSTM to learn temporal dependencies in the data, enabling effective classification of normal and attack behaviors.

The LSTM network is composed of three primary gates: 1) the forget gate; 2) the input gate; and 3) the output gate, which control the flow of information through the network. The state updates in the LSTM model are described by the following equations:

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

Equation (1) computes the forget gate f_t, which decides what proportion of the previous hidden state h_{t-1} and the current

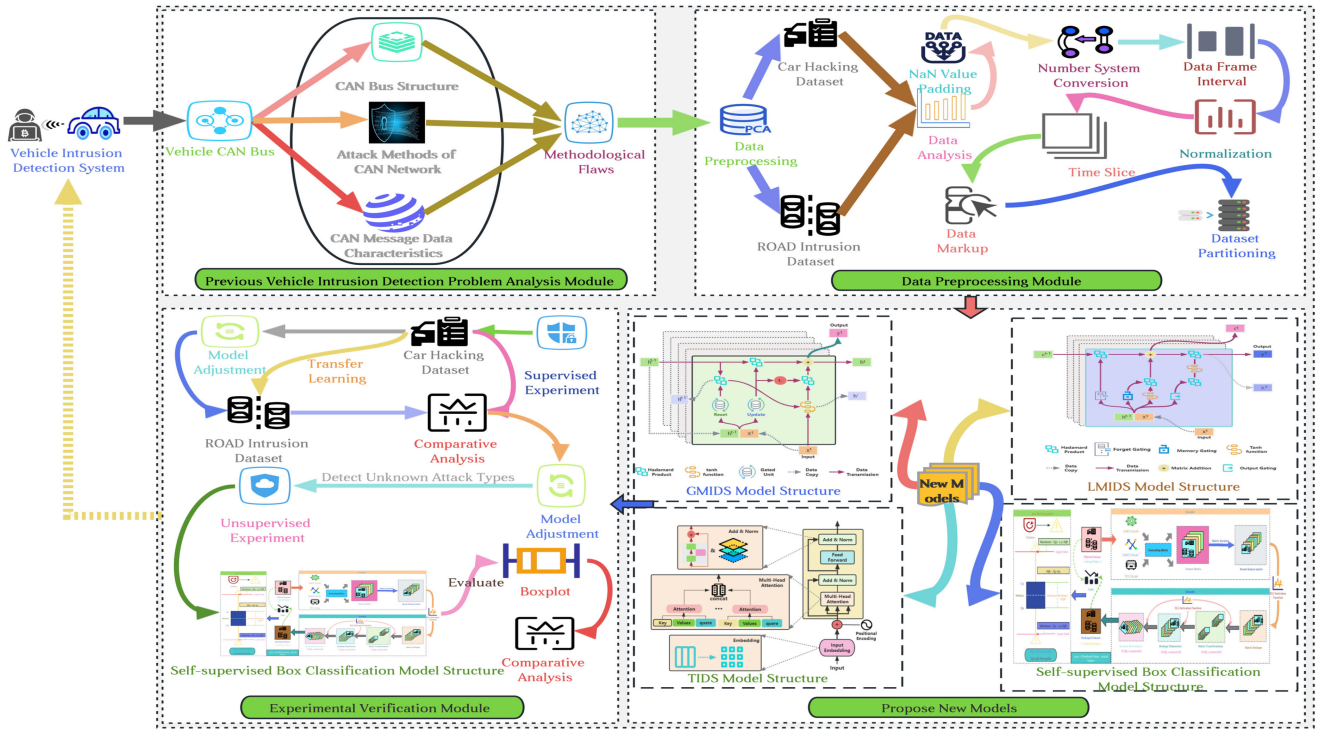


Fig. 1. Overarching structure of the intrusion detection system model.

input x_t to forget. The sigmoid function δ maps this value to the range [0, 1]

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

Equation (2) updates the cell state C_t. The input gate i_t controls how much of the new information $C_t^{\sim}$ should be added to the memory. The forget gate f_t scales the previous memory C_{t-1}, and the final memory C_t is updated accordingly

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

In (3), the output gate o_t controls the amount of memory C_t that is output to the hidden state h_t. This gate helps determine what information should be passed on to the next layer.

The LMIDS model detects attacks by learning patterns of normal behavior in the temporal sequence of CAN messages. Anomalous behavior is flagged when the model identifies deviations from learned normal patterns.

B. GMIDS Model Construction

The gated recurrent unit (GRU) [40] is a simplified version of LSTM, with only two gating units: 1) the reset gate and 2) the update gate. GMIDS, based on GRU, is proposed as a comparative model to LMIDS for identifying attack patterns in vehicle CAN data.

The GRU model uses the following gates to control the flow of information:

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

Equation (4) defines the reset gate r_t and the update gate z_t, which regulate the blending of previous hidden states with new input information. Both gates pass through the sigmoid activation function δ

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

In (5), the reset gate r_t controls how much of the previous hidden state h_{t-1} should be discarded and replaced by the current input x_t

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

Equation (6) computes the update gate z_t, which controls the extent to which the model retains previous hidden state h_{t-1} or updates it with new information $h_t^{\sim}$.

The GMIDS model, similar to LMIDS, learns normal behavior patterns from the temporal sequences of CAN messages, using the reset and update gates to capture short-term dependencies and identify anomalous behavior indicative of attacks.

C. TIDS Model Construction

The Transformer model [43] is a sequence-to-sequence architecture comprising an encoder and a decoder. Originally applied in machine translation, it utilizes solely the encoder, and the resultant encoder output serves as a representation for input sequences in classification or sequence labeling tasks.

The proposed Transformer-based intelligent vehicle intrusion detection system (TIDS) [43] utilizes a self-attention mechanism to dynamically compute weights for connections between input and output. This process allows the model to capture complex temporal and spatial patterns in vehicle

Algorithm 1 Generalized Intrusion Detection Algorithm

Input: Raw CAN messages $\mathcal{D}$
Output: Predicted labels y or anomaly scores $\mathcal{A}$
Step 1: Data Preprocessing
Extract and normalize features X' from $\mathcal{D}$
Split dataset into $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$
Step 2: Model Initialization
if LMIDS (LSTM-based) **then**
  Initialize LSTM model $\mathcal{M}_{LSTM}$
else if GMIDS (GRU-based) **then**
  Initialize GRU model $\mathcal{M}_{GRU}$
else if TIDS (Transformer-based) **then**
  Initialize Transformer model $\mathcal{M}_{TIDS}$
end if
Step 3: Feature Extraction
if LMIDS **then**
  Use LSTM to capture temporal dependencies
else if GMIDS **then**
  Use GRU to capture short-term dependencies
else if TIDS **then**
  Apply self-attention and positional encoding
end if
Step 4: Classification
Apply classification layer: $y = \text{softmax}(f(X'))$
Step 5: Anomaly Detection
Calculate anomaly score $\mathcal{A}$
if $\mathcal{A} > \mathcal{A}_{\text{threshold}}$ **then**
  Flag as Anomaly
else
  Flag as Normal
end if
Step 6: Output
Return y or $\mathcal{A}$

data, crucial for detecting both known and unknown attack behaviors. The attention mechanism can be mathematically expressed as follows:

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

where Q, K, and V represent the query, key, and value vectors, respectively. The input X is projected into these spaces using learnable weight matrices W_Q, W_K, and W_V.

The multihead Attention mechanism is utilized to process multiple attention heads in parallel, each focusing on different features of the input data, and then aggregating the results as shown in

$$\begin{aligned} \text{MultiHead}(Q, K, V) = \text{Concat}(&\text{Attention}(Q_i, K, V) \\ &\cdots \textit{Attention}(Q_k, K, V)). \end{aligned} \tag{8}$$

This allows the model to attend to multiple aspects of the input simultaneously, improving the feature extraction capabilities. To preserve temporal patterns, the model incorporates positional encodings, as shown in

Algorithm 2 Transfer Learning for Intrusion Detection

Input: Source domain dataset $D_s = \{\mathcal{G}^s, P(G^s)\}$, Target domain dataset $D_t = \{\mathcal{G}^t, P(G^t)\}$
Output: Adapted model for target domain
Step 1: Train Source Domain Model
Initialize model $f_s(\cdot)$, parameters θ_s
Train on source domain: $\theta_s \leftarrow \arg\min_{\theta_s} \mathcal{L}_{\text{cross-entropy}}(r_i^s, f_s(g_i^s))$
Step 2: Define Target Domain Task
Define target task: $T_t = \{\mathcal{R}^t, f_t(\cdot)\}$
Task goal: $f_t(g_i^t) \rightarrow r_i^t$
Step 3: Transfer Knowledge
Initialize target model with $\theta_t \leftarrow \theta_s$
Transfer learning: $f_t(g_i^t, \theta_t) \leftarrow P(r_i^t | g_i^t, \theta_t)$
Step 4: Domain Similarity Evaluation
Compute MMD between D_s and D_t
Step 5: Feature Extraction
Extract target features using $f_s(\cdot)$
Map $g_i^t \rightarrow \mathcal{G}^s$
Step 6: Fine-Tuning
Freeze lower layers, unfreeze higher layers
Fine-tune: $\theta_t \leftarrow \arg\min_{\theta_t} \mathcal{L}_{\text{task}}(r_i^t, f_t(g_i^t, \theta_t))$
Step 7: Adapted Model
Apply adapted model to D_t
Output: Final model $f_t(\cdot)$ for target domain detection

$$\begin{cases} \text{PE}_{\text{pos},2i} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \\ \text{PE}_{\text{pos},2i+1} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right). \end{cases} \tag{9}$$

These encodings help the model maintain information about the relative positions of elements within the sequence, which is critical for detecting temporal attack patterns.

D. Transfer Learning Construction

Given the similarity between the source and target domains in intelligent vehicle intrusion detection, we applied an instance-based transfer learning approach [44]. The pseudocode for the transfer learning process for intrusion detection is described in Algorithm 2.

1) Source Domain Model Training: The initial phase of our transfer learning process involves training a model on the source domain using the Car Hacking dataset, which consists of feature vectors and their corresponding labels. Let the source domain dataset D_s be represented by the feature space $\mathcal{G}^s$ and the marginal probability distribution $P(G^s)$, where $G^s = \{g_1^s, g_2^s, \ldots, g_n^s\} \in \mathcal{G}^s$ denotes the feature vectors. The goal is to learn a function that maps the feature space $\mathcal{G}^s$ to the label space $\mathcal{R}^s$, where $r_i^s \in \mathcal{R}^s$ represents the corresponding labels, by minimizing the cross-entropy loss. The source domain dataset is formulated as

$$D_s = \big\{\mathcal{G}^s, P\big(G^s\big)\big\}. \tag{10}$$

The objective is to learn the representation of normal and attack behaviors in the source domain through the model.

2) Target Domain Task Definition: Subsequently, we acquired the target domain dataset, namely the ROAD dataset,

which encompasses a diverse set of intelligent vehicle intrusion attack scenarios. The objective in the target domain is to learn a model capable of predicting the labels associated with the feature vectors in this dataset. Let the target domain dataset D_t be defined by the feature space $\mathcal{G}^t$ and the label space $\mathcal{R}^t$, where $g_i^t \in \mathcal{G}^t$ represents the feature vectors, and $r_i^t \in \mathcal{R}^t$ denotes the corresponding labels. The target task T is characterized by the label space and the prediction function, as described in

$$T = \left\{\mathcal{R}^t, f(\cdot)\right\}. \tag{11}$$

The goal is to train a model that can map the feature vectors g_i^t in the target domain D_t to their corresponding labels r_i^t. Let $G^t = \{g_1^t, g_2^t, \ldots, g_n^t\} \in \mathcal{G}^t$ represent the feature vectors in the target domain. The objective is to learn a function that maps the feature space $\mathcal{G}^t$ to the label space $\mathcal{R}^t$, where $r_i^t \in \mathcal{R}^t$ denotes the labels of the feature vectors g_i^t.

3) Transfer Learning Objective: The goal of transfer learning is to leverage the knowledge learned from the source domain to improve the model's performance in the target domain. This is achieved by maximizing the prediction function $f_t(\cdot)$ in the target domain D_t, as shown in (12). Specifically, the model uses the source domain data D_s to learn a mapping that improves the predictions in the target domain D_t

$$T_t = \{r_t, f_t(D_t|D_s)\}. \tag{12}$$

In this context, the model utilizes the feature representations learned from the source domain D_s to improve its performance on the target domain task.

4) Domain Similarity Evaluation: To evaluate the domain shift between the source and target domains, we employed the maximum mean discrepancy (MMD) metric [45]. MMD quantifies the discrepancy between the feature distributions of the source and target domains. The kernel function used for computing MMD is defined as

$$\acute{\eta}\left(\mu_i, \nu_j\right) = \left\langle \mu_i, \nu_j \right\rangle \tag{13}$$

where $\langle \mu_i, \nu_j \rangle$ represents the dot product of vectors μ_i and ν_j. The calculating MMD distance is shown in (14) at the bottom of the page, where Ψ and Π are two sample sets, m and n are the number of samples in each set, $\acute{\eta}(\psi_i, \psi_j)$ denotes the kernel function value between samples ψ_i and ψ_j, $\acute{\eta}(\pi_i, \pi_j)$ denotes the kernel function value between samples π_i and π_j, and $\acute{\eta}(\psi_i, \pi_j)$ denotes the kernel function value between samples ψ_i and π_j. The MMD distance serves as a quantitative measure of domain similarity, ensuring the feasibility of transferring knowledge from the source to the target domain.

5) Feature Extraction and Fine Tuning: Following domain similarity evaluation, we employed a feature extraction method based on reusing the pretrained model. This approach utilizes all layers of the source domain model to extract relevant features from the target domain data. The target domain data is then mapped to the feature space of the source domain. To adapt the model to the target domain, we performed fine tuning: the lower layers are frozen to retain source domain knowledge, while the higher layers are unfrozen for task-specific adaptation [46]. This process allows the model to maintain valuable source domain features while learning new domain-specific features.

6) Adapting the Model to the Target Domain: The transfer learning approach enables the model to adapt to the target domain's data distribution and feature space. By transferring knowledge from the source domain, the model improves its performance on intrusion detection tasks in the target domain, as represented by the ROAD dataset. This adaptation enhances the model's ability to detect previously unseen or underrepresented intelligent vehicle intrusions.

E. Self-Supervised Box Classification Model Construction

Intelligent vehicle intrusion detection faces diverse and evolving attacks. Supervised classification requires labeled attack data for training but cannot detect new, unseen attacks. This article proposes a self-supervised box classification approach, where the model learns the characteristics of normal data and identifies anomalies to detect intrusions, including novel attacks. The pseudocode for the self-supervised box classification model for intrusion detection, including anomaly detection and novel attack identification, is described in Algorithm 3.

1) Detection of Novel Unknown Attacks: The key advantage of this self-supervised approach is its ability to detect novel attack types. Unlike traditional supervised models that rely on labeled attack data, the self-supervised model learns the system's normal behavior, constructing a "normality" representation. When presented with new, unseen attack data, the model's reconstruction error increases due to the deviation from learned normal patterns, indicating an anomaly. This enables the model to identify novel or evolving attacks based solely on their deviation from normal behavior, without prior exposure to such attacks.

