Beyond vanilla: improved autoencoder-based ensemble in-vehicle intrusion detection system.

RAJAPAKSHA, S., KALUTARAGE, H., AL-KADRI, M.O., PETROVSKI, A. and MADZUDZO, G.

2023

Contents lists available at ScienceDirect

Journal of Information Security and Applications

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

Beyond vanilla: Improved autoencoder-based ensemble in-vehicle intrusion detection system

Sampath Rajapaksha[a,*], Harsha Kalutarage[a], M. Omar Al-Kadri[b], Andrei Petrovski[a], Garikayi Madzudzo[c]

[a] *Robert Gordon University, Aberdeen, UK*
[b] *University of Doha for Science and Technology, Doha, Qatar*
[c] *Horiba Mira Ltd, Nuneaton, UK*

ARTICLE INFO

Keywords:
Controller Area Network (CAN)
Machine learning
Automotive cybersecurity
Deep learning
Autoencoder
Anomaly detection

ABSTRACT

Modern automobiles are equipped with a large number of electronic control units (ECUs) to provide safe, driver assistance and comfortable services. The controller area network (CAN) provides near real-time data transmission between ECUs with adequate reliability for in-vehicle communication. However, the lack of security measures such as authentication and encryption makes the CAN bus vulnerable to cyberattacks, which affect the safety of passengers and the surrounding environment. Detecting attacks on the CAN bus, particularly masquerade attacks, presents significant challenges. It necessitates an intrusion detection system (IDS) that effectively utilizes both CAN ID and payload data to ensure thorough detection and protection against a wide range of attacks, all while operating within the constraints of limited computing resources. This paper introduces an ensemble IDS that combines a gated recurrent unit (GRU) network and a novel autoencoder (AE) model to identify cyberattacks on the CAN bus. AEs are expected to produce higher reconstruction errors for anomalous inputs, making them suitable for anomaly detection. However, vanilla AE models often suffer from overgeneralization, reconstructing anomalies without significant errors, resulting in many false negatives. To address this issue, this paper proposes a novel AE called Latent AE, which incorporates a shallow AE into the latent space. The Latent AE model utilizes Cramér's statistic-based feature selection technique and a transformed CAN payload data structure to enhance its efficiency. The proposed ensemble IDS enhances attack detection capabilities by leveraging the best capabilities of independent GRU and Latent AE models, while mitigating the weaknesses associated with each individual model. The evaluation of the IDS on two public datasets, encompassing 13 different attacks, including sophisticated masquerade attacks, demonstrates its superiority over baseline models with near real-time detection latency of 25ms.

1. Introduction

Modern automobiles are becoming highly autonomous to provide a more comfortable, safe and convenient experience to drivers and passengers [1]. This includes services such as adaptive cruise control, lane departure warning, automated parking assistance, and infotainment systems. To provide these services, automobiles are equipped with numerous microprocessor-based ECUs, such as engine and transmission control units. These ECUs need to communicate with each other to exchange various information. This demands a unified network which supports near real-time data transmission with sufficient bandwidth and adequate reliability [2]. The CAN bus, a message-based protocol, meets these requirements and is therefore considered the de facto standard for in-vehicle communication. However, even though the CAN bus provides the required communication capabilities, it lacks security features such as encryption and authentication. These security flaws along with ID-based priority mechanism and broadcast transmission make the CAN bus vulnerable to cyberattacks. Many security researchers have demonstrated that it is possible to attack modern automobiles by compromising the CAN bus. This can be achieved through obtaining access to OBD-II port [3,4], over the air (OTA) updates or communication channels such as Bluetooth, Wifi and cellular networks [5]. Attackers could gain physical control and activate several vehicle systems such as engaging the brakes, turning on the wind-shield wipers, activating the locking mechanism, and changing the vehicle's speedometer. In [6], the authors used the head unit (radio) to gain remote access, which resulted in obtaining physical control of the 2014 Jeep Cherokee model.

* Corresponding author.
E-mail addresses: s.rajapaksha@rgu.ac.uk (S. Rajapaksha), h.kalutarage@rgu.ac.uk (H. Kalutarage), omar.alkadri@udst.edu.qa (M.O. Al-Kadri), a.petrovski@rgu.ac.uk (A. Petrovski), garikayi.madzudzo@horiba-mira.com (G. Madzudzo).

https://doi.org/10.1016/j.jisa.2023.103570

Available online 14 August 2023

Cyberattacks which provide physical control risk the safety of the vehicle passengers.

Intrusion detection systems (IDSs) have been proposed to identify malicious activities in in-vehicle networks (IVNs) [7]. IDSs can be categorized as signature-based and anomaly-based detection [8]. Anomaly-based detection has gained the attention of researchers due to the capability of novel attack detection. Anomaly-based CAN IDSs can be further classified as statistical, machine learning (ML), rule-based, and physical fingerprint methods [9,10]. ML-based CAN IDSs have been proven to be effective than other approaches due to their capability of handling large data volumes of CAN traffic with multiple features [7,9]. However, the structure and semantics of CAN data are not available to open access and are considered confidential proprietary for vehicle manufacturers [7]. These specifications are stored in a file known as CAN DataBase (DBC). This makes developing IDSs for widespread adoption difficult and achieving full accuracy for some attacks is unrealistic [11]. IDSs that utilize only the CAN ID field effectively detect injection attacks with near real-time detection [12,13]. However, time-series CAN payload data are critical for detecting advanced attacks such as masquerade attack [14,15]. CAN payload field support data up to 64 bits which can transmit different information from one ECU to other ECUs [13,16]. Since a modern vehicle includes up to 150 ECUs [17], this creates multi-variate time-series data for each CAN ID. Additionally, the CAN bus transmits up to 2500 messages per second [12]. These characteristics make the IVNs data highly complex. Therefore, identifying a wide variety of attacks on the CAN bus is challenging with a single IDS which uses part of the CAN frame.

AE-based IDSs are used for anomaly detection in many application domains including IVNs [18–22]. It is generally assumed that reconstruction errors are significantly higher for anomalous samples than benign samples. However, AEs are prone to overgeneralization [23] and this causes the reconstruction of the anomalous inputs without higher reconstruction errors leading to many false negatives. This is a critical issue for some cybersecurity problems such as IVN security. To alleviate this problem and address the aforementioned challenges, this paper proposes an ensemble IDS of a GRU network and a novel AE model. The AE model modifies the latent space of the vanilla AE with a novel feature selection technique to identify cyberattacks on the CAN payload data in near real-time. The main contribution of this paper can be summarized as follows:

1) This paper proposes an ensemble IDS by integrating a GRU-based model [12] and an improved AE model to detect both point and contextual anomalies in the CAN bus. The proposed AE model, named as Latent AE, exploits the association of CAN payload variables to identify anomalies. Latent AE address the issue of overgeneralization of vanilla AEs for anomaly detection in the CAN bus.
2) Latent AE is a lightweight model; Cramér's statistic-based [24] feature selection technique reduces the complexity of the data by removing weakly associated features. This helps improve the model's computation efficiency and accuracy due to the removal of noise data. Feature selection avoids the need for a large dataset compared to the dataset requirement for a model with all variables.
3) Latent AE is efficient and can be deployed in resource-efficient edge devices. The proposed data structure incorporates CAN IDs into the payload variable, allowing to use only one model for all CAN IDs without building separate models for each CAN ID compared to the existing proposals in the literature.

The rest of this paper is structured as follows: Section 2 provides the background. Section 3 presents the related work. The proposed algorithm is explained in Section 4. In Section 5, the experiment results and performance evaluations are presented. Finally, Section 6 concludes the paper.

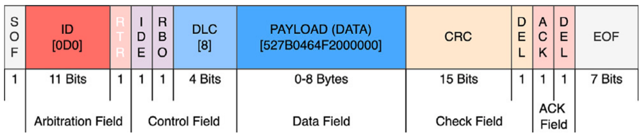


Fig. 1. CAN bus data frame with example values for ID and payload fields.

2. Preliminaries

2.1. Controller area network (CAN bus)

CAN is a lightweight broadcast-based communication protocol. A CAN data frame consists of seven fields that support data transmission. These are: start of frame (SOF), arbitration field (CAN ID), control field (DLC), payload (data), cyclic redundancy code (CRC), acknowledgement (ACK) and end of frame (EOF). These fields are depicted in Fig. 1 with the respective bit-lengths. CAN ID and payload are considered as the most important fields of the CAN frame [25]. CAN ID is unique for an ECU and two or more ECUs cannot use the same ID. ID represents the identifier of the message and used to prioritize messaged based on the ID values. Lower IDs get the higher priority, while higher IDs get the lower priority. This is used to manage the concurrent messages in the CAN bus. CAN payload values contain the information that must be transmitted over the network. This supports up to 64bits (8 bytes) of data transmission. Generally, both ID and payload are available in hexadecimal format in raw CAN data. Specification related to the CAN frame is available in DBC file, which is not publicly available. These specifications change based on vehicle make, model, year and trim [25].

2.2. Attacks on IVNs

ECUