Future Generation Computer Systems 140 (2023) 266–281

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

DESC-IDS: Towards an efficient real-time automotive intrusion detection system based on deep evolving stream clustering

Pengzhou Cheng[a], Mu Han[b,c,*], Gongshen Liu[a]

[a] School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, 201100, China
[b] School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang 212013, China
[c] Jiangsu Key Laboratory of Security Technology for Industrial Cyberspace Jiangsu University, Zhenjiang 212013, China

ARTICLE INFO

Article history:
Received 22 March 2022
Received in revised form 6 October 2022
Accepted 20 October 2022
Available online 28 October 2022

Keywords:
In-vehicle network
Control area network
Anomaly detection
Unsupervised learning
Convolutional auto-encoder (CAE)
DenStream

ABSTRACT

Controller area network (CAN) is a widely used communication protocol for in-vehicle networks. With the up-gradation of traditional vehicle ad-hoc networks (VANETs) to the internet of vehicles (IoV), the connectivity is incremental between the vehicular network and the outside world, making cyber-security become a stringent problem. Although existing machine learning-based methods for automotive intrusion detection are powerful, there are still limitations in detection performance and resistance attack types during the unsupervised learning process that lacks massive amounts of labels. Therefore, this paper proposes an in-vehicle intrusion detection system that incorporates a combination of sparse regularization convolutional auto-encoder (SRCAE) and streams clustering to construct a deep evolving stream clustering model, namely DESC-IDS. Specifically, the method encodes continuous messages as 2-D data frames, which are fed into the SRCAE built by the temporal convolutional network (TCN) to obtain a low-dimensional non-linear spatial–temporal mapping of the high-dimensional data. Thereafter, the stream clustering model can describe a contour baseline of normal communication messages by the spatial–temporal features. Based on this baseline, DESC-IDS can detect any abnormal changes in vehicular communication. In particular, this paper exploits the SRCAE to reconstruct message matrices, which are considered as variants of known attacks due to reconstruction deviation. The extensive evaluation results illustrate that the proposed model provides enough performance and real-time competitiveness in anomaly detection, with 96.44% accuracy on the HCRL intrusion dataset and 98.80% accuracy on the ORNL intrusion dataset. For the mixed attack of fabrication and masquerade, the proposed model achieves stable F1-scores of 93.48% and 86.99%, respectively. Moreover, the performance in unknown attacks is righteous with 98.43% accuracy and 97.15% F1-scores.

1. Introduction

Initially, the in-vehicle system, which controls the vehicle status via mechanical linkages, is considered to be an incredibly desirable application [1]. But with the maturity of automotive electronics technology, the mechanical linkages are replaced by electronic control units (ECUs) [2]. ECUs are inter-connected to exchange various vehicle information via in-vehicle networks (IVNs), including controller area networks (CAN), local interconnected networks (LIN), and FlexRay [3]. Among these, CAN as a message broadcast system has become a de facto standard communication bus in IVNs to provide a better efficient, stable, and economical communication environment [4].

At present, CAN is opening more external interfaces in order to satisfy increasing requirements for intelligent in-vehicle services [5]. It also makes IVNs transfer from closure to open connectivity environment straightway [6]. Although it is an inevitable trend in the automotive ecosystem development, a massive of security threats are presented to IVNs, because CAN was not designed with any safety protection mechanisms in mind at the outset [7]. Recent researches have proven that security threats to IVNs from malicious attacks have severely disrupted normal communication [8]. It affects not only individual automobiles, but the entire vehicular network environment as a whole, including road infrastructure, other vehicles, and pedestrians [9]. Hence, cybersecurity countermeasures for IVNs are urgently required.

Based on existing studies, message authentication code (MAC) or digital signature is not a straightforward proposition owing to CAN having only a short data-load that 8 bytes and limited bandwidth [10]. Moreover, extra hardware and reconstruction protocol are necessary steps to encrypt CAN messages, which are

* Corresponding author at: School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang 212013, China.
E-mail addresses: pengzhouchengai@gmail.com (P. Cheng), Hanmuktz888@gmail.com (M. Han), lgshen@sjtu.edu.cn (G. Liu).

https://doi.org/10.1016/j.future.2022.10.020
0167-739X/© 2022 Elsevier B.V. All rights reserved.

hardly adapted to IVNs in terms of applicability and implementation cost [6]. In contrast, various Intrusion Detection Systems (IDSs) present some advantages in terms of guaranteeing IVNs secure communication [11,12]. However, the signature-based IDSs are under the limitation to just identifying pre-definition attacks, the fingerprint-based IDSs are strongly influenced by the signal characteristics in the physical layer of the vehicle, and the frequency-based IDSs have a drawback that cannot detect the non-periodical attacks [2]. Hence, an IDS that can detect various attacks independently of the vehicle environment should be explored.

Many IDSs based on machine learning (ML) and deep learning (DL) have recently been proposed owing to being validated in automotive environments with limited resources [2,13,14]. They still have the advantage of learning the normal behavior of CAN messages to distinct irregularities and legitimate traffic. However, many studies are presented based on supervised learning to achieve a two-class classification of messages, which is only valid for known attacks. In addition, feature modeling of high-dimensional messages is heavily influenced by automotive computing performance, which cannot represent the valuable information for in-vehicle streaming data accurately.

Therefore, the objective of this paper is to provide an anomaly detection method that can reduce data complexity to construct spatial–temporal features and then detect various attacks as many as possible. To satisfy these requirements, we propose an anomaly detecting system based on sparse regularization convolutional auto-encoder (SRCAE) and stream clustering model. Unlike the previous clustering methods, the significant idea of this work is to make the detection model learn the parametric nonlinear mapping relationship from the CAN messages matrix to the spatial–temporal feature with low-dimensional. Further, the stochastic gradient descent algorithm (SGD) is utilized to learn the mapping by back-propagation of the clustering target, with the mapping being parameterized by a deep neural network. We call this clustering algorithm deep evolving stream clustering (DESC).

The main contributions are summarized as follows:

(1) We proposed DESC-IDS, a novel intrusion detection architecture based on improved auto-encoder and stream clustering, which can capture sufficient spatial–temporal features from the multivariate stream data, and then low dimensional stream features are fed into the clustering module to complete anomaly detection.
(2) We designed an auto-encoder based on sparse regularization convolutional neural network, called SRCAE. It consists of temporal convolutional network (TCN) and three significant constraint terms, which can extract enough valuable low dimensional stream features for clustering module.
(3) We improved the clustering layer based on Denstream. The stream clustering model only maintains p-micro-clusters to learn normal pattern of CAN messages during the learning phase, while the detection phase only needs to determine whether the stream data can be fused to the nearest p-micro-cluster to distinguish anomalies, which reduces computational complexity and inference time.
(4) After the proposed model is optimized by auto hyper-parameters selected algorithm, we evaluated the performance and overall efficiency on the ORNL and HCRL intrusion dataset, and discusses its feasibility in real-world IoV devices. Moreover, evaluated results show the model has better generalization on mixed attack and reconstruction unknown attack with different ratio. Compare the proposed model with other baseline models, our work achieves higher detection performance on for both known attacks (96.44% vs. 98.80%), mixed attacks (94.18% vs. 86.63%), and unknown attacks (98.43%).

The rest of this paper is organized as follows: Section 2 reviews the research progress of in-vehicle intrusion detection. Section 3 presents the preparatory knowledge about IVNs. In Section 4, the security of IVNs including attack models are presented. Section 5 introduces all the algorithms about DESC-IDS in detail. Section 6 presents and discusses the experimental results. Finally, we conclude this study.

2. Related works

This section provides an in-depth overview of research works on vehicle-based intrusion detection, which can be broadly categorized into signature-based, statistical, physical feature-based, and machine learning-based approaches. Their characteristics are summarized in Table 1.

The main purpose of the signature-based IDS is to detect deviant messages through a list of predefined attack signatures. Studnia et al. [15] proposed a list of signatures derived from a CAN data set to implement IDS, but the length of the CAN bus words may not be known in advance, thus the benefit is limited. In contrast, Olufowobi et al. [12] presented a real-time specification-based IDS. The method detects anomalies without using the predefined specifications by extracting the timing model of CAN messages in real-time, but it performed poorly in the real attack dataset. Moreover, Dagan et al. [16] proposed an anti-spoofing system that can detect counterfeit messages from the illegal ECU based on arbitration bits. When the system detects malicious messages, it will send an interrupt pulse to the CAN bus to override the spoofed message. However, high communication delays and false positives are the primary drawbacks.

Statistics-based methods generally utilize statistics that can be obtained from CAN messages at the network layer. Gimden et al. [17] presented a method that monitor CAN message frequency. While some common injection attacks are detected precisely through their scheme, detection performance is poor on slow frequency attacks of the spoofing type. Song et al. [18] presented a lightweight study that injection attacks can be quickly detected by a temporal interval of messages. However, it has been proven invalid if the attacker launches injection attacks with legitimate or non-periodic frequency. A further work was presented for the timing interval in the frequency of CAN messages domain by Young et al. [19]. Their study was demonstrated that proposed frequency-based IDS has high detection accuracy in various driving models including key-on, shifting to reverse, acceleration and maintaining speed. In general, statistics-based IDS presents efficient and lightweight features, but may neglect low-volume attacks in that adversaries inject anomaly frames with lower than trigger thresholds.

Compared with the specification and statistics methods, the physical feature-based approaches work mainly on physical layer of the CAN bus in order to build up a profile of signals and voltages. Foruhandeh et al. [20] introduced a real-time intrusion detection and identification system through exploiting physical layer features of ECUs, called SIMPLE. This method not only permitted attacks to be detected based on the single frame, but also to be effectively nullified. A further study was the EASI method proposed by Kneib et al. [21]. This scheme can achieve high accuracy with a small number of samples. However, the required sampling rate depended on the length of the payload so updating the sampling rate constantly was a necessary requirement. In addition, Schell et al. [22] proposed a lightweight voltage-based IDS with lower resource requirements. By utilizing the individual voltage levels on the network during communication, the method can detect unauthorized message transmissions without any complex sampling methods and feature calculations. Most

Table 1
Comparison of recent intrusion detection methods for IVNs. It is worth noting that the absence of this feature in the method means that it is not available or not mentioned.

Type	Paper	Feature engineering	Known attack detection	Unknown attack detection	Model optimization	Lower model complexity	Real-time detection	Unsupervised learning
signature-based IDS	Studnia *et al.* [15]		✓			✓		
	Olufowobi *et al.* [12]		✓			✓	✓	
	Dagan *et al.* [16]		✓			✓		
Statistics-based IDS	Gimden *et al.* [17]		✓			✓	✓	
	Song *et al.* [18]		✓			✓	✓	
	Young *et al.* [19]		✓			✓	✓	
physical feature -based IDS	Foruhandeh *et al.* [20]	✓	✓			✓	✓	
	Kneib *et al.* [21]	✓	✓			✓	✓	
	Schell *et al.* [22]		✓			✓	✓	
ML-based IDS	Song *et al.* [2]	✓	✓		✓		✓	
	Tariq *et al.* [23]	✓	✓	✓			✓	
	Zhang *et al.* [24]	✓	✓	✓				
	Taylor *et al.* [25]	✓	✓					
	Amarbayasgalan et al. [26]	✓	✓					
	Hanselmann *et al.* [27]	✓	✓					
	Barletta *et al.* [28]	✓	✓	✓				
	ours	✓	✓	✓	✓	✓	✓	✓

importantly, voltage-based IDS is a potential source of unexpected false alarms, as voltage variations can be influenced by sources of electromagnetic interference.

In recent years, machine learning-based methods have been intensively proposed, whose applicability in IVNs has also been discussed. On the one hand, many supervised learning-based methods are reliant on large amounts of fully labeled data when training models. Song et al. [2] presented a deep convolutional neural network model for in-vehicle IDS. Their method not only reduced the false positive rate but also conducted a reconfiguration of the inception-resent network parameters, which greatly dwindled the network fitting complexity. Tariq et al. [23] proposed a LSTM-based intrusion detection method, named CANtransfer. This method showed improved detection performance through mining valuable spatial–temporal features. Also, they achieved better performance on unknown attacks based on transfer learning, but the overall detection performance was lower than previous methods. Zhang et al. [24] developed an in-vehicle IDS based on deep learning. They presented how to detect unknown attacks adaptively, but the real-time nature and complexity of the network parameters are not well addressed. Whereas it is relatively easy to capture normal CAN messages from vehicles, it is very difficult to capture deviation messages, including types of common and abrupt. Therefore, data with labels is not sufficient, which restricts the learning capability of supervised learning-based models.

To eliminate this challenge, many unsupervised-based and semi-supervised-based methods are presented. Taylor et al. [25] proposed the one-class support vector machines (OCSVM) based anomaly detection that identified deviations of message frequencies. Although getting rid of the partial label dependency, the method was only effective for frequency anomaly attacks and had a high false alarm rate. Further, Amarbayasgalan et al.. [26] introduced an IDS using deep auto-encoder with density based clustering. This method calculated compressed data and error threshold from a deep auto-encoder and then sent the results to a density-based cluster, whose detection performance yet was relatively low and relied on more parameters. To consider the temporal features, Hanselmann et al. [27] presented a LSTM-based auto-encoder IDS, which was evaluated on real and synthetic CAN data as well as showing a significant performance with the previous method. However, the method did not perform comprehensive evaluation results on real attack datasets. Furthermore, Barletta et al. [28] proposed a novel model based on kohonen self-organizing map (SOM) network and k-means clustering algorithm. They implemented anomaly detection in the clustering layer using the distance between input vectors and neurons, but describing stream-based CAN data is not an advantage of k-means.

Although intrusion detection has made a significant contribution to securing intelligent connected vehicles (ICVs), an in-depth review reveals that there are still many security challenges, especially on how to effectively learn CAN message features on unsupervised-based IDS [27,29]. Moreover, Seo et al. [30] introduced unknown attacks definition through the generative adversarial network. The method extended known attacks to unknown attacks for the first time, demonstrating the generalization capability of the model. In contrast, Song et al. [5] presented a novel self-supervised method for IVNs anomaly detection using noised pseudo normal data. The method demonstrated that unknown attacks constructed from normal data can affect the detection performance of existing models. Hence, this paper reconsiders the unsupervised learning-based model in terms of detection performance and generalization ability, so that designs a deep evolving stream algorithm, where data pre-processing, model design, and evaluation all pay more attention to these limitations.

3. In vehicle networks

In order to achieve intelligent mobility, traditional vehicles opened up more interfaces and adopt more electronic devices such as GPS, Bluetooth, and more sensors [31]. Fig. 1 shows how intelligent mobility is currently implemented. It is clear that more vehicles are no longer independent individuals, but seem to have established connections, including V2P, V2I, and V2V. At present, the IVNs have shown many advantages, for instance, V2V can provide spacing protection to reduce the possibility of vehicle collisions, and also perform more accurate assisted driving to improve the efficiency of traffic operations [32].

In order to realize these intelligent services, the vehicle communication plays a crucial role in IVNs. In other words, the vehicle communication network plays an indispensable role in processing these transmission messages in real-time and also makes the corresponding ECUs receive them correctly, and then update the vehicle status. Apparently, communication security will most likely prevent the rapid development of these intelligent requirements.

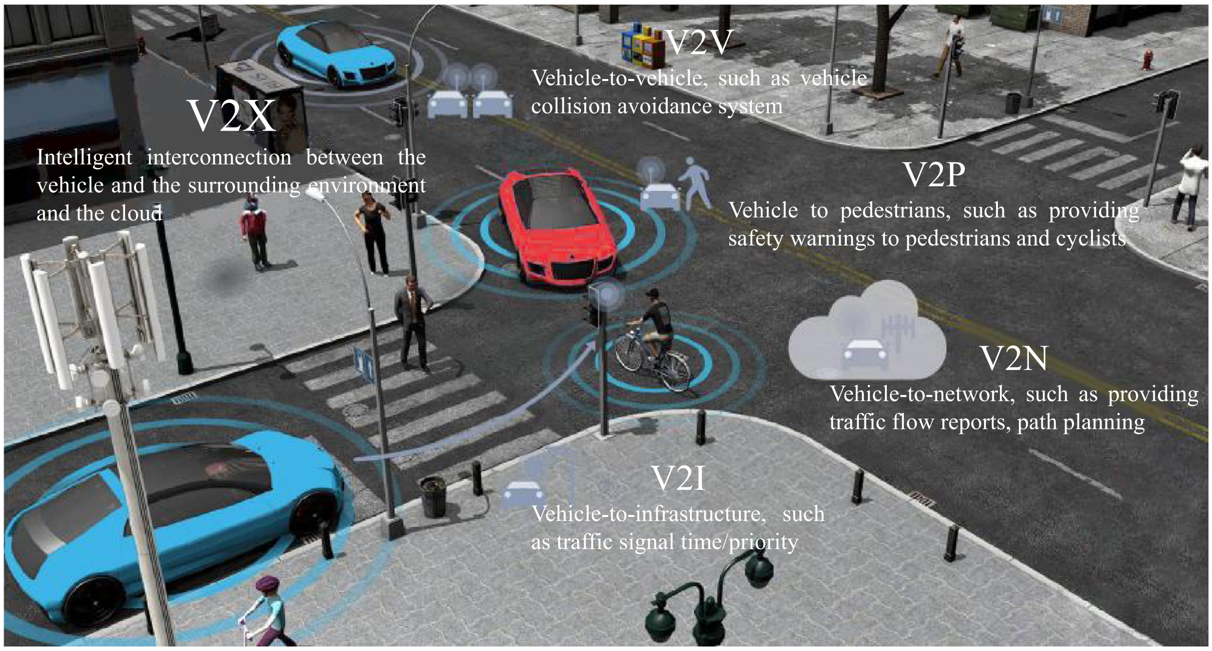

V2X
Intelligent interconnection between the vehicle and the surrounding environment and the cloud
V2V
Vehicle-to-vehicle, such as vehicle collision avoidance system
V2P
Vehicle to pedestrians, such as providing safety warnings to pedestrians and cyclists
V2N
Vehicle-to-network, such as providing traffic flow reports, path planning
V2I
Vehicle-to-infrastructure, such as traffic signal time/priority

Fig. 1. Basic architecture of communication environment for the internet of vehicles under intelligent transportation.

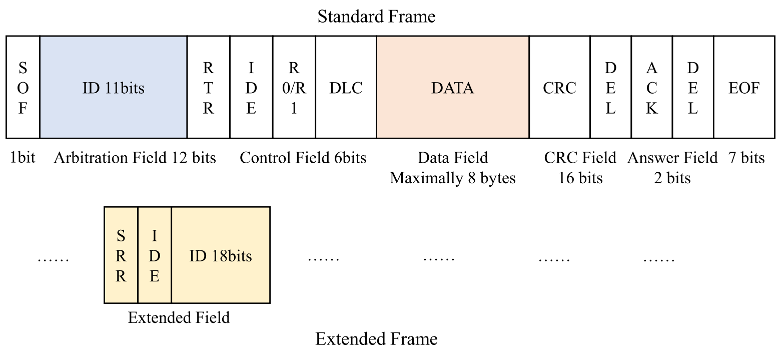


Fig. 2. CAN data frame formats, including standard and extended frames.

3.1. Controller area network

The CAN bus, undertaking tough communication tasks, was first developed by Bosch in 1985, in order to cope with the cost, complexity, and weight of in-vehicle wiring [3]. The creation of CAN ensured high stability and efficiency of in-vehicle communication and was considered the de-facto standard for modern vehicle in-vehicle communication systems [1]. Unlike the well-known TCP/IP network communication on the Internet, CAN is a frame message-based broadcast communication, where each ECU node connected to it sends data in the predefined data frames [28].

As shown in Fig. 2, message frames are available in standard and extended frames, which only difference is the arbitration field. The extended frame is designed to address the current problem that the increasingly ECU nodes generate more types of communication messages, which will require more arbitration field (CAN ID) bits. For the CAN 2.0 A device, it applied the standard frame with 11 bits of CAN ID. In contrast, the CAN 2.0B devices adopt two kinds of frame formats simultaneously, where the extended frame has 29 bits of CAN ID. Moreover, other fields of CAN frame consist of (1) start of frame (1 bit); (2) control field (6 bits); (3) data field (Maximally 8 bytes); (4) cyclic redundancy code (CRC) field; (5) acknowledge (ACK) field; and (6) end of frame.

In the CAN bus, the message frames are transmitted via the CAN ID. The ECU can receive the corresponding frame to execute the command through CAN ID while filtering frames that are not of interest. Further, the CAN ID also determines the priority at which nodes can send messages. This means that the node with a smaller CAN ID value can able to send the message, while the other nodes enter the receiving state [33]. It is noticeable that each CAN ID has a specific time interval (TI), which broadcasts vehicle status information to the network, such as steering angle and current speed, to retain the vehicle status consistent [34].

However, there is no unified industry standard for the transmission details of CAN frames currently, especially for passenger vehicle areas [35]. The concomitant problem is the inability to make defense mechanisms based on traditional methods and strike a balance between precision and universal defenses. The precise defense requires access to semantic information defined by the vehicle company, while the universal defense may enhance detection universality, but security is not always guaranteed. Hence, it is one of the main reasons why security researchers have studied intrusion detection through the ML approach. It does not need to understand the specific meaning of the transmission content but can enhance the detection accuracy of malicious attacks.

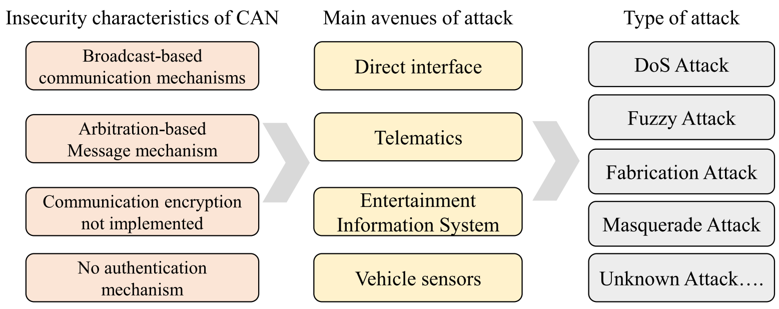

Insecurity characteristics of CAN
Broadcast-based communication mechanisms
Arbitration-based Message mechanism
Communication encryption not implemented
No authentication mechanism
Main avenues of attack
Direct interface
Telematics
Entertainment Information System
Vehicle sensors
Type of attack
DoS Attack
Fuzzy Attack
Fabrication Attack
Masquerade Attack
Unknown Attack….

Fig. 3. Illustrate the reasons for the current insecurity of CAN, the ways in which it can be attacked, and the various means of attack.

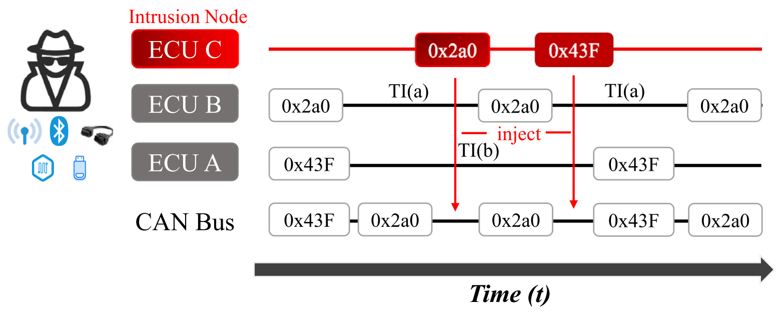


Fig. 4. Conceptual diagram of CAN message transmission mechanism and fabrication attack.

4. In-vehicle network security

With the ever-increasing number of vehicle network connections between internal and external, the CAN insecurity problems are growing in magnitude. Forensic studies have found that the CAN bus is insecure owing to its arbitration message mechanism, broadcast delivery mechanism, lack of authentication, and unencrypted communication [36]. It is the lack of sufficiently secure communications that allows attackers to compromise the in-vehicle network in a variety of ways, such as through direct interface, telematics and infotainment system, and various sensors [5].

The direct interface attack is conducted via the On-Board Diagnostics (OBD)-II port. There are security incidents, in spite of the low success rate of attacks. Similar attack pathways can be found in entertainment information processing and sensors as well (e.g., CDs, USB sticks, temperature or humidity sensors). The most dangerous is the remote attack via various wireless communication components (e.g., Bluetooth, Wi-Fi), which is by far the most difficult to defend against. Recent automotive security incidents have confirmed that cyber-attacks such as masquerade attacks, DoS attacks, Fuzzy attacks, even some unknown attacks have put vehicles in an unsafe environment, as shown in Fig. 3.

Since all nodes connected to the same CAN bus can broadcast and receive all messages, the adversary potentially sniffs communication messages or inserts masquerade messages to induce ECU failures or influence the vehicle's status. Fig. 4 shows a method of how an intruder injects masquerade messages to compromise the ECU communication and then takes control of the entire in-vehicle network. We observed that ECU_A and ECU_B send message frames $0x2a0$, $0x43F$ at time intervals $TI(a)$ and $TI(B)$ respectively. These data frames are transmitted in an orderly manner over the CAN bus by means of the arbitration mechanism. When two ECU nodes are transmitting simultaneously, the node (ECU_B) with the higher priority can get the bus control authority due to the lower CAN ID. Unfortunately, the lack of security mechanisms makes it easy for an attacker to compromise the vehicle network and take illegal control of the ECU nodes. Once adversaries have endangered an existing node (ECU_C) to access the CAN bus, they can sniff and send data frames on the bus. Horrifyingly, other nodes and the whole network are infected and even paralyzed, when they inject malicious messages.

4.1. Attack model

In this study, an assumption is determined that the attacker already has access to the vehicular network in some ways. In this situation, some kinds of cyber-attacks are conducted as follows:

(1) Attacks on confidentiality: Since CAN protocol is broadcast communication in plain text, attackers are able to execute attacks on the confidentiality of vehicle communication. They eavesdrop on all of the vehicle's driving critical information transmitted by the CAN bus. If attackers have used reverse engineering techniques to analyze which are significant transmitted information, they execute a specific spoofing attack (e.g., engine RPM and drive gear).

(2) Attacks on availability: When an in-vehicle command needs to be transmitted, the CAN performs an arbitration mechanism in order to avoid conflicts. The bus determines which message to send immediately based on its CAN ID priority. This arbitration mechanism in turn aids a vehicle available attack by an attacker, in which denial-of-service attack (DoS) is common, i.e., injecting a high priority CAN message to prevent normal transmission between nodes.

(3) Attacks on integrity: An important security problem currently confronting the CAN bus is the integrity of transmitting data, namely tampering with transmitted data. An attacker can inject tampered data to overwrite the original data according to the CAN specification of the target vehicle so that other ECUs receive the data and perform the operations as the attacker wishes. For instance, if an attacker wishes to transform the gear that is actually in a neutral state into a driving state, he must send continually altered messages with the CAN ID that provides drive gear data, since the original ECU still broadcasts drive gear data regularly.

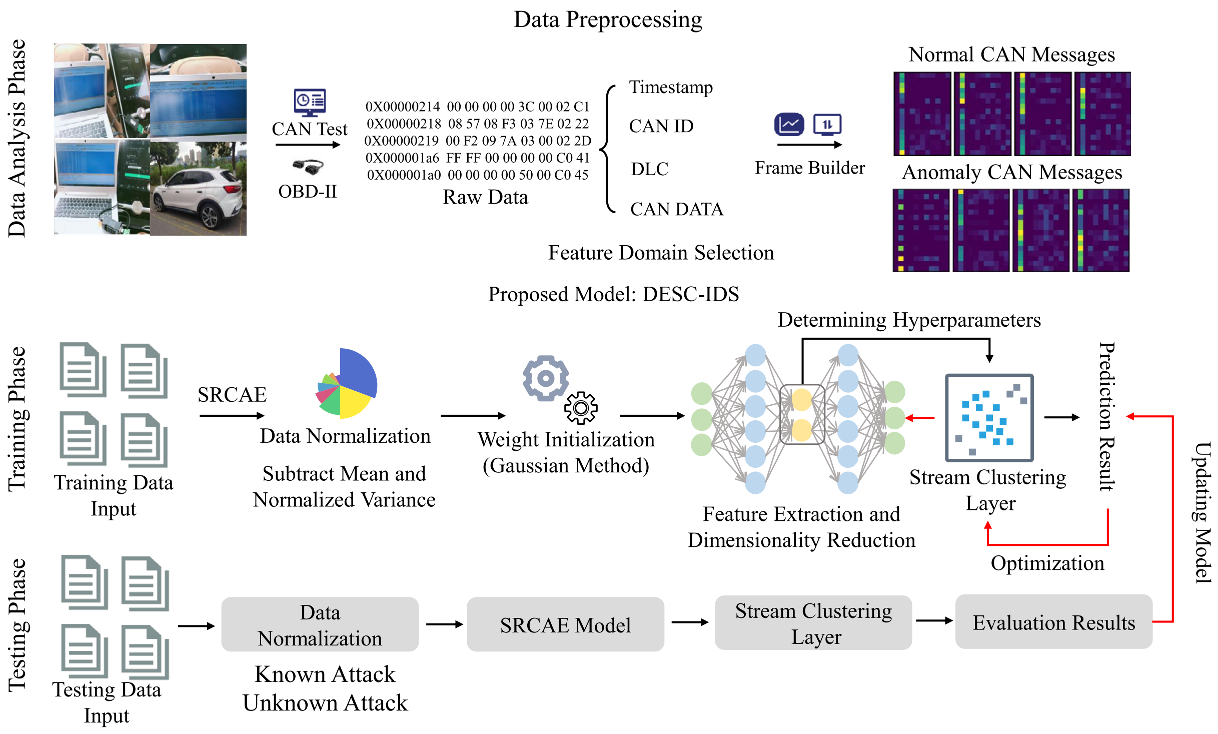


Fig. 5. An overview of the proposed framework.

Table 2
Class label and size of car hacking for intrusion detection dataset.

Class label	Original number of samples	Number of training set samples	Number of test set samples
Normal	14,037,293	9,826,105	4,211,188
DoS	587,521	411,265	176,256
Fuzzy	491,847	344,293	147,554
RPM Spoofing	654,897	458,428	196,469
Gear Spoofing	597,252	418,076	179,176

(4) Attacks on fuzzing: It is a method of discovering software vulnerabilities by providing unintended inputs to a target system and monitoring for unusual results. The attacker typically injects messages of spoofed random CAN ID and DATA values.

5. Our proposed model:DESC-IDS

Based on the in-detail analysis of some insecurity characteristics and confronting attacks for CAN, we design an IDS based on deep evolving stream clustering, consisting of the data preprocessing module, the feature extraction, and the stream clustering module. The model design is shown in Fig. 5. In this study, public datasets preprocessing, feature selection, the frame builder algorithm, and time cost testing platform construction, are precisely designed to optimize and evaluate the proposed model.

5.1. Data preprocessing

The data preprocessing module is a fundamental part of building an IDS based on deep learning. For the publicly available dataset, the car hacking for intrusion detection dataset from the hacking and countermeasure research lab (HCRL) was first used [30]. The dataset was generated by logging CAN packets via the OBD-II port of a vehicle when CAN is injected in different attacks. In the dataset, the features of each message include a timestamp, CAN ID, data length code (DLC), and the 8-byte data field (DATA[0]-DATA[7]). Moreover, the message ends with the two-class label. Since the feature "timestamp" is strongly correlated with the simulation period of a cyber-attack, it may deviate from the model learning and prediction results, so "Time Diff" is substituted for this feature. The "Time Diff" feature is the difference between two consecutive timestamps, formally known as an inter-packet delay. In addition, the Data domain is subdivided into eight feature domains, where each domain contains two hexadecimal values. If there are missing bytes in the data field, the value "00" is assigned to fill. Also, all hexadecimal values in the ID and Data domain are transformed to decimal values. Table 2 presents the attack types and size of this dataset, with samples for the training and test sets also provided. Eventually, the 11 attributes in the CAN frame are taken as input to the model.

In order to validate the model generalization capability, the model was then evaluated for performance on the Real ORNL Automotive Dynamometer (ROAD) CAN intrusion datasets [37]. The datasets are the most available and representative datasets currently as it includes more comprehensive data on attacks such as fuzzing attacks, targeted ID fabrication and masquerade attacks, and accelerator attacks. In particular, these attacks were executed on a variety of scenarios and target IDs, which were more significant in assessing the detection performance of the model compared to the former. In this paper, the preprocessing operation of the datasets are completed using the same feature selection and transformation method. Although the proposed model is unsupervised learning, in order to evaluate model performance, this study obtains the labeled through its

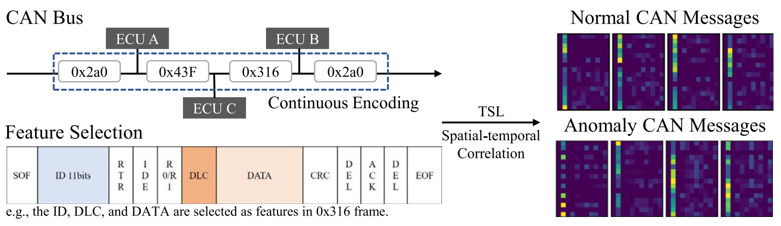


Fig. 6. The left figure illustrates the feature selection and continuous encoding process of CAN frame, while the right figure shows the CAN 2-D data frames.

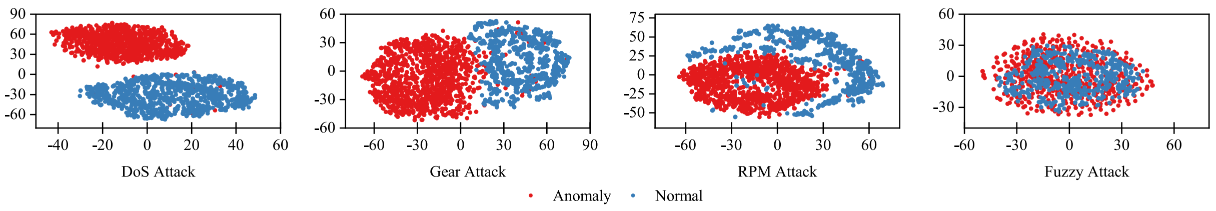


Fig. 7. TSNE-based visualization of four attack datasets for HCRL.

Table 3
Class label and size of the Real ORNL Automotive Dynamometer (ROAD) CAN intrusion dataset.

Class Label	Messages	# Logs
Correlated Signal Fabrication Attack	192,748	3
Correlated Signal Masquerade Attack	187,258	3
Max Engine Coolant Temp Fabrication Attack	61,923	1
Max Engine Coolant Temp Masquerade Attack	61,881	1
Max Speedometer Fabrication Attack	572,842	3
Max Speedometer Masquerade Attack	561,151	3
Reverse Light Off Fabrication Attack	308,385	3
Reverse Light Off Masquerade Attack	302,908	3
Reverse Light On Fabrication Attack	465,234	3
Reverse Light On Masquerade Attack	457,203	3
Fuzzy Attack	94,931	3

injection rules. For instance, the fuzzy attack injected frames with random IDs (cycling from 0 × 000 to 0 × 255 in sequence) with the maximum payload (0 × FFFFFFFFFFFFFFFF) into the CAN bus every 0.005 s. This typically changes many physical states of the vehicle, including the accelerator failing, dashboard lights and headlights coming on, and seat position shifting. Moreover, the original fabrication attacks are malicious modifications on load field values of specific target ID, while the masquerade attacks are captured using a fabrication attack, i.e. the legitimate target ID frame is removed before each injected frame, which makes frequency-based intrusion detection models ineffective. Table 3 shows 11 attack types, 29 attack captures and its size selected from the raw datasets. All selected datasets are divided into training set and test set.

5.2. Frame builder

In general, convolutional neural network (CNN)-based auto encoder is designed to extract valuable feature and reduce dimensional from the input with grid format. Hence, we processed the preprocessing datasets as a spatial–temporal correlation using the frame builder that transforms continuously CAN message data into 2-D data frames according to time-series length (TSL), as illustrated in Fig. 6. In short, the frame builder extracts the 11 feature from each frame and build a width = 11, height = TSL message matrix by stacking the most nearest frames. The major advantage of such an encoding is that it prevents efficiency bottlenecks in DL-based IDS, and thus matches the requirements of the CAN bus to detect thousands of messages per second. Moreover, each selected feature domain is representative of temporal features, so that is represented as:

$$T\left(f_i\right)=\left\{f_i\left(t_1\right),f_i\left(t_2\right),f_i\left(t_3\right),\ldots,f_i\left(t_n\right)\right\}, \tag{1}$$

where $f_i(t_j)$ represent that the ith feature of the jth time step of the 2-D data frames $T(f_i)$. Hence, $T(f_i)$ fact with temporal characteristics, and all feature domains are given as:

$$T\left(f_{1,\ldots,n}\right)=\left\{T\left(f_1\right),T\left(f_2\right),T\left(f_3\right),\ldots,T\left(f_n\right)\right\}, \tag{2}$$

where $T(f_{1,\ldots,n})$ is the 2-D data frames with a custom sequence length. The custom length in this study is 128, i.e., the input size for DESC-IDS is 128 × 11 × 1. Furthermore, to balance comparability between features with a large and small initial range, 2-D data frames utilized z-score normalization to normalize all features, calculated as follows:

$$x'=\frac{x-\mu}{\sigma}, \tag{3}$$

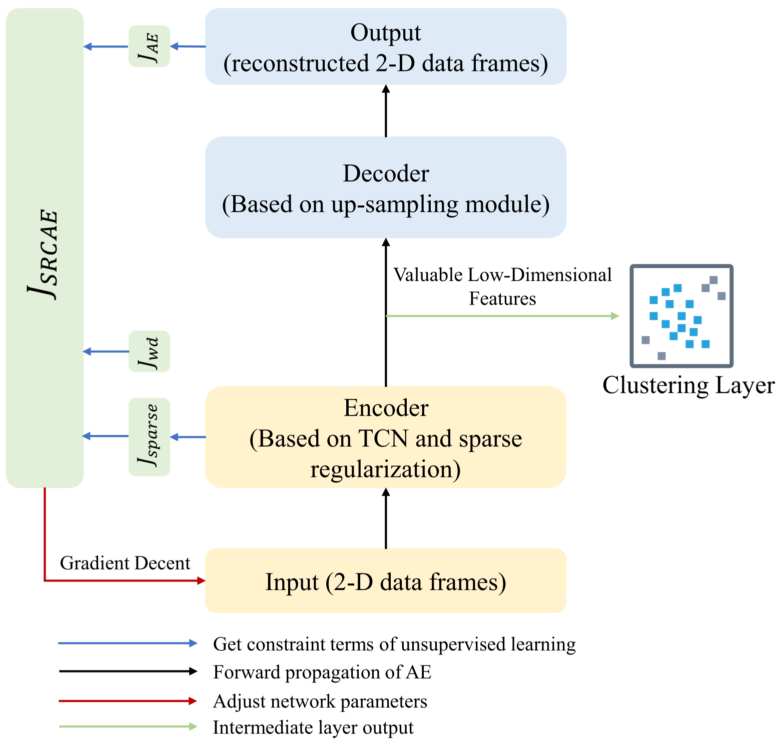
where x is raw data, x' represents normalized data. Also, μ and σ represent the mean value and standard deviation of original data, respectively.

In particular, we verified the feasibility of 2-D data frames suitable for clustering on the data processed from the frame builder, using the t-distribution and stochastic neighbor embedding module (TSNE), as shown in Fig. 7. Interestingly, the first three attack datasets can be effectively classified as normal messages and attack messages, while fuzzy attacks are more difficult to distinguish due to randomness injection. Thus, the prerequisites for deep evolutionary stream clustering are available, but representative features need to be obtained in the deep feature extraction module in order to assist the stream clustering model to improve detection performance.

5.3. Proposed DESC-IDS

5.3.1. Feature extraction module-SRCAE

A basic auto-encoder (AE) consists of an encoding part and an decoding part [38], which integrates the advantages of dimensionality reduction and feature extraction. On the one hand, the complexity of the data is reduced by applying the unsupervised learning-based encoding-decoding process to unlabeled samples; on the other hand, the valuable features of the samples are preserved through the powerful feature extraction module, which improves the classification accuracy of the model. In this section,



J_{SRCAE}
J_{AE}
J_{wd}
J_{sparse}
Output
(reconstructed 2-D data frames)
Decoder
(Based on up-sampling module)
Valuable Low-Dimensional Features
Clustering Layer
Encoder
(Based on TCN and sparse regularization)
Gradient Decent
Input (2-D data frames)
Get constraint terms of unsupervised learning
Forward propagation of AE
Adjust network parameters
Intermediate layer output

Fig. 8. The framework of SRCAE module.

we propose an AE with the ability of extraction spatial–temporal feature using sparse regularization and temporal convolutional network (TCN), called SRCAE. Fig. 8 shows the structure of SRCAE.

In such a structure, n samples are indicated as $X = \{x_i\}_{i=1}^n$. Each sample is represented by $x_i = [x_{i1}, x_{i2}, \dots, x_{id}] \in R^d$, which are fed into encoding module, aiming to realize valuable feature mapping y_i from high dimensional to low dimensional. In other words, the encoder function $y_i = f_\theta(x_i) = s(Wy_i + b)$ maps the input sample x_i to y_i in the l- dimensional hidden layer. Also, the decoder module is aim to reconstruct the input data, i.e., the decoder function $z_i = g_\theta(x_i) = s(W'y_i + b')$ is utilized to recover x_i from z_i. Thus, the steps of SCRAE module are succinctly given as:

$$a_1 = Wx_i + b, y_i = s(a_1), \tag{4}$$

$$a_2 = W'y_i + b', z_i = s(a_2), \tag{5}$$

where W and b denote the weights and bias in the encoding module, while W' and b' represent the weights and bias in the decoder module. Hence, a_1 and a_2 are expressed its output. To make the output features realize non-linear fitting, $s(x)$ is introduced as activation function, i.e., Leakey-ReLU function, calculated as follows:

$$y = max(0, x) + leak \times min(0, x), \tag{6}$$

where the leak is a very minor constant, which preserves some of the negative axis values so that not all information on the negative axis is lost. In addition, $X = [x_1, x_2, \dots, x_n] \in R^{d\times n}$, $Y = [y_1, y_2, \dots, y_n] \in R^{l\times n}$, $Z = [z_1, z_2, \dots, z_n] \in R^{d\times n}$ are represented the sample feature matrices of the input layer, intermediate layer, and output layer. It is important to note that valuable intermediate layer features y_i are deployed in the clustering layer.

Importantly, three constraint term, i.e., the weight decay term J_{wd}, the sparse regularization term J_{sparse}, and the average reconstruction error term J_{AE} are taken into account in this structure to improve the representation performance. Firstly, J_{wd} could reduce the magnitude of weight parameters and prevent the weight from over-fitting. Thus, J_{wd} is defined as the sum of the weights from the entire intermediate layer, calculated as follows:

$$J_{wd} = (1/2)\sum_{i=1}^{d}\sum_{j=1}^{1}\left(W_{ji}\right)^2. \tag{7}$$

Further, it is well known that direct access to the necessary semantic features is difficult, especially for 2-D data frames. In this study, J_{sparse} is implemented as an unsupervised learning strategy to restrict the hidden layers and make them sparsely expressed. When the SRCAE model is propagated forward, the model needs to calculate the activation probabilities for each neuron. Subsequently, it activates some neurons and suppresses others by comparing the obtained activation probabilities with the given sparse parameters. Note that a certain scale of neurons in the structure are initialized. The mean activation probability ρ_j^* is calculated as follows:

$$\rho_j^* = (1/n)\sum_{i=1}^{n} a_j(x_i), \tag{8}$$

where n is the number of samples, $a_j(x_i)$ is expressed as the activation value on the jth neuron, and the ith sample x_i. In order to measure the similarity between the mean activation probability ρ_j^* and the sparse parameter ρ, the Kullback–Leibler (KL) divergence is used in this structure, calculated as follows:

$$\begin{aligned} J_{\text{sparse}} &= \sum_{j=1}^{l} KL(\rho \parallel \rho_j^*) \\ &= \sum_{j=1}^{l} \rho \log(\frac{\rho}{\rho_j^*}) + (1-\rho)\log(\frac{(1-\rho)}{(1-\rho_j^*)}). \end{aligned} \tag{9}$$

Finally, J_{AE} is the underlying cost function of AE, which represents the average reconstruction error between the input and

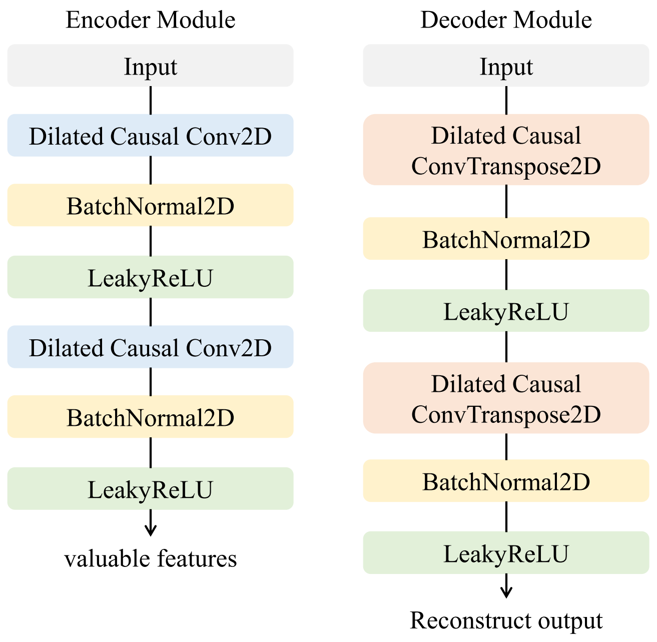


Fig. 9. Illustrate the SCRAE module in detail.

output data for all samples. A lower J_{AE} guarantees approximate equivalence between the input and output data, allowing perfect reconstruction of the input data, which is calculated as follows:

$$\begin{aligned}\min_{\theta} J_{AE}(\theta) &= \min_{\theta} \sum_{i=1}^{n} \|x_i - \widetilde{x}_i\|^2 \\ &= \min_{\theta} \sum_{i=1}^{n} \|x_i - D_\theta(E_\theta(x_i))\|^2,\end{aligned} \tag{10}$$

where x_i and $\widetilde{x}_i$ represent the raw input and reconstruct output, respectively. Also, D_θ and E_θ indicate the encoder and decoder function, respectively. In summary, the SRCAE is proposed as an unsupervised learning method that adjusts the weights and biases between each pair of joint layers in the training step using the three constraint terms. Subsequently, the feature representations of the hidden layers are extracted as input to the classifier in the test step to further implement the classification application. Similarly, we also recognize the similar matrices reconstructed from the normal messages as the unknown attack. Thus, the objective function of the SRCAE is finally defined as follows:

$$J_{\text{SRCAE}} = J_{AE} + \lambda J_{wd} + \beta J_{\text{sparse}}, \tag{11}$$

where λ and β are two constant coefficients to balance the term relations of J_{AE} to J_{wd} and J_{sparse}.

Fig. 9 presents detailed structural design of the SRCAE module. The encoder module consists of two convolution blocks, where each component includes dilated causal convolution layer, batch-normal layer, and leaky-ReLU. The convolutional layer is used to extract the local-spatial feature, while the batch-normal layer make the data normalization to balance every batch data. In contrast, the decoder module is used the dilated causal convolution transposition layer to realize up-sampling and reconstruct input. In particular, dilated causal convolution could extract valuable temporal features, as shown in Fig. 10.

For the first convolution component, the dilation value is set to 1 in order to reserve sufficient important. In addition, the convolution kernel is switched to 1×11, thereby capturing important features of each message by top-down. After the first convolution block is completed, the model can obtain a 1-dimensional time-series feature. Next, the second convolution block extracts temporal features based on dilation value that is set to 2. Thus, model can recall multi history time steps to build the intermediate feature with spatial–temporal correlation. Notably, the SRCAE requires only two convolution components to extract valuable features for the clustering layer, which reduces model parameters and improves detection efficiency.

5.3.2. Deep evolving stream clustering module-DESC

Massive, time-series, fast-changing, and potentially infinite, high-dimensional are all synonymous with streaming data [39]. Density-based clustering of data streams is currently making effective progress in real-time, especially for isolated detection [40]. Compared with other clustering algorithms, DESC is a stream-based clustering implementation on the improved Denstream algorithm, which can adapt damp windows for the DBSCAN algorithm.

In other words, the Denstream aggregates the incoming dense points with the help of p-micro clusters and temporarily retains the data points that do not belong to the p-micro-clusters by an aberrant buffer. This buffer holds these points until they create a dense o-micro-cluster, and eventually evolve into a p-micro-cluster. However, we consider the detection of attack message on the valuable spatial–temporal features extracted by the SR-CAE as an anomaly detection problem in this study. Thus, the Denstream is improved by removing certain functions so that the normal models generated by p-micro-clusters would not be tainted by instances classified as attack 2-D data frames. Although this improvement may cause the cluster to miss some normal instances, it guarantees the accuracy of the predictions because the streaming nature of DenStream will accommodate gradual changes in normal behavior.

Firstly, normal instances are modeled based on p-micro-clusters in DenStream, while points that are not merged with p-micro-clusters are classified as anomalies and placed in the outlier buffer until they are removed. By removing the ability to turn an o-micro-cluster into a p-micro-cluster, the improved DESC can only cluster instances that are considered normal, as shown in algorithm 1.

Specifically, three phases that initial phase, online phase, and offline phase are executed by DESC. In the initial phase, all 2-D data frames Z are extracted significant features P. The p-micro-clusters C and wights W are generated by DBSCAN through initial N points. Note that density-based DBSCAN has the advantage of not requiring a predefined number of clusters, and can form clusters of arbitrarily shaped [41]. In order to build p-micro-clusters C, two important parameters, $\mathcal{E}$ and the minimum point respectively, should be defined. The parameter $\mathcal{E}$ defines the $\mathcal{E}$-neighborhood of a point, of which all points are within a distance of the first point. More formally, we calculate an $\mathcal{E}$-neighborhood of a point p in the dataset D as:

$$N_\epsilon(p) = \{q \in D : \text{dist}(p, q) \leqslant \epsilon\}, \tag{12}$$

$$\text{dist}(p, q) = \sqrt{p^2 - q^2}, \tag{13}$$

where dist (p, q) represents the euclidean distance between the points p and q. To determine the core object, the DBSCAN needs to calculate $\mathcal{E}$-neighborhood of a single point, until there are more points in the neighborhood than the minimum point parameter. After the core object has been determined, it performs a cluster expansion of the points in the $\mathcal{E}$-neighborhood of the core object. Once a core object representing a cluster has been found, DBSCAN repeats the above steps until it completes the traversal of all points. In the online phase, DESC maintains core-micro-clusters

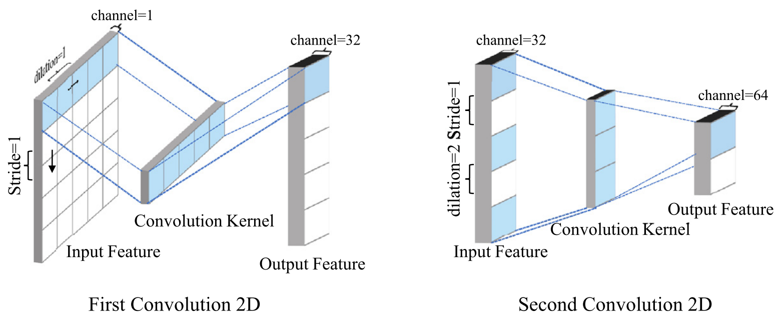


Fig. 10. The structural design elaboration for temporal feature extraction.

Algorithm 1 DESC

```
Require: 2-D data frames: $Z$, a set of stream data: $P$, p-micro-clusters: $C$, initial point number: $N$, radius threshold: $\epsilon$, p-micro-clusters weight: $W$, nearest micro cluster: $C_{nearest}$, weight threshold: $\beta\mu$, current time: $t$, decay factor: $\lambda$.
1: Initial Phase:
2: for $i = 1 \leq$ length($Z$)) do
3:     $M_i$ = SRCAE($Z_i$);
4: end for
5: $C$, $W$ = DBSCAN($P$[: $N$]);
6: Online Phase:
7: for $i = 1 \leq$ length($P$)) do
8:     Merging($P_i$, $C$, $W$, $\epsilon$);
9:     Periodic inspection algorithm ($\beta\mu$, $\lambda$, $C$, $W$, $t$);
10: end for
11: Offline Phase:
12: detection sample $p$;
13: $C_{nearest}$ = getNearestMicroCluster($p$, $C$);
14: if Merging($p$, $C$, $W$, $\epsilon$) then
15:     consider as normal;
16: else
17:     consider as anomalies;
18: end if
```

Algorithm 2 getNearestMicroCluster(P_i, C)

```
Require: current distance: $dist_c$, smallest distance: $dist_s$, nearest micro cluster: $C_{nearest}$.
1: for $i \leq$ length($C$) do
2:     dist $_c = \|C_{i-\text{ center}} - P_i\|$;
3:     if $dist_c \leq dist_s$ then
4:         $dist_s = dist_c$;
5:         $C_{nearest} = C_i$;
6:     end if
7: end for
```

based on normal point. The core-micro-clusters are compact representations of large sets of dense data weighted by a fading function, calculated as follows:

$$f(t) = 2^{-\lambda t}, \tag{14}$$

where $\lambda > 0$ is regarded as decay factor, and t is the current time-step. In addition, there are three new attributes named weight, center, and radius, which need to be defined and used in the core-micro-clusters. For each core-micro-cluster, its radius must be less than $\mathcal{E}$, and weight must be greater than another parameter μ, where μ is defined as 10. These parameters allow for the calculation on a set of close points $P_1, P_2, P_3 \dots P_n$ at time t, with time-steps $T_1, T_2, T_3 \dots T_n$ through the following formulas:

$$w = \sum_{i=1}^{n} f(t - T_i), \tag{15}$$

$$c = \sum_{i=1}^{n} \frac{f(t - T_i) P_i}{w}, \tag{16}$$

$$r = \sum_{i=1}^{n} \frac{f(t - T_i) \operatorname{dist}(p_i, c)}{w}. \tag{17}$$

As new data points arrive, p-micro-clusters need to be updated. Hence, these parameters of core-micro-clusters are redefined by the weighted linear $\overline{CF^1}$ and squared sums $\overline{CF^2}$, calculated as follows:

$$\overline{CF^1} = \sum_{i=1}^{n} f(t - T_i) P_i, \tag{18}$$

$$\overline{CF^2} = \sum_{i=1}^{n} f(t - T_i) P_i^2. \tag{19}$$

Thus, radius and center are calculated as follows:

$$c = \overline{CF^1}/_w, \tag{20}$$

$$r = \sqrt{\frac{\overline{CF^2}}{w} - \left(\frac{\overline{CF^1}}{w}\right)^2}. \tag{21}$$

Most importantly, the merging algorithm presents an update process based on stream of p-micro-clusters. In order to realize this algorithm, the nearest p-micro-cluster of the current point needs to be searched, as shown in the algorithm 2. The important step is to traverse the euclidean distance between the center of each micro-cluster and the current point. Once get the nearest p-micro-cluster, the merging algorithm determine whether the current point could be combined with it, as shown in algorithms 3.

As time rolls on, the weight of an existing p-micro-cluster decrease. If the weight falls below the threshold, the micro-cluster is not representative and cannot merge the latest stream data. Thus, a periodic check method is required to remove these "aged" p-micro-clusters, as shown in algorithm 4, where the minimum check period calculates based on a decay factor and a weighting threshold. Once the modeling of representative p-micro-clusters describing normal traffic is complete, the model is applied in the offline phase to detect traffic with attack messages, i.e. by means of a merging algorithm. This is a significant operation that boosts prediction accuracy, as DenStream's time-based stream properties will adapt to changes in normal behavior.

Algorithm 3 Merging(P_i, C, ϵ)

```
Require: nearest micro cluster: $C_{nearest}$.
1: $C_{nearest}$ = getNearestMicroCluster($P_i$, $C$);
2: if $r_c$ (the new radius of $C_{nearest}$) $\leq \epsilon$ then
3:     Merge $P_i$ into $C_{nearest}$;
4:     return True;
5: else
6:     Try to merge $P_i$ into outlier-buffer;
7:     return False;
8: end if
```

Algorithm 4 Periodic inspection algorithm ($\beta\mu$, λ, C, W, t)

```
1: Time period: $T_p = \left[\frac{1}{\lambda}\log\left(\frac{\beta\mu}{\beta\mu-1}\right)\right]$;
2: if ($t$ mod $T_p$) = 0) then
3:     for $i = 1 \leq$ length($C$) do
4:         if $W_i \leq \beta\mu$ then
5:             Delete $C_i$;
6:         end if
7:     end for
8: end if
```

5.4. Runtime complexity

In this section, we also analyzed the complexity of SRCAE. Clearly, the computational consumption occurs mainly in the multiplication of the matrix. Since convolution is considered on 2-D data frames, the number of neurons in the encoder and decoder is assumed to be k with shared characteristics. Additionally, the model input is assumed to have m samples for training and the number of neurons in the intermediate layer is n. Due to all samples in the SRCAE training step include both feed-forward and feedback processes, the runtime complexity of first computing the reconstruction error term is

$$T\left(J_{AE} + \partial J_{AE}\right) = O(2mnk + 2mnk) = O(mnk), \tag{22}$$

where the time complexities of the reconstruction error term in the feed-forward phase J_{AE} and the feedback phase ∂J_{AE} are both $O(2mnk)$. Since the feedback phase follows the feed-forward phase, the superposition of these two phases adds up to the final reconstruction error time complexity $O(mnk)$. Similarly, the time complexity of the wight decay term and sparse regularization term is calculated as follows:

$$T\left(J_{\text{sparse}} + \partial J_{\text{sparse}}\right) = O(0 + mnk) = O(mnk), \tag{23}$$

$$T\left(J_{wd} + \partial J_{wd}\right) = O(0 + mnk) = O(mnk). \tag{24}$$

It is worth noting that both constraints only adjust the weights and biases of the model in the feedback phase, and do not have the time consumption of the feed-forward phase. Hence, the time complexities and sparse regularization term and wight decay term are both $O(mnk)$. Finally, the total time complexity of SRCAE can be calculated as follows:

$$\begin{aligned} T\left(J_{\text{SRCAE}} + \partial J_{SRCAE}\right) &= O(mnk + mnk + mnk) \\ &= O(3mnk) \\ &= O(mnk). \end{aligned} \tag{25}$$

However, the nature of shared convolution kernel makes the number of neurons much smaller than the number of samples. Also, the number of neurons in the intermediate layer is low-dimension spatial–temporal features. Therefore, the time complexity can be reduced again to $O(n)$.

In the clustering layer, the time complexity includes three phase. Since the initialization of p-micro-clusters is based on the DBSCAN algorithm, so that the time complexity at N initialization points is $O(nlogn)$. In the online phase, the time complexity of merging algorithm at the n samples and m p-micro-clusters is $O(nm)$, while the time complexity of periodic inspection algorithm is $O(logn)$. Also, the time complexity of offline phase is to search for the nearest micro-cluster, i.e., $O(m)$. It is worth noting that the number of micro-clusters m is much smaller than the number of samples n, so the time complexity of the whole clustering layer is only $O(nlogn)$. Actually, the training process of the proposed DESC-IDS can be done on a server machine with high computational speed, while the testing process should be implemented in the vehicle system. Thus, the time complexity is unaffected by the DBSCAN algorithm, forming the final detection phase is only $O(n)$.

6. Experiment

To measure the performance of our in-vehicle intrusion detection system, we have done a detailed experimental evaluation.

6.1. Evaluation metrics and experiment setup

Four statistics metrics, including true positive (TP), false positive (FP), false negative (FP), and true negative (TN), are used to measure evaluation metrics. We first measured the false negative rate (FNR) and error rate (ER) to evaluate the detection performance of the model. In our anomaly detection setup, FNR represents the proportion of undetected attacks belonging to the attack 2-D data frames and ER is defined as the proportion of false detection, which are calculated as:

$$FNR = \frac{FN}{TP + FN}, \tag{26}$$

and

$$ER = \frac{FN + FP}{TP + TN + FP + FN}. \tag{27}$$

There is clearly an expectation that the lower FNR value, the less likely our anomaly detection model will destroy vehicle communication due to a small number of undetected attacks. Similarly, precision, recall, and F1-score are calculated, in order to compare with existing unsupervised algorithms. Precision represents the proportion of actual attack frames out of the detected attack frames, calculated as follows:

$$P = \text{TP}/(\text{TP} + \text{FP}). \tag{28}$$

A lower FP rate will improve the service quality of the IDS so that normal in-vehicle communication is unaffected by frequent false alarms. Recall is the proportion of all attack frames that are detected, also known as the true truth rate (TPR). It can be simply calculated by subtracting FNR from 1 or using the following formula:

$$R = TP/(TP + FN). \tag{29}$$

The F1-score is a compromise between precision and recall. When a dataset has an unequal class distribution, the F1-score is commonly used to quantify detection performance. It is calculated as:

$$F1 = 2 \times (P \times R)/(P + R). \tag{30}$$

To develop the proposed DESC-IDS, the SRCAE module is implemented using the pytorch framework in python, while the clustering layer was realized by Scikit-learn in python. The experiments setup of model training was carried out on Intel(R) Core

Table 4
DESC-IDS performance comparison with baseline methods.

DoS	ER (%)	FNR (%)	P (%)	R (%)	F1 (%)
DESC-IDS	3.74	4.38	**96.69**	95.62	**96.08**
OC-SVM [25]	30.95	0.69	55.72	99.31	71.39
DAE [26]	3.76	0.12	91.27	99.88	95.36
Canet [27]	6.50	19.2	–	80.8	–
SOMK-C [28]	**2.22**	**0.00**	90.88	**100.00**	95.22
Fuzzy	ER (%)	FNR (%)	P (%)	R (%)	F1 (%)
DESC-IDS	**4.63**	4.67	**95.25**	95.33	**95.27**
OC-SVM [25]	32.22	94.50	6.94	5.50	6.14
DAE [26]	6.13	**3.74**	90.05	**96.26**	93.05
Canet [27]	25.70	74.6	–	25.4	–
SOMK-C [28]	26.80	99.80	0.46	0.20	0.28
Gear	ER (%)	FNR (%)	P (%)	R (%)	F1 (%)
DESC-IDS	**2.11**	2.16	**97.88**	97.84	**97.86**
OC-SVM [25]	73.80	28.85	35.43	71.15	47.30
DAE [26]	11.98	18.2	94.63	81.80	87.75
Canet [27]	2.50	12.20	–	87.80	–
SOMK-C [28]	4.40	**0.00**	79.70	**100.00**	88.70
RPM	ER (%)	FNR (%)	P (%)	R (%)	F1 (%)
DESC-IDS	**3.76**	4.00	**96.41**	96.00	**96.17**
OC-SVM [25]	60.54	34.85	33.43	65.15	44.18
DAE [26]	8.40	4.27	95.73	95.73	92.10
Canet [27]	7.00	20.80	–	79.20	–
SOMK-C [28]	4.51	**0.00**	79.04	**100.00**	88.29

(TM) i7-9500U CPU@3.6GHZ, 64 GB of RAM, and GPU RTX 2080 Ti. In the test phase, the first step is that inject malicious messages with the same specification through CAN Test software, and then the performance and efficiency of the model were evaluated based on real vehicles through NVIDIA Jetson AGX Xavier (16 GB).

6.2. Hyperparameter

Hyperparameters are a critical factor in determining the performance of a model. Currently, the Automatic Hyperparameter Optimization (AHPO) algorithm is one of the important channels for dealing with the constant trial and error stages of deep learning developers [42]. In order to improve computational efficient and reduce trail times, the SRCAE module obtains optimal hyperparameters using Bayesian Optimization (BO) based on AHPO in this study. Specifically, learning rate (lr), leaky-ReLU, filters, and optimizer are the optimization purpose. Next, the validation accuracy is set to the evaluation metrics. Thereafter, we chose a sampling number of 10 for each hyperparameter set and a training number of 3. Finally, the best performance result and hyperparameters set are presented in the experiment list.

After selecting the optimal hyperparameters (i.e., lr = 10^{-3}, leaky-ReLU = 0.2, filters = 32, 64), the SRCAE model performed the training phase, as shown in Fig. 11. Intuitively, the optimal hyperparameters allow the model to dramatically improve the speed of inference, where spoofing attack of Gear has the earliest drop in loss and the lowest convergence values. Similarity, the RPM attack with the same spoofing facts shows sub-optimal performance. The training loss convergence performance of the DoS attack with high priority injection is remarkably close to that of the spoofing attack. However, the training of fuzzy attacks injected with random data forms a convergence state with maximum loss value, due to high complexity.

6.3. Detection performance

To evaluate the proposed IDS for intrusion detection performance, we compared it with four baseline methods, i.e., classification-based (OC-SVM [25]), auto encoder-based (DAE [26] and Canet [27]), and deep-clustering (SOMK-C [28]), respectively. The ER, FNR, precision, recall, F1-score of different baseline

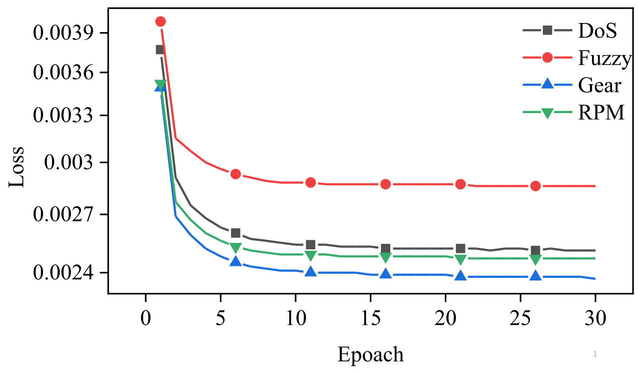


Fig. 11. Training loss on each dataset from HCRL.

methods against DESC-IDS are reported in Table 4, where the highest performance values are highlighted in bold, and '-' only represents the baseline method has not been evaluated on this metric.

In Table 4, we can observe that after multi-times testing for ER metrics on the DESC-IDS achieves excellent performance that Fuzzy (4.63%), Gear (2.11%), and RPM (3.76%), respectively. Although inferior to the SOMK-C scheme on DoS attacks, it has a clear advantage over other schemes. In addition, the SOMK-C has high error rate on Fuzzy attacks, which clearly indicates a lower generalization capability, i.e. it cannot adapt to different attacks. Similarly, the Caenet also exhibits unstable ER on Fuzzy attacks but outperforms the DAE model on spoofing attacks. The worst ER performance occurs with OC-SVM, especially on spoofing attacks, 73.80%, and 60.54%, respectively. It is well demonstrated that the semi-supervised classification-based model (OC-SVM) is not suitable for accurately detecting attacks on the CAN bus.

The experiments results on FNR presents a positive phenomenon. Compared with density-based method (DAE), the FNR of the proposed IDS has decreased from 11.23% to 3.08% on spoofing attacks, but slightly inferior to it on DoS and fuzzy attacks. Similarly, the OC-SVM and Canet both present worst performance, other than on OC-SVM for DoS attack. In contrast, the SOMK-C presents excellent results, but have no contribution

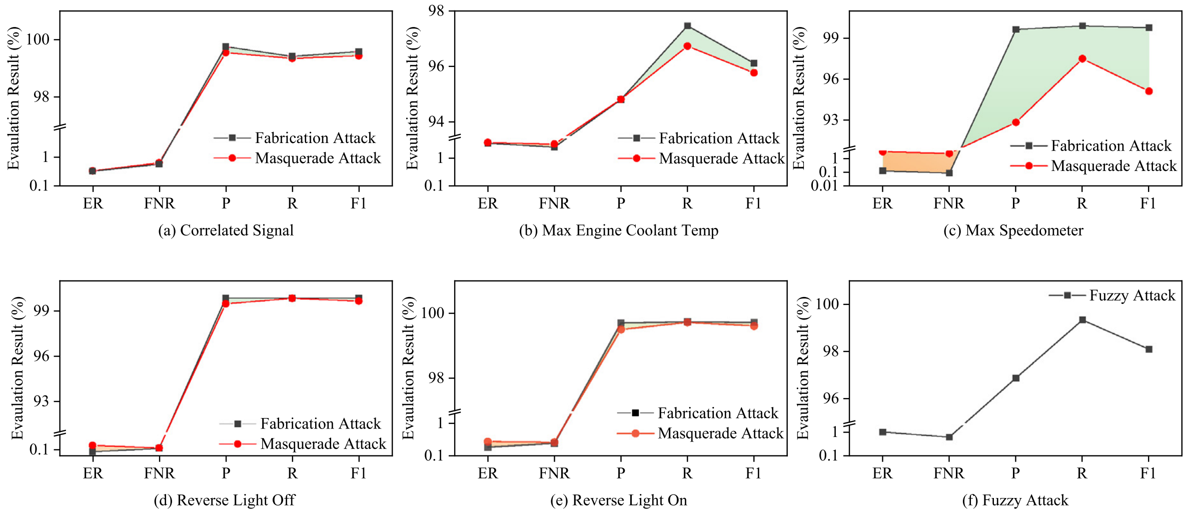


Fig. 12. Performance evaluation on each type attack of the ORNL dataset, where each type from (a) to (e) consists of fabrication attack and masquerade attack. In particular, the upper and the bottom deviation of both attacks are filled with color respectively. (f) represents the evaluation result of fuzzy attack.

on fuzzy attacks. Obviously, the proposed method could keep great performance on various attack datasets. Since the model only considers the normal pattern of traffic, it cannot take into account the fact that it achieves state-of-the-art performance on every attack. The results of FNR also reflect equivalent recall.

It is worth noting that the proposed method outperforms all the baseline models in terms of precision, achieving 96.69%, 95.25%, 97.88%, 96.41%, respectively. The average improvement of 3.95% compared to DAE is attributed to the model learning the correct morphology of the stream-based messages. Since the 2-D morphology of the fuzzy attack is mixed, the morphology of the distance-based metric message streams is not suitable. This leads to poor precision of SOMK-C, which fails especially on fuzzy attacks. Furthermore, the precision of OC-SVM is on average below 50% and can hardly be applied to intrusion detection.

As a result of the high precision, and good recall, our model exhibits a stable F1-score, achieving 96.08%, 95.27%, 97.86%, and 96.17% on the four attack datasets respectively. Low accuracy performance of the DAE and SOMK-C models responds to lower F1 scores, while SOMK-C and OC-SVM have little ability to detect fuzzy attacks.

6.4. Generalization performance

In this subsection, the proposed model performs generalized performance analysis on a more comprehensive ORNL dataset. Moreover, detection performance against mixed attacks is also reported in this study. In particular, unknown attacks, i.e. data reconstructed by the SRCAE model, are used to validate the migration capability of the model.

The experimental results on the ORNL dataset are shown in Fig. 12, where each attack target presents evaluation result of fabrication attack and masquerade attack, except for Fig. 12(f). In addition, we present evaluation deviation between fabrication attack and masquerader attack through deviation filling. On the one hand, the validation results reveal that the masqueraded attack is higher than the fabrication attack in terms of ER and FNR metrics, while undoubtedly lower than the fabrication attack in terms of precision, recall and F1-scores, where the max speedometer shows the most significant evaluation deviation. This is attributed to the fact that the masquerade attacks are generated by removing the legitimate target ID frames preceding each injected frame, which removes message confliction and forges the bus periodicity with legal. Hence, many frequency-based IDS almost fail to provide accurate detection. However, the proposed model plays an important role in retrieving valuable features on spatial–temporal features, which provides a more competitive detection performance on masquerade attacks.

Table 5
Performance evaluation on mixed attack from ORNL dataset.

Attack Type	ER (%)	FNR (%)	P (%)	R (%)	F1 (%)
Mixed Fabrication Attack	5.82	4.27	91.33	95.73	93.48
Mixed Masquerade Attack	13.37	9.43	83.69	95.62	86.99

On the other hand, the proposed model provides stable detection performance, where the average ER and FNR are only 1.21% and 1.03%. For detection accuracy, the highest was achieved for the Correlated Signal type, while the lowest, Max Engine Coolant Temp type, also exhibited 94.81%. Also, the Reverse Light On type achieves a maximum of 99.67% for recall, while the lowest is still the Max Engine Coolant Temp type which achieves 97.10%. Thus, the excellent accuracy and recall rates reflect a great average F1-scores.

Since the ORNL dataset also provides instances of fuzzy attacks, this study presents detection results for this type simultaneously, as shown in Fig. 12(f). The validation results show that the model reduces the ER and FNR of fuzzing attacks to 1.07% and 0.65%. In addition, the precision is close to that of HCRL, while the performance of recall and F1 score has improved, reaching 96.89%, 99.35%, and 98.11%, respectively. This is attributed to the fact that the fuzzing attack of HCRL injects random ID and control instructions, yet the data instruction of the fuzzy attack is fixed (i.e., "FFFFFFFFFFFFFFFF") on the ORNL dataset so that the model can easily learn the attack pattern.

From Table 5, it can be seen that the evaluation result of mixed attack between fabrication attack and masquerade attack is also reported, in order to prove the comprehensive detection performance. By implementing the proposed methods, the ER and FNR for mixed fabrication attacks are at a low level of only 5.82%, and 4.27%. Similarly, the mixed masquerade attacks are more difficult to capture, yielding 13.37% and 9.43% deviation compared to the former. Compared to the single target message attack, the

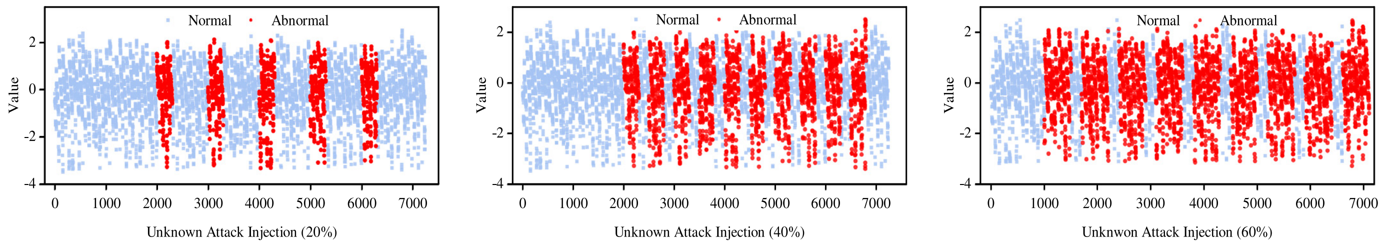


Fig. 13. Visualization of unknown attacks injection on different frequencies and percentages.

Table 6
Performance evaluation on unknown attack from normal message.

Injection rate	ER (%)	FNR (%)	P (%)	R (%)	F1 (%)
20%	1.24	0.84	91.33	99.16	95.07
40%	1.99	1.74	97.71	98.26	97.98
60%	1.48	1.93	98.77	98.07	98.42
Average	1.57	1.50	95.93	98.49	97.15

precision and recall of the mixed fabrication attack decreased, but also reach 91.33% and 95.73%. Although the mixed masquerade attack decreases in precision by 7.64%, the recall decreases by only 0.11%, thus in line with the former, both showing good F1-scores.

6.5. Improvement from unknown attack

At this stage, we utilized the decoder of the SRCAE module to reconstruct normal 2-D data frames, which are regarded as an unknown attack due to the presence of reconstruction errors. The reconstructed unknown attack replaces the normal message by a different percentage and frequency on corresponding time, as shown in Fig. 13.

After obtaining a mixed dataset of unknown attacks, the proposed model is evaluated again and the results are shown in Table 6. For injection types of low frequency and low volume, the model achieves a minimum ER of 1.24% and FNR of 0.84%, but with relatively low accuracy. For the high frequency and missive quantity injection type, the model instead shows a more consistent detection performance due to the large amount of injected data, which brings the bus down directly. In contrast, the detection performance of the model for a limited number of injections at high frequency is modest, but with the highest probability of false alarm.

Nonetheless, the average ER and FNR of the model are only 1.57% and 1.50%, while achieving an average of 95.93% and 98.49% in precision and recall, thus responding to an F1-score of 97.15%.

6.6. Time cost

In this work, we tested the time cost of proposed model in real-vehicle device. Owing to CAN being an intensely temporal-related system, the detection latency is a crucial metric for realizing real-time detection. The detection latency t_l represents as follows:

$$t_l = t_d - t_a \left(t_{cs} \le t_a \le t_{ce}\right), \tag{31}$$

where t_d represents the time that an attack is detected, and t_a is the time when the attack occurred. It is worth noting that t_a is located at the start point t_{cs} and end point t_{ce} of the data aggregation duration of a set batch, i.e., time latency of frame builder module can be represented as t_{bf}. In addition, t_d is the sum of t_{ce}, t_{dp} and t_i, where t_i represents the inference time of proposed model and t_{dp} stands for the time latency of data pre-processing. Thus, the detection t_l is also calculated as follows:

$$t_l = t_{ce} + t_{dp} + t_i - t_a \left(t_i + t_{dp} \le t_l \le t_{bf} + t_i + t_{dp}\right), \tag{32}$$

where $t_{bf} = t_{ce} - t_{cs}$. Clearly, the maximum detection latency occurs the $t_{cs} = t_a$, while the minimum detection latency takes the $t_{ce} = t_a$. In order to realize real-time detection, all time latencies should be minimized. However, the suitable batch size is the crucial to reduce t_l. Hence, we present the testing time on different batch-size, as shown in Fig. 14.

For 2-D frames detection, a significant trend is that as the batch size grows, the cost of time subsequently increases. There is no doubt that it is due to combined factors from data pre-processing, frame builder, and model inference. Since the model detects each 2-D data frame equivalent to detecting consecutive messages, the time cost results of single frame are also reported. However, it shows the opposite trend, which can be attributed to superfluous frame processing and frequent detection inferences. Thus, a compromise solution should be adopted (e.g., the batch size is 256), in order to reduce intrusion detection latency caused by excessive message collection or frequent calls to model inference. We can observe that the proposed model can detect intrusions based on 2-D frames and single frame under time costs of approximately 5.92 ms and 0.0462 ms when using a batch size of 256.

Correspondingly, the proposed model can complete detection inference in 2.5 ms. However, the inference efficiency slows down slightly as the batch size increases. For the single frame, the inference overhead is only 0.02 ms on average. In particular, a similar result shows that the optimal inference efficiency of the model is 0.0193 ms on a single frame and 2.47 ms on a 2-D frame under the same setting. In fact, the time cost and detection performance of the proposed model can be balanced as long as the batch size is set to avoid extreme values, i.e., 64 or 1024.

Moreover, the CAN bus broadcasts approximately 2000 messages per second, which means that the proposed model can achieve 22 times detection volume and 25 times detection inference volume in optimal batches, thus demonstrating the feasibility of implementation in resource-constrained vehicular networks.

7. Conclusion

In this paper, we strive to build an unsupervised-based deep stream clustering model for in-vehicle intrusion detection (DESC-IDS) that can detect various types of known attacks and different injection modes of unknown attacks. The proposed model consists of data preprocessing, feature engineering, feature extraction and deep stream clustering. Through data preprocessing and feature engineering, the feature domains of spatial–temporal are selected, and then constructed by frame builder model. The feature extraction model consists of an encoder and decoder based

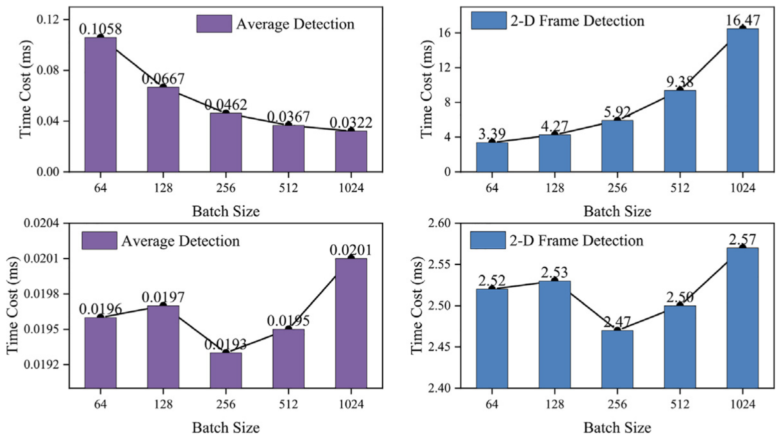


Fig. 14. The testing of average time cost on 2-D frame and single-frame. The upon sub-figures correspond to the total time cost, and the bottom sub-figures correspond to the detection time cost.

on the dilation convolutional neural block under a lightweight design, called SRCAE. It can map valuable spatial–temporal features from high-dimensional 2-D data frames by three crucial constraint terms. Further, the deep stream model is designed by an improved Denstream model, which can capture valid features while dynamically updating the morphology of normal messages based on the characteristics of the data stream.

The performance evaluation of the proposed model implement on the two public datasets that present various types of attacks in the HCRL and attack different targets signal for fabrication and masquerade attacks in ORNL datasets. Compared to baseline methods, the proposed model presents an excellent detection performance under auto bayesian optimization, while ensuring a lower ER and FNR on the HCRL datasets. Similarly, it also shows stable detection results on the ORNL datasets, as well as competitive detection performance against mixed attacks. In particular, the model can detect unknown attacks under different ratios and frequencies of injection, which are reconstructed from the decoder of the SRCAE module through normal messages. Simultaneously, the real-time detection capability of the model has been validated that only costs 0.04 ms to detect each message.

In the future, we plan to further improve the performance in practical applications. It is a feasible way to use the semantics of in-vehicle traffic to correct the representation of the feature extraction module. In addition, the performance of against low-capacity with high-frequency attacks need to be further studied.

CRediT authorship contribution statement

Pengzhou Cheng: Methodology, Data Curation, Visualization, Experiment, Software, Writing – original draft. **Mu Han:** Conceptualization, Resources, Writing – review & editing, Formal analysis. **Gongshen Liu:** Conceptualization, Resources, Writing – review & editing, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Acknowledgments

This work was supported by the following funds: the Joint Funds of the National Natural Science Foundation of China (Grant No. U21B2020); Shanghai Science and Technology, China Plan Project (Grant No. 22511104400); Project of Jiangsu University, China Senior Talents Fund (Grant No. 1281170019). This work also greatly appreciates Dr. Shuaimin Tang's supervision of writing-review & editing.

References

[1] G. Leen, D. Heffernan, Expanding automotive electronic systems, Computer 35 (1) (2002) 88–93, http://dx.doi.org/10.1109/2.976923.
[2] H.M. Song, J. Woo, H.K. Kim, In-vehicle network intrusion detection using deep convolutional neural network, Veh. Commun. 21 (2020) 100198.
[3] M. Han, P. Cheng, S. Ma, PPM-InVIDS: Privacy protection model for in-vehicle intrusion detection system based complex-valued neural network, Veh. Commun. 31 (2021) 100374.
[4] K. Pawelec, R.A. Bridges, F.L. Combs, Towards a CAN IDS based on a neural network data field predictor, in: Proceedings of the ACM Workshop on Automotive Cybersecurity, AutoSec '19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 31–34, http://dx.doi.org/10.1145/3309171.3309180.
[5] H.M. Song, H.K. Kim, Self-supervised anomaly detection for in-vehicle network using noised pseudo normal data, IEEE Trans. Veh. Technol. 70 (2) (2021) 1098–1108.
[6] E. Aliwa, O. Rana, C. Perera, P. Burnap, Cyberattacks and countermeasures for in-vehicle networks, ACM Comput. Surv. 54 (1) (2021) 1–37.
[7] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno, et al., Comprehensive experimental analyses of automotive attack surfaces, in: USENIX Security Symposium, vol. 4, (no. 447–462) San Francisco, 2011, p. 2021.
[8] C. Miller, C. Valasek, Remote exploitation of an unaltered passenger vehicle, Black Hat USA 2015 (S 91) (2015).
[9] J. Liang, Q. Lin, J. Chen, Y. Zhu, A filter model based on hidden generalized mixture transition distribution model for intrusion detection system in vehicle ad hoc networks, IEEE Trans. Intell. Transp. Syst. 21 (7) (2019) 2707–2722.
[10] S. Ohira, A.K. Desta, I. Arai, K. Fujikawa, PLI-TDC: Super fine delay-time based physical-layer identification with time-to-digital converter for in-vehicle networks, in: Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security, 2021, pp. 176–186.
[11] A.R. Javed, S. Ur Rehman, M.U. Khan, M. Alazab, T. Reddy, CANintelliIDS: Detecting in-vehicle intrusion attacks on a controller area network using CNN and attention-based GRU, IEEE Trans. Netw. Sci. Eng. 8 (2) (2021) 1456–1466.
[12] H. Olufowobi, C. Young, J. Zambreno, G. Bloom, Saiducant: Specification-based automotive intrusion detection using controller area network (can) timing, IEEE Trans. Veh. Technol. 69 (2) (2019) 1484–1494.

[13] L. Yang, A. Moubayed, A. Shami, MTH-IDS: A multitiered hybrid intrusion detection system for internet of vehicles, IEEE Internet Things J. 9 (1) (2021) 616–632.
[14] H. Sun, M. Chen, J. Weng, Z. Liu, G. Geng, Anomaly detection for in-vehicle network using CNN-LSTM with attention mechanism, IEEE Trans. Veh. Technol. 70 (10) (2021) 10880–10893.
[15] I. Studnia, E. Alata, V. Nicomette, M. Kaâniche, Y. Laarouchi, A language-based intrusion detection approach for automotive embedded networks, Int. J. Embed. Syst. 10 (1) (2018) 1–12.
[16] T. Dagan, A. Wool, Parrot, a software-only anti-spoofing defense system for the CAN bus, ESCAR EUROPE (2016) 34.
[17] M. Gmiden, M.H. Gmiden, H. Trabelsi, An intrusion detection method for securing in-vehicle CAN bus, in: 2016 17th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering, STA, IEEE, 2016, pp. 176–180.
[18] H.M. Song, H.R. Kim, H.K. Kim, Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network, in: 2016 International Conference on Information Networking, ICOIN, IEEE, 2016, pp. 63–68.
[19] C. Young, H. Olufowobi, G. Bloom, J. Zambreno, Automotive intrusion detection based on constant can message frequencies across vehicle driving modes, in: Proceedings of the ACM Workshop on Automotive Cybersecurity, 2019, pp. 9–14.
[20] M. Foruhandeh, Y. Man, R. Gerdes, M. Li, T. Chantem, SIMPLE: Single-frame based physical layer identification for intrusion detection and prevention on in-vehicle networks, in: Proceedings of the 35th Annual Computer Security Applications Conference, 2019, pp. 229–244.
[21] M. Kneib, O. Schell, C. Huth, EASI: Edge-based sender identification on resource-constrained platforms for automotive networks, in: Network and Distributed System Security Symposium, 2020.
[22] O. Schell, M. Kneib, VALID: Voltage-based lightweight intrusion detection for the controller area network, in: 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom, IEEE, 2020, pp. 225–232.
[23] S. Tariq, S. Lee, S.S. Woo, CANTransfer: Transfer learning based intrusion detection on a controller area network using convolutional LSTM network, in: Proceedings of the 35th Annual ACM Symposium on Applied Computing, 2020, pp. 1048–1055.
[24] J. Zhang, F. Li, H. Zhang, R. Li, Y. Li, Intrusion detection system using deep learning for in-vehicle security, Ad Hoc Networks 95 (2019) 101974.
[25] A. Taylor, N. Japkowicz, S. Leblanc, Frequency-based anomaly detection for the automotive CAN bus, in: 2015 World Congress on Industrial Control Systems Security, WCICSS, 2015, pp. 45–49, http://dx.doi.org/10.1109/WCICSS.2015.7420322.
[26] T. Amarbayasgalan, B. Jargalsaikhan, K.H. Ryu, Unsupervised novelty detection using deep autoencoders with density based clustering, Appl. Sci. 8 (9) (2018) 1468.
[27] M. Hanselmann, T. Strauss, K. Dormann, H. Ulmer, CANet: An unsupervised intrusion detection system for high dimensional CAN bus data, Ieee Access 8 (2020) 58194–58205.
[28] V.S. Barletta, D. Caivano, A. Nannavecchia, M. Scalera, Intrusion detection for in-vehicle communication networks: An unsupervised kohonen som approach, Future Internet 12 (7) (2020) 119.
[29] S.F. Lokman, A.T. Othman, S. Musa, M.H. Abu Bakar, Deep contractive autoencoder-based anomaly detection for in-vehicle controller area network (CAN), in: Progress in Engineering Technology, Springer, 2019, pp. 195–205.
[30] E. Seo, H.M. Song, H.K. Kim, Gids: Gan based intrusion detection system for in-vehicle network, in: 2018 16th Annual Conference on Privacy, Security and Trust, PST, IEEE, 2018, pp. 1–6.
[31] R.I. Davis, S. Kollmann, V. Pollex, F. Slomka, Controller area network (CAN) schedulability analysis with FIFO queues, in: 2011 23rd Euromicro Conference on Real-Time Systems, 2011, pp. 45–56, http://dx.doi.org/10.1109/ECRTS.2011.13.
[32] S. Shreejith, S.A. Fahmy, M. Lukasiewycz, Reconfigurable computing in next-generation automotive networks, IEEE Embedded Syst. Lett. 5 (1) (2013) 12–15.
[33] M. Han, P. Cheng, S. Ma, CVNNs-IDS: Complex-valued neural network based in-vehicle intrusion detection system, in: International Conference on Security and Privacy in Digital Economy, Springer, 2020, pp. 263–277.
[34] C. Young, J. Zambreno, H. Olufowobi, G. Bloom, Survey of automotive controller area network intrusion detection systems, IEEE Des. Test 36 (6) (2019) 48–55.
[35] W. Wu, R. Li, G. Xie, J. An, Y. Bai, J. Zhou, K. Li, A survey of intrusion detection for in-vehicle networks, IEEE Trans. Intell. Transp. Syst. 21 (3) (2019) 919–933.
[36] P. Cheng, M. Han, A. Li, F. Zhang, STC-IDS: Spatial–temporal correlation feature analyzing based intrusion detection system for intelligent connected vehicles, Internat. J. Intell. Syst. 37 (11) (2022) 9532–9561, http://dx.doi.org/10.1002/int.23012.
[37] M.E. Verma, M.D. Iannacone, R.A. Bridges, S.C. Hollifield, P. Moriano, B. Kay, F.L. Combs, Addressing the lack of comparability & testing in CAN intrusion detection research: A comprehensive guide to CAN IDS data & introduction of the ROAD dataset, 2022, arXiv:2012.14600.
[38] Y. Zhang, E. Zhang, W. Chen, Deep neural network for halftone image classification based on sparse auto-encoder, Eng. Appl. Artif. Intell. 50 (2016) 245–255.
[39] F. Cao, M. Estert, W. Qian, A. Zhou, Density-based clustering over an evolving data stream with noise, in: Proceedings of the 2006 SIAM International Conference on Data Mining, SIAM, 2006, pp. 328–339.
[40] Z. Miller, B. Dickinson, W. Deitrick, W. Hu, A.H. Wang, Twitter spammer detection using data stream clustering, Inform. Sci. 260 (2014) 64–73.
[41] M. Ester, H.P. Kriegel, J. Sander, X. Xu, et al., A density-based algorithm for discovering clusters in large spatial databases with nois, in: Kdd, Vol. 96, no. 34, 1996, pp. 226–231.
[42] T. Yu, H. Zhu, Hyper-parameter optimization: A review of algorithms and applications, 2020, arXiv preprint arXiv:2003.05689.

Pengzhou Cheng received the M.S. Degree with the Department of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, China, in 2022. He is currently pursuing the Ph.D. Degree with the Department of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, 201100, China.

His primary research interests include cybersecurity, machine learning, time-series data analytic, intelligent transportation systems, and intrusion detection system.

Mu Han (Member, IEEE) received the Ph.D.degree in Computer Science from Nanjing University of Science and Technology. She is an Associate Professor with the School of Computer Science and Communication Engineering, Jiangsu University. Her primary research interests are in the areas of Cryptography, Security and Communication in-vehicle networks, the design of security protocols for smart cars and Information Security.

Gongshen Liu received his Ph.D. degree in Department of Computer Science from Shanghai Jiao Tong University. He is currently a professor with the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University. His research interests cover Natural Language Processing, Machine Learning and Artificial Intelligent Security.