2) Model Structure: The model uses an encoder–decoder structure. The encoder extracts time-series features (both global and local), and the decoder reconstructs the data from these features. Fig. 2 illustrates the model framework. LMIDS, GMIDS, or TIDS are used as the encoder, processing input data and generating feature matrices. These features are updated iteratively, as shown in

$$\begin{aligned}\text{hidden}_{\text{new}} &= f(\text{encoder\_output}, \text{hidden}_{\text{old}}, \text{parameters}) \\ &= \text{elu}\left(\text{encoder\_output} \cdot \theta_f + \text{hidden}_{\text{old}} \cdot \phi_f + b_f, \alpha_{\text{elu}}\right)\end{aligned} \tag{15}$$

where $\text{hidden}_{\text{new}}$ denotes the new feature matrix, encoder_output is the output matrix from the encoder,

$$\text{MMD}(\Psi, \Pi) = \sqrt{\frac{\sum_{i=1}^{m}\sum_{j=1}^{m}\acute{\eta}\left(\psi_i, \psi_j\right)}{m(m-1)} + \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\acute{\eta}\left(\pi_i, \pi_j\right)}{n(n-1)} - \frac{2\sum_{i=1}^{m}\sum_{j=1}^{n}\acute{\eta}\left(\psi_i \pi_j\right)}{mn}} \tag{14}$$

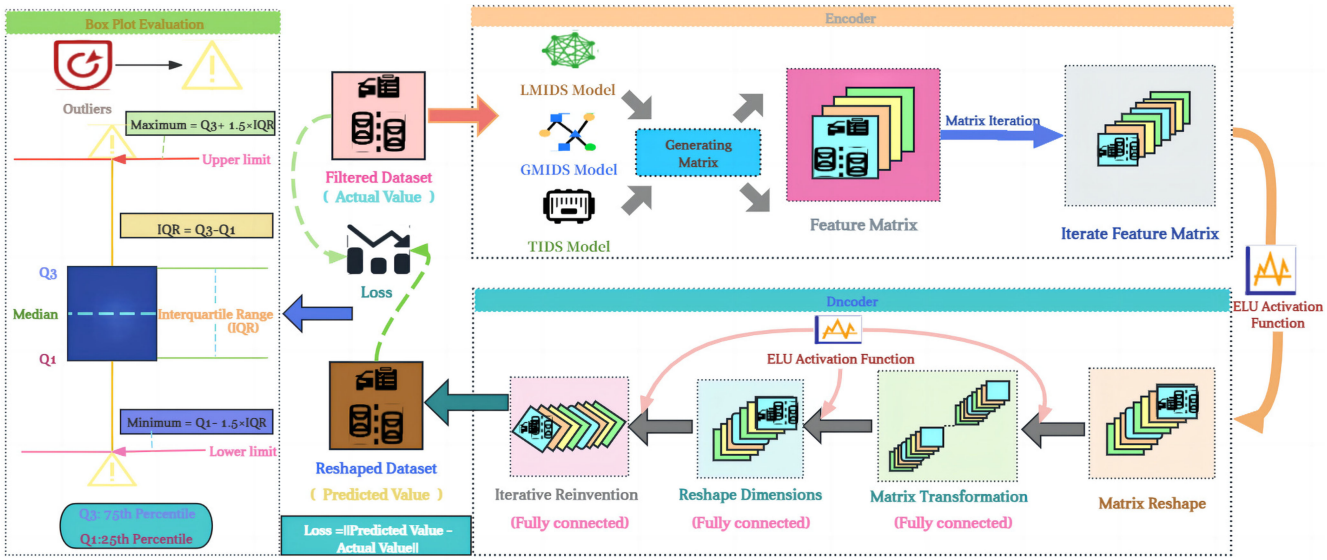


Fig. 2. Self-supervised box classification model structure.

$\text{hidden}_{\text{old}}$ is the feature matrix from the previous iteration, and f is the feature matrix update function. The elu represents the ELU activation function, and parameters are the necessary parameters for the f function. Specifically, θ_f and ϕ_f are the weight matrices, b_f is the bias vector, and α_{elu} is the parameter for the ELU activation function. The ELU activation function is defined as

$$\text{elu}(x, \alpha) = \begin{cases} x, & \text{if } x > 0 \\ \alpha(\exp(x) - 1), & \text{otherwise} \end{cases} \tag{16}$$

where x is the input and α is the ELU parameter, typically set to 1.0. After feature extraction, the fully connected layer size is adjusted to match the input size for loss calculation, as shown in

$$\text{output} = \text{hidden} \cdot \Lambda + \zeta \tag{17}$$

where hidden is the feature matrix, Λ is the weight matrix, and ζ is the bias vector. In Fig. 2, the arrows marked with ELU represent the operation of the fully connected layer.

3) Reconstruction Error Calculation: The data is input into the model, and the encoder extracts features. The decoder then reconstructs the data and compares it with the original to compute the loss. Among various loss functions, L_2 loss (mean-squared error, MSE) is chosen for its effectiveness in detecting anomalies. The L_2 loss is defined as

$$L_2 \text{ Loss(MSE)} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2 \tag{18}$$

where x_i represents the actual value of the ith data point, $\hat{x}_i$ is the predicted or reconstructed value of the ith data point, and n is the total number of data points. The L_2 loss measures the squared differences between the reconstructed data and the original data, which helps the model to identify and quantify potential intrusion behaviors.

4) Reconstruction Error Distribution and Anomaly Detection: Once the reconstruction errors are computed using L_2 loss, all sample error values can be used to construct a box plot. The box plot includes the following key elements:

1) the first quartile (Q1): the 25th percentile of the reconstruction error values;

2) the third quartile (Q3): the 75th percentile of the reconstruction error values; and

3) the interquartile range (IQR): $\text{IQR} = Q3 - Q1$, the range between the first and third quartiles.

The box plot can be used to define the range for normal reconstruction errors. Assume that the lower and upper bounds are defined as $Q1 - 1.5 \times \text{IQR}$ and $Q3 + 1.5 \times \text{IQR}$. Any data points with reconstruction errors beyond these bounds are considered anomalies.

5) Threshold Adjustment for Anomaly Detection: To refine anomaly detection, a threshold parameter $h\%$ is introduced to adjust the sensitivity. Let $\Delta \in [a, b]$ represent the normal error range, where a and b are the lower and upper bounds. The anomaly detection condition is then defined as

$$Loss_B \in [0, a + (b - a) \times h\%] \cup [b - (b - a) \times h\%, +\infty]. \tag{19}$$

By adjusting $h\%$, the model's sensitivity can be controlled, balancing between false positives and false negatives.

6) Anomaly Classification: After comparing the reconstruction errors of new data with the defined normal range, any data point with a reconstruction error outside the normal range is considered an anomaly, suggesting a potential intrusion. Particularly in the context of intelligent vehicle intrusion detection, it is more critical to minimize false negatives (missed detections) than false positives (incorrect detections). Thus, the focus of this evaluation is on the model's ability to detect anomalies within intelligent vehicle network data, rather than identifying normal data. From the broader perspective of intelligent vehicle intrusion detection, prioritizing caution over false negatives is advisable.

Algorithm 3 Self-Supervised Box Classification Model

Input: $X = \{x_1, x_2, \ldots, x_n\}$ {Time-series data}
Output: Anomaly classifications $C = \{c_1, c_2, \ldots, c_n\}$
Initialize Encoder-Decoder model: $f(\cdot), g(\cdot)$
Initialize weights: θ_f, ϕ_f, b_f
Initialize hidden states: $hidden_{\text{old}} = 0$
Initialize anomaly threshold: $h\%$
for each iteration $t = 1, 2, \ldots, T$ **do**
  Step 1: Feature Extraction:
  $encoder\_output_t = f(X_t, \theta_f, hidden_{\text{old}})$ {Encoder output at time t}
  $hidden_{\text{new}} = \text{ELU}(encoder\_output_t \cdot \theta_f + hidden_{\text{old}} \cdot \phi_f + b_f)$
  $hidden_{\text{old}} \leftarrow hidden_{\text{new}}$
  Step 2: Reconstruction Error Calculation:
  $\hat{X}_t = g(hidden_{\text{new}})$ {Reconstructed data}
  $L_2\text{Loss}(X_t, \hat{X}_t) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$
  Step 3: Error Distribution:
  Compute quantiles: $Q1, Q3, IQR = Q3 - Q1$
  Lower bound: $a = Q1 - 1.5 \times IQR$
  Upper bound: $b = Q3 + 1.5 \times IQR$
  Step 4: Anomaly Detection:
  for each $x_i \in X$ **do**
    if $(x_i - \hat{x}_i)^2 > b$ or $(x_i - \hat{x}_i)^2 < a$ **then**
      $a_i = 1$ {Anomaly detected}
    else
      $a_i = 0$ {Normal behavior}
    end if
  end for
  Step 5: Threshold Adjustment:
  Define dynamic range for anomaly threshold:

$$Loss_B \in [0, a + (b-a) \times h\%] \cup [b - (b-a) \times h\%, +\infty]$$

  Step 6: Anomaly Classification:
  Classify anomalies based on error deviation:

$$C = \{c_1, c_2, \ldots, c_n\}$$

end for

V. Experimental Results and Analysis

The experiments were conducted on a system equipped with an AMD Ryzen 5 5600H CPU (6 cores and 12 threads) and 16 GB of RAM, running Windows 10, without the use of GPU acceleration. The implementation was carried out in Python 3.9 using the PyTorch deep learning framework within the Jupyter IDE.

To validate the proposed framework, we compared its performance against conventional machine learning and deep learning methods using two prominent intelligent vehicle cybersecurity datasets: 1) the Car-Hacking dataset [16], [33] and 2) the ROAD dataset [39]. A four-fold cross-validation strategy was employed to ensure robustness and mitigate potential overfitting. Given the imbalanced nature of these datasets, evaluation metrics included accuracy, precision, recall, F1-score, and false negative rate (FNR). Additionally, we analyzed the training duration of each model to assess computational efficiency.

TABLE IV
Average Experimental Results from Cross-Validation on the Car Hacking Test Dataset

Method	Accuracy	Precision	Recall	F1-score	FNR
NB	80.73%	76.32%	76.32%	76.31%	23.67%
DTREE	89.24%	89.24%	89.24%	89.24%	10.76%
SVM	89.19%	89.19%	89.19%	89.19%	10.81%
BP	95.86%	95.83%	95.83%	95.80%	4.17%
KNN	90.41%	90.41%	90.41%	90.41%	9.59%
CNN	99.18%	99.42%	98.17%	98.74%	1.83%
GMIDS	99.77%	99.79%	99.80%	99.80%	0.20%
LMIDS	99.38%	99.45%	99.37%	99.41%	0.63%
DNN [50]	97.82%	95.22%	95.22%	95.22%	4.78%
GIDS [33]	96.53%	96.53%	98.65%	97.56%	1.35%
P-LeNet [45]	98.15%	97.98%	97.98%	97.81%	2.02%
DBN-LSTM [47]	99.13%	98.32%	98.73%	98.52%	1.27%
DCNN [16]	99.68%	99.62%	99.62%	99.56%	0.38%
TLSE-ID [48]	95.43%	96.83%	97.46%	97.05%	2.54%
H-IDFS [49]	98.35%	99.43%	96.71%	98.06%	3.29%
TIDS	**99.85%**	**99.86%**	**99.88%**	**99.87%**	**0.12%**

A. Experiments on the Car Hacking Dataset

The Car-Hacking dataset was partitioned into four subsets of equal size. During each iteration of cross-validation, one subset was used as the validation set, while the remaining three subsets served as the training set. This process was repeated four times, ensuring that each subset was used as the validation set exactly once.

1) Experimental Results and Discussion: We conducted experiments on the Car-Hacking dataset to evaluate the performance of the proposed TIDS model, comparing it with various baseline classifiers, including naive Bayes (NB), decision tree (DTREE), support vector machine (SVM), back-propagation (BP) neural network, K-nearest neighbors (KNN), as well as advanced methods like CNN, GMIDS, LMIDS, and several recent techniques, such as DBN-LSTM [47], TLSE-ID [48], and H-IDFS [49]. The average results from the cross-validation are presented in Table IV.

The TIDS model achieved a detection accuracy of 99.85%, significantly outperforming traditional machine learning models and demonstrating superior precision in identifying malicious traffic. It effectively integrated spatiotemporal features, which enhanced its ability to capture complex intrusion patterns in vehicle network traffic.

When compared to recent deep learning approaches, such as DBN-LSTM [47], TLSE-ID [48], and H-IDFS [49], the TIDS model showed marked improvements. Specifically, TIDS minimized the FNR to 0.12%, which highlights its robustness in detecting various types of attacks. This result demonstrates the importance of incorporating both spatial and temporal features to enhance detection capabilities, significantly reducing the likelihood of misclassification.

The BP neural network, despite its nonlinear feature learning, had a higher FNR of 4.17%, suggesting its limitations in handling complex temporal dependencies. The comparative analysis illustrates that the TIDS model's design, which emphasizes the extraction and fusion of spatiotemporal attributes, provides a significant advantage in intelligent vehicle intrusion detection.

TABLE V
MODULE STACKING TRAINING RESULTS ON THE CAR HACKING DATASET

Cond[1]	Sqe[2]	Hs[3]	$LMIDS_1$	$LMIDS_2$	$LMIDS_3$	$GMIDS_1$	$GMIDS_2$	$GMIDS_3$	$TIDS_1$	$TIDS_2$	$TIDS_3$
RES[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
RPD[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**	–	–
RDM[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 Car Hacking Dataset during early stopping
[5] Increase the training results of pooling different models on the Car Hacking Dataset [6] Training results of different models on the Car Hacking Dataset

Comparing TIDS with other recent methods, the latter models had lower accuracy and higher FNR, confirming the superiority of TIDS in both precision and recall. These findings emphasize the importance of designing IDSs that can handle both spatial and temporal dimensions effectively, ensuring robust and accurate identification of threats in dynamic and complex vehicle network environments.

2) Ablation Study on Module Stacking With Car Hacking Dataset: To explore potential architectural improvements and enhance our model's training accuracy, we performed ablation experiments by incorporating additional modules. These experiments aimed to assess the impact of module stacking on enhancing intrusion detection in intelligent vehicles, providing insights into optimizing model configurations.

In this study, we focused on evaluating the stacking of modules within the GMIDS, LMIDS, and TIDS methods, which previously demonstrated optimal performance. Module 1 serves as the baseline, employing the original preprocessing pipeline. Here, the processed data is expanded into an 1-D vector, passed through a fully connected layer, and classified into five categories. Module 2 adds a linear layer of length 256 before the classification stage, enhancing feature representation. Module 3 further introduces an 1-D convolutional layer, followed by a fully connected layer to capture sequential patterns.

Table V summarizes the results using the early stopping strategy on the car hacking dataset. The table shows that the newly introduced modules can improve accuracy, especially with smaller hidden sizes. However, as the hidden size increases, the performance gains diminish, and some modules even reduce accuracy despite faster convergence.

The TIDS model consistently exhibits robust performance in both accuracy and training time. Although GMIDS shows faster convergence due to its simpler structure, it sacrifices accuracy. Conversely, the TIDS model achieves higher accuracy while requiring only half the training time compared to LMIDS.

Introducing maximum pooling, a common technique to reduce dimensionality and computational cost, surprisingly results in decreased accuracy and increased training time. As seen in Table V, pooling negatively affects model performance. Based on these observations and following the principle of simplicity (Occam's razor), the original model with a fully connected layer proves to be the optimal choice.

The performance comparison in Table V reveals that increasing the hidden size enhances accuracy for both LMIDS and GMIDS but significantly prolongs training time. In contrast, TIDS maintains a runtime of approximately 2 minutes, achieving a comparable accuracy. The time ratio for LMIDS, GMIDS, and TIDS is about 18:12:5, respectively. Although the training duration is not excessive, real-time detection requires rapid processing. The TIDS model effectively reduces processing time, enhancing the system's ability to promptly identify abnormalities.

Fig. 3 presents the heatmap of the classification results using our proposed models (LMIDS, GMIDS, and TIDS), while Fig. 4 shows the results of a standard CNN model. Our models achieve higher accuracy in detecting various attacks, particularly in gear and RPM spoofing, demonstrating superior performance over CNN in time series tasks. Fig. 5 further illustrates the classification results using the CNN+LSTM method [52]. Although it focuses on binary tasks, its performance in detecting fuzzy attacks is inferior, highlighting the challenges of CNN-based approaches. In contrast, the TIDS model shows higher accuracy and faster computation, making it a more effective solution.

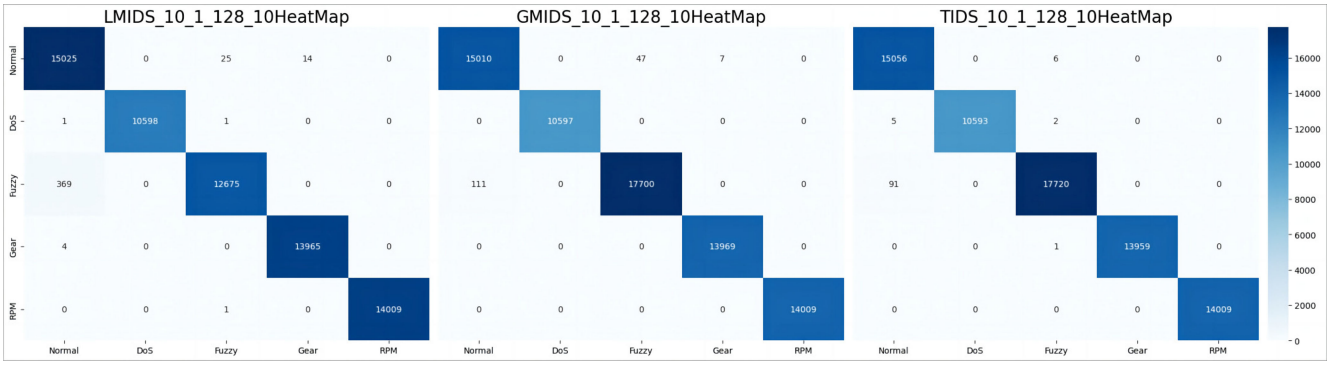


Fig. 3. Classification heat map of the car hacking dataset.

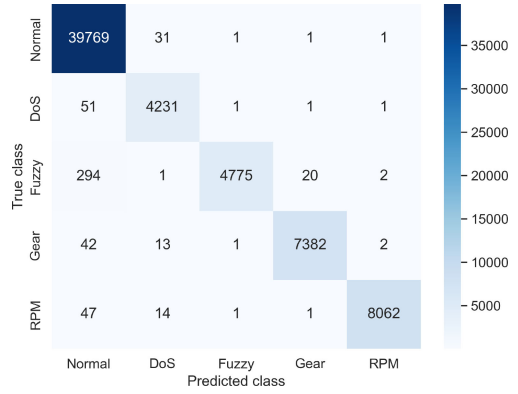


Fig. 4. Performance of the CNN model [51] on the car hacking dataset.

TABLE VI
OPTIMIZED PARAMETERS FOR THE ROAD DATASET

Parameter	Epoch	BatchSize	LearningRate	Optimizer	Classes
Value	20	64	0.0005	Adam	6

TABLE VII
PERFORMANCE EVALUATION FOLLOWING FINE-TUNING STRATEGY ON THE ROAD DATASET

Sql_len	Hidden size	LMIDS	GMIDS	TIDS
10	**32**	0.9375	0.9418	**0.9752**
		01:06	**00:53**	01:12
	64	0.9838	0.9804	**0.9892**
		01:44	01:18	**01:14**
	128	0.9954	**0.9972**	0.9942
		03:17	02:33	**01:18**

TABLE VIII
COMPARISON OF IMPROVEMENT RESULTS WITH STACKED MULTIHEAD SELF-ATTENTION MECHANISM IN TIDS

Sql_len	Hidden size	Layer	TIDS
10	**128**	**1**	0.9942
			01:18
		2	0.9974
			01:59
		3	**0.9981**
			02:45

B. Experiments on the ROAD Dataset

To validate the TIDS model's adaptability for intelligent vehicle intrusion detection, we adopted an instance-based transfer learning strategy, using MMD to assess dataset similarity between the car hacking and ROAD datasets[33], [39]. Figs. 6 and 7 illustrate the MMD distances after PCA reduction from random samples of 1000 points each. The overlapping distribution of samples confirms the high similarity between the datasets, facilitating effective transfer learning.

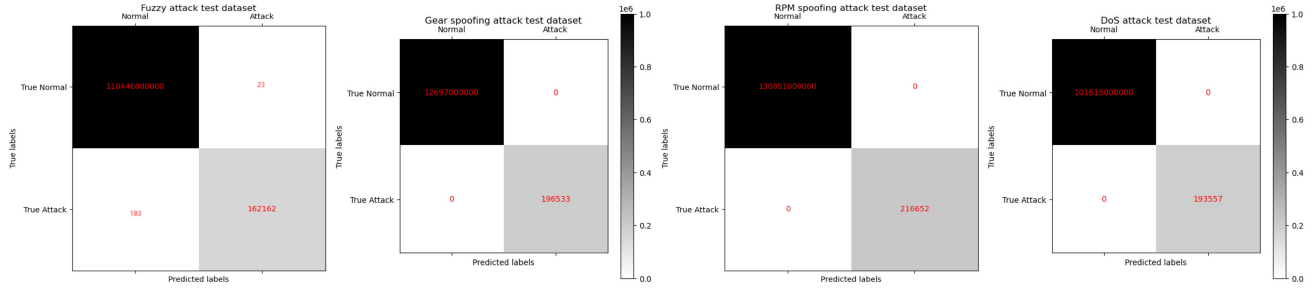
To fine-tune the model, we reused feature extractors and optimized key parameters as listed in Table VI. This adjustment maximizes the model's adaptability on the ROAD dataset.

We adopted a fine-tuning strategy by freezing the TIDS encoder layers while updating the decoder layers to leverage pretrained features effectively. Given the limited size of the ROAD dataset, we applied four-fold cross-validation with a 3:1 training-to-test ratio to ensure robustness. Table VII displays the performance outcomes of TIDS, GMIDS, and LMIDS under different configurations, highlighting that TIDS achieved higher accuracy at smaller hidden sizes, but performance slightly decreased with larger hidden sizes, indicating potential feature extraction limitations.

To enhance feature extraction, we integrated a stacked multihead self-attention mechanism into TIDS. Table VIII shows that this enhancement improved accuracy without significantly increasing processing time, outperforming LMIDS and GMIDS models.

Utilizing the optimized parameters, we compared TIDS with KNN, GMIDS, LMIDS, DNN [50], GIDS [33], and added DBN-LSTM [47] and H-IDFS [49] models. As shown in Table IX, TIDS achieved the highest accuracy (99.52%), precision, and recall, demonstrating its effectiveness in detecting known intelligent vehicle intrusion attacks.

While DBN-LSTM [47] achieved 97.61% accuracy with high precision and recall values of 97.83%, it still lagged slightly behind TIDS in overall performance. Similarly, H-IDFS [49] performed well with an accuracy of 95.48%, but its precision and recall were comparatively lower at 94.98% and 97.98%, respectively. Despite these results, both models demonstrated significant potential in intrusion detection tasks, with DBN-LSTM achieving a notable FNR of 2.17%, and H-IDFS having a lower FNR of 2.02%.



(a) (b) (c) (d)

Fig. 5. Classification results using CNN+LSTM [52]. (a) Fuzzy attack. (b) Gear spoofing attack. (c) RPM spoofing attack. (d) DoS attack.

TABLE IX
CROSS-VALIDATION PERFORMANCE SUMMARY ON THE ROAD TEST DATASET

Method	Accuracy	Precision	Recall	F1-score	FNR
KNN	91.62%	91.48%	90.48%	90.98%	9.52%
GMIDS	99.28%	99.64%	99.64%	99.64%	0.36%
LMIDS	98.77%	99.38%	99.38%	99.38%	0.62%
DNN [50]	97.62%	96.71%	96.71%	95.82%	3.28%
GIDS [33]	96.84%	96.30%	96.30%	95.76%	3.70%
DBN-LSTM [47]	97.61%	97.83%	97.83%	97.83%	2.17%
H-IDFS [49]	95.48%	94.98%	97.98%	96.47%	2.02%
TIDS	**99.52%**	**99.76%**	**99.76%**	**99.76%**	**0.24%**

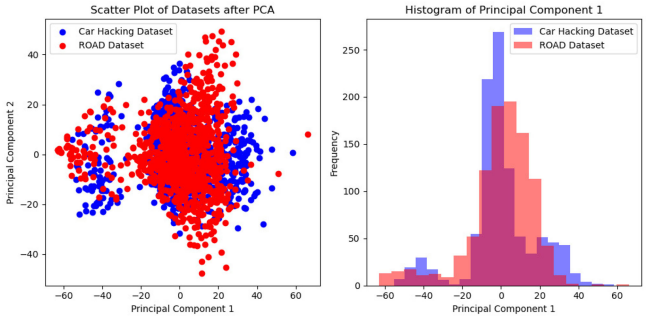


Fig. 6. Assessing dataset similarity using MMD: first random sampling.

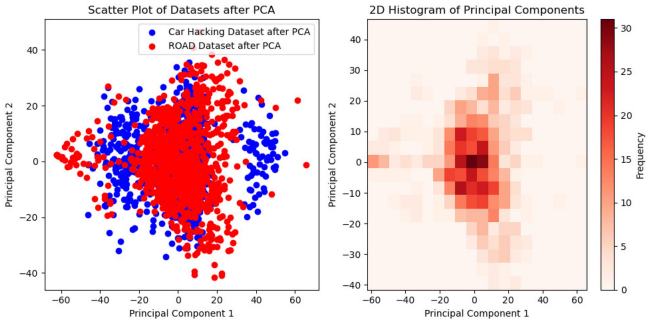
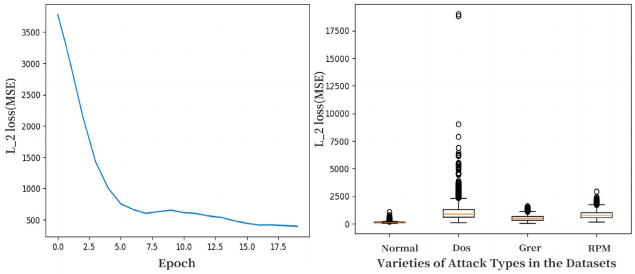


Fig. 7. Assessing dataset similarity using MMD: second random sampling.

In contrast, KNN performed the worst, with low precision and recall, while GMIDS and LMIDS had similar but slightly lower performance than TIDS. These results highlight TIDS's robustness and superior feature extraction capabilities, especially after applying transfer learning.

The performance of LMIDS and GIDS models was limited due to small sample sizes for certain attack types, while DNN and GMIDS models struggled with learning temporal features. In contrast, integrating multihead self-attention within the TIDS framework enhanced feature extraction, effectively addressing both data scarcity and temporal learning challenges. This improvement leads to more accurate intrusion detection in intelligent vehicles. Additionally, the use of transfer learning allows for efficient adaptation to domain shifts, highlighting its potential for future advancements. These results emphasize the importance of transfer learning in overcoming the limitations of traditional models and suggest promising avenues for future research in intelligent vehicle intrusion detection.



(a) (b)

Fig. 8. Change in (a) LMIDS loss and (b) LMIDS boxplot distribution.

C. Experimental Testing of Self-Supervised Box Classification

In this experiment, the Car Hacking dataset, commonly used for supervised classification, is adapted for self-supervised box classification. The goal is to detect unknown intrusion types, improving the sensitivity of in-vehicle IDSs against emerging threats.

For training, only normal (attack-free) data is used, while testing is conducted on datasets containing various attack types. After training, N points are randomly selected from normal data as benchmarks, and N points from each attack subset are evaluated for prediction. Figs. 8–10 show detailed results and visualizations. The dataset includes normal data and four attack types: DoS, Fuzzy, Gear, and RPM. Notably, Fuzzy attack data is excluded from the right column of the figures to avoid skewing due to large outliers.

From the boxplot distributions in Figs. 8–10, it is evident that the TIDS model minimizes the ranges of normal, gear, and RPM data, resulting in a relatively small distribution difference. Despite its proficiency in learning normal data, the TIDS model may still be susceptible to deception. In

TABLE X
LMIDS, GMIDS, AND TIDS CLASSIFICATION RESULTS

LMIDS, GMIDS and TIDS hidden= 20, 50, 20 epoch =20 lr=0.0005 time=17:17, 17:15, 30:06															
H	Average			Dos			Fuzzy			Gear			RPM		
IQR	**0.9982**	0.9959	0.98435	0.9993	0.9993	**1.0**	**1.0**	0.9993	0.9997	**0.9947**	0.986	0.9407	0.9987	**0.999**	0.997
0.25	**0.9939**	0.9846	0.9577	0.9977	0.9977	**1.0**	**0.9993**	0.998	0.9987	**0.9817**	0.9507	0.8417	**0.997**	0.992	0.9907
0.2	**0.9913**	0.9805	0.9484	0.997	0.9967	**1.0**	**0.9993**	0.998	0.998	**0.9737**	0.938	0.807	**0.9953**	0.9893	0.9887
0.15	**0.9884**	0.9745	0.9406	0.9957	0.9943	**1.0**	**0.9993**	0.9977	0.9967	**0.9643**	0.921	0.7797	**0.9943**	0.985	0.986
0.1	**0.986**	0.9680	0.9303	0.9957	0.9937	**1.0**	**0.999**	0.997	0.996	**0.9553**	0.9017	0.7443	**0.994**	0.9797	0.981
0.05	**0.9818**	0.9624	0.9197	0.9947	0.992	**1.0**	**0.999**	0.996	0.9953	**0.9407**	0.8867	0.7107	**0.9927**	0.9753	0.973
0	**0.9775**	0.9559	0.9106	0.993	0.99	**1.0**	**0.999**	0.993	0.9947	**0.927**	0.8697	0.6807	**0.991**	0.971	0.967

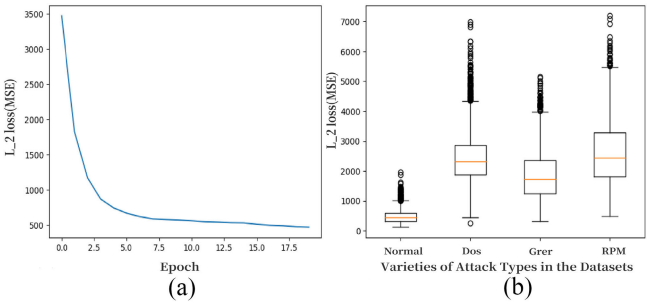


Fig. 9. Change in (a) GMIDS loss and (b) GMIDS boxplot distribution.

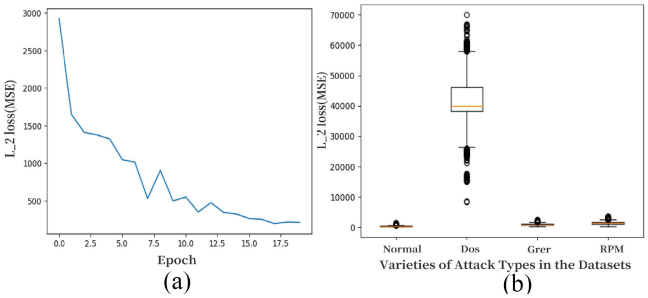


Fig. 10. Change in (a) TIDS loss and (b) TIDS boxplot distribution.

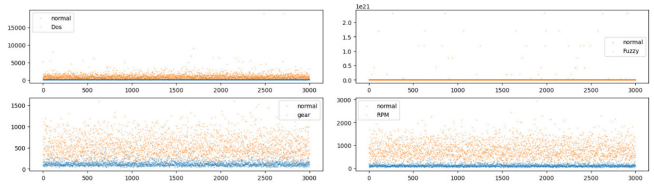


Fig. 11. Classification results of the LMIDS model.

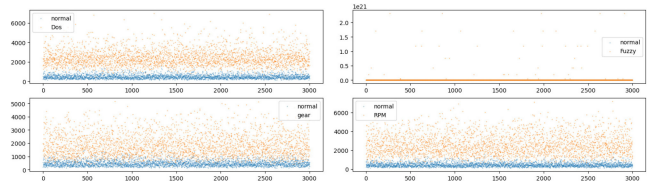


Fig. 12. Classification results of the GMIDS model.

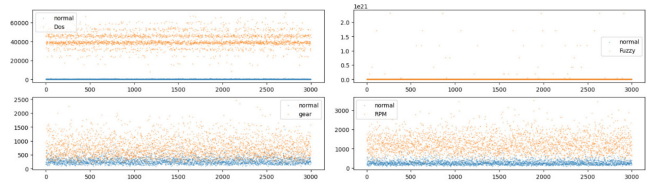


Fig. 13. Classification results of the TIDS model.

comparison, the LMIDS model, while also reducing the data range, exhibits a larger distribution difference compared to TIDS, contributing to its higher accuracy. Similarly, GMIDS does not significantly restrict the data range but demonstrates a larger distribution difference, leading to results superior to TIDS. This highlights the importance of considering data distribution variance over data range in evaluating model performance and accuracy in intelligent vehicle IDSs.

Regarding training loss, both LMIDS and GMIDS converge to around 500 with a rapid, smooth curve. In contrast, TIDS achieves convergence to under a hundred with a fluctuating curve. This characteristic enables TIDS to explore a wider range of optimization points during training, potentially facilitating the discovery of optimal solutions.

Figs. 11–13 illustrate the test results via scatter plots, where each point represents a test instance, showcasing the models' abilities to distinguish different attacks. Table X provides quantitative performance metrics, where "H" is the allowable error threshold and "Average" indicates the mean accuracy across four attacks.

Table X indicates that LMIDS achieves the highest accuracy, especially for fuzzy attacks, due to its effectiveness in capturing subtle data patterns. Although GMIDS shows slightly lower accuracy, it compensates with faster processing, making it suitable for real-time applications. TIDS, while having lower overall accuracy, excels in detecting DoS attacks, emphasizing the importance of considering specific attack characteristics when evaluating detection models.

These findings highlight the nuanced tradeoffs between accuracy, processing time, and attack detection capabilities, emphasizing the need for a tailored approach to model selection based on the specific requirements of the intelligent vehicle environment. The insights gained from this study offer valuable guidance for researchers and practitioners working in the field of intelligent vehicle intrusion detection, facilitating the development of more robust and effective IDSs.

VI. CONCLUSION

This article presents a novel Transformer-based transfer learning method (TIDS) for intelligent vehicle intrusion

detection. The TIDS model effectively captures both spatial and temporal features of attacks, offering superior detection compared to traditional IDS. By leveraging instance-based transfer learning, the model adapts well to new domains, achieving high accuracy in detecting known attack types. Additionally, the self-supervised box classification method enhances the model's ability to identify unknown attacks, improving the system's robustness. Empirical results show that TIDS outperforms conventional methods in detection accuracy and robustness, offering significant potential to enhance intelligent vehicle network security. This research addresses key challenges in integrating spatiotemporal features, domain adaptation, and unknown attack detection, laying a solid foundation for more efficient and adaptable security solutions. Future work will focus on improving feature extraction methods to enhance both accuracy and computational efficiency. We will continue optimizing transfer learning, particularly fine tuning and domain adaptation, and expand datasets to cover more attack types. Additionally, exploring hybrid models that combine different learning techniques may further address emerging complex attack patterns.

REFERENCES

[1] K. Koscher et al., "Experimental security analysis of a modern automobile," in *Proc. IEEE Symp. Security Privacy*, 2010, pp. 447–462.

[2] S. Teng, N. Wu, H. Zhu, L. Teng, and W. Zhang, "SVM-DT-based adaptive and collaborative intrusion detection," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 1, pp. 108–118, Jan. 2018.

[3] C. Schmittner and Z. Ma, "Towards a framework for alignment between automotive safety and security standards," in *Proc. 33rd Int. Conf. Comput. Safety Rel. Security (SAFECOMP)*, 2014, pp. 133–143.

[4] Q. Guan et al., "Exploiting the potential anomaly detection in automobile safety data with multitype neural network," in *Proc. Int. Conf. Mobile Ubiquitous Syst. Comput. Netw. Services*, 2023, pp. 489–501.

[5] Q. Kong, R. Lu, F. Yin, and S. Cui, "Blockchain-based privacy-preserving driver monitoring for MaaS in the vehicular IoT," *IEEE Trans. Veh. Technol.*, vol. 70, no. 4, pp. 3788–3799, Apr. 2021.

[6] Q.-L. Han, "Driving into the future with reliable, secure, efficient, and intelligent MetaVehicles," *IEEE/CAA J. Automatica Sinica*, vol. 10, no. 6, pp. 1355–1356, Jun. 2023.

[7] X. Duan, H. Yan, D. Tian, J. Zhou, J. Su, and W. Hao, "In-vehicle CAN bus tampering attacks detection for connected and autonomous vehicles using an improved isolation forest method," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 2122–2134, Feb. 2023.

[8] Q. Guan et al., "Transformer model with multitype classification decisions for intrusion attack detection of track traffic and vehicle," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, 2024, pp. 4510–4514.

[9] A. Tomlinson, J. Bryans, and S. A. Shaikh, "Towards viable intrusion detection methods for the automotive controller area network," in *Proc. 2nd ACM Comput. Sci. Cars Symp.*, 2018, pp. 1–9.

[10] M. Aloqaily, S. Otoum, I. A. Ridhawi, and Y. Jararweh, "An intrusion detection system for connected vehicles in smart cities," *Ad Hoc Netw.*, vol. 90, Jul. 2019, Art. no. 101842.

[11] A. Bertolino et al., "A tour of secure software engineering solutions for connected vehicles," *Softw. Qual. J.*, vol. 26, no. 4, pp. 1223–1256, 2018.

[12] A. Alshammari, M. A. Zohdy, D. Debnath, and G. Corser, "Classification approach for intrusion detection in vehicle systems," *Wireless Eng. Technol.*, vol. 9, no. 4, pp. 79–94, 2018.

[13] J. Xiao, L. Yang, F. Zhong, H. Chen, and X. Li, "Robust anomaly-based intrusion detection system for in-vehicle network by graph neural network framework," *Appl. Intell.*, vol. 53, no. 3, pp. 3183–3206, 2023.

[14] D. Ding, Y. Wei, C. Cheng, and J. Long, "Intrusion detection for in-vehicle CAN bus based on lightweight neural network," *J. Circuits Syst. Comput.*, vol. 32, no. 7, 2023, Art. no. 2350110.

[15] A. Taylor, S. Leblanc, and N. Japkowicz, "Anomaly detection in automobile control network data with long short-term memory networks," in *Proc. IEEE Int. Conf. Data Sci. Adv. Anal. (DSAA)*, 2016, pp. 130–139.

[16] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Veh. Commun.*, vol. 21, Jan. 2020, Art. no. 100198. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214209619302451

[17] R. Qaddoura, A. M. Al-Zoubi, H. Faris, and I. Almomani, "A multilayer classification approach for intrusion detection in IoT networks based on deep learning," *Sensors*, vol. 21, no. 9, p. 2987, 2021.

[18] L. Yang, A. Moubayed, and A. Shami, "MTH-IDS: A multitiered hybrid intrusion detection system for Internet of Vehicles," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 616–632, Jan. 2022.

[19] W. Wu, R. Li, G. Xie, J. An, Y. Bai, and J. Zhou, "A survey of intrusion detection for in-vehicle networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 919–933, Mar. 2020.

[20] F. Nobis, M. Geisslinger, M. Weber, J. Betz, and M. Lienkamp, "A deep learning-based radar and camera sensor fusion architecture for object detection," in *Proc. IEEE Sensor Data Fusion Trends Solutions Appl. (SDF)*, 2019, pp. 1–7.

[21] G. Karopoulos, G. Kambourakis, E. Chatzoglou, J. L. Hernández-Ramos, and V. Kouliaridis, "Demystifying in-vehicle intrusion detection systems: A survey of surveys and a meta-taxonomy," *Electronics*, vol. 11, no. 7, p. 1072, 2022.

[22] M. R. Islam, I. Oh, and K. Yim, "Universal intrusion detection system on in-vehicle network," in *Proc. Int. Conf. Innov. Mobile Internet Services Ubiquitous Comput.*, 2023, pp. 78–85.

[23] T. Zhang, W. Guan, H. Miao, X. Huang, Z. Liu, and C. Wang, "VSRQ: Quantitative assessment method for safety risk of vehicle intelligent connected system," *IEEE Trans. Veh. Technol.*, early access, Sep. 27, 2024, doi: 10.1109/TVT.2024.3469389.

[24] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *J. Big Data*, vol. 3, no. 1, pp. 1–40, 2016.

[25] Y. Xun, Y. Zhao, and J. Liu, "VehicleEIDS: A novel external intrusion detection system based on vehicle voltage signals," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 2124–2133, Feb. 2022.

[26] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*. Hershey, PA, USA: IGI Global, 2010, pp. 242–264.

[27] M. Müter, A. Groll, and F. C. Freiling, "A structured approach to anomaly detection for in-vehicle networks," in *Proc. IEEE 6th Int. Conf. Inf. Assurance Security*, 2010, pp. 92–98.

[28] S.-F. Lokman, A. T. Othman, and M.-H. Abu-Bakar, "Intrusion detection system for automotive controller area network (CAN) bus system: A review," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, pp. 1–17, Jul. 2019.

[29] Y. He, Q. Gui-He, S. M. Hui, Y. Xin, and W. X. Zhe, "Cyber security and anomaly detection method for in-vehicle CAN," *J. Jilin Univ.*, vol. 46, no. 4, pp. 1246–1253, 2016.

[30] R. Kulkarni, S. Dhavalikar, and S. Bangar, "Traffic light detection and recognition for self-driving cars using deep learning," in *Proc. 4th Int. Conf. Comput. Commun. Control Autom. (ICCUBEA)*, 2018, pp. 1–4.

[31] X. Wang, T. Xiao, Y. Jiang, S. Shao, J. Sun, and C. Shen, "Repulsion loss: Detecting pedestrians in a crowd," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 7774–7783.

[32] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[33] E. Seo, H. M. Song, and H. K. Kim, "GIDS: GAN-based intrusion detection system for in-vehicle network," in *Proc. 16th Annu. Conf. Privacy Security Trust (PST)*, Aug. 2018, pp. 1–6.

[34] K. Pawelec, R. A. Bridges, and F. L. Combs, "Towards a CAN IDS based on a neural network data field predictor," in *Proc. ACM Workshop Autom. Cybersecurity*, 2019, pp. 31–34.

[35] A. R. Javed, S. Ur Rehman, M. U. Khan, M. Alazab, and T. Reddy, "CANintelliIDS: Detecting in-vehicle intrusion attacks on a controller area network using CNN and attention-based GRU," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 1456–1466, Apr.–Jun. 2021.

[36] Q. Wang, D. Wang, C. Cheng, and D. He, "Quantum2FA: Efficient quantum-resistant two-factor authentication scheme for mobile devices," *IEEE Trans. Depend. Secure Comput.*, vol. 20, no. 1, pp. 193–208, Jan./Feb. 2023.

[37] M. Song and D. Wang, "AB-PAKE: Achieving fine-grained access control and flexible authentication," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 6197–6212, 2024.

[38] D. Javaheri, S. Gorgin, J.-A. Lee, and M. Masdari, "Fuzzy logic-based DDoS attacks and network traffic anomaly detection methods: Classification, overview, and future perspectives," *Inf. Sci.*, vol. 626, pp. 315–338, May 2023.

[39] M. E. Verma, M. D. Iannacone, R. A. Bridges, S. C. Hollifield, B. Kay, and F. L. Combs. "ROAD: The real ORNL automotive dynamometer controller area network intrusion detection dataset (with a comprehensive CAN IDS dataset survey & guide)." [Online]. Available: https://arxiv.org/abs/2012.14600
[40] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *Proc. NIPS Workshop Deep Learn.*, 2014, pp. 1–9.
[41] S. Longari, D. H. N. Valcarcel, M. Zago, M. Carminati, and S. Zanero, "CANnolo: An anomaly detection system based on LSTM autoencoders for controller area network," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 2, pp. 1913–1924, Jun. 2021.
[42] D. Ding, L. Zhu, J. Xie, and J. Lin, "In-vehicle network intrusion detection system based on bi-LSTM," in *Proc. 7th Int. Conf. Intell. Comput. Signal Process. (ICSP)*, 2022, pp. 580–583.
[43] C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, "Advances in Neural Information Processing Systems 28 (NIPS 2015)," in *Proc. 29th Annu. Conf. Neural Inf. Process. Syst.*, 2015, pp. 3294–3302.
[44] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[45] S. T. Mehedi, A. Anwar, Z. Rahman, and K. Ahmed, "Deep transfer learning based intrusion detection system for electric vehicular networks," *Sensors*, vol. 21, no. 14, p. 4736, 2021.
[46] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2208–2217.
[47] A. Chen, Y. Fu, X. Zheng, and G. Lu, "An efficient network behavior anomaly detection using a hybrid DBN-LSTM network," *Comput. Security*, vol. 114, Mar. 2022, Art. no. 102600.
[48] H. Du and Y. Zhang, "Network anomaly detection based on selective ensemble algorithm," *J. Supercomput.*, vol. 77, pp. 2875–2896, Jul. 2021.
[49] A. Derhab, M. Belaoued, I. Mohiuddin, F. Kurniawan, and M. K. Khan, "Histogram-based intrusion detection and filtering framework for secure and safe in-vehicle networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2366–2379, Mar. 2022.
[50] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PLoS ONE*, vol. 11, no. 6, 2016, Art. no. e0155781.
[51] M. D. Hossain, H. Inoue, H. Ochiai, D. Fall, and Y. Kadobayashi, "An effective in-vehicle CAN bus intrusion detection system using CNN deep learning approach," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2020, pp. 1–6.
[52] W. Lo, H. Alqahtani, K. Thakur, A. Almadhor, S. Chander, and G. Kumar, "A hybrid deep learning based intrusion detection system using spatial-temporal representation of in-vehicle network traffic," *Veh. Commun.*, vol. 35, Jun. 2022, Art. no. 100471.

Tian Zhang (Student Member, IEEE) received the Bachelor of Science degree in network space and information security from Guangdong University of Technology, Guangzhou, China, in 2022. He is currently pursuing the master's degree with Jinan University, Guangzhou.

His research interests include vehicle network intrusion detection, quantitative assessment of vehicle safety risk, and artificial intelligence.

Cuifeng Du received the master's degree in business administration from Guangdong University of Foreign Studies, Guangzhou, China, in 2010.

She is affiliated with the Operator Business Department, Cetc Potevio Science and Technology Company Ltd., Guangzhou. Her research interests focus on communication system security, electronic computing, and component security.

Yuyu Zhou received the M.S. degree in software engineering from South China University of Technology, Guangzhou, China, in 2006, and the Ph.D. degree in mechanical engineering from Huazhong University of Science and Technology, Wuhan, China, in 2012.

He has extensive research experience, having served as an Associate Researcher with Jinan University, Guangzhou. His research interests include genetic algorithm, simulated annealing, and vehicle security.

Quanlong Guan (Senior Member, IEEE) received the Ph.D. degree in management science and engineering from Jinan University, Guangzhou, China, in 2014.

He is a Professor with the Faculty of Computer Science and the Deputy Dean with the Research Institute for Guangdong Intelligent Education, Jinan University, Guangzhou. He leads the Guangdong Research and Development Institute, Guangzhou, focusing on big data applications in education. His research interests include smart education, artificial intelligence applications, system reliability and security, data protection, and safety enhancements.

Zhiquan Liu (Member, IEEE) received the B.S. degree from Xidian University, Xi'an, China, in 2012, and the Ph.D. degree in computer science from Xidian University in 2017.

He is currently an Associate Professor with the College of Cyber Security, Jinan University, Guangzhou, China. His research interests include trust management and privacy preservation in vehicular and UAV networks.

Xiujie Huang (Member, IEEE) received the Ph.D. degree in communication and information systems from Sun Yat-sen University, Guangzhou, China, in 2012.

She was a Postdoctoral Fellow with the University of Hawaii, Honolulu, HI, USA, from 2012 to 2013. Since November 2013, she has been with Jinan University, Guangzhou, where she is currently the Director with the College of Intelligent Systems Science and Engineering. Her research interests include genetic algorithm, simulated annealing, and vehicle security.

Zhiguo Gong (Senior Member, IEEE) received the Ph.D. degree in computer science from the Institute of Mathematics, Chinese Academy of Science, Beijing, China, in 1998.

He is a Professor with the Faculty of Science and Technology, University of Macau, Macau, China. His current research interests include machine learning, data mining, databases, and information retrieval.

Lianbing Deng (Member, IEEE) received the Doctoral degree from Huazhong University of Science and Technology, Wuhan, China, in 2013.

He is the Director of Guangdong Qinzhi Science and Technology Research Institute, Zhuhai, China. He serves as the General Manager of Zhuhai Da Hengqin Science and Technology Development Company Ltd., Zhuhai. He is the Vice Chairman of the China Big Data Council under the Ministry of Industry and Information Technology. His research interests include big data analytics, project management, and economic research.

Yang Li received the bachelor's degree in computer science and engineering from Guangdong Pei Zheng College, Guangzhou, China, in 2012.

He is a Research and Development Manager with the Smart Transportation R&D Department, Guangzhou Fundway Smart Transportation Research and Development Company Ltd., Guangzhou, and the Science and Technology Innovation Business Department, PCI Technology and Service Company Ltd., Guangzhou. His research focuses on vehicle networks and intelligent transportation systems tailored for railway infrastructures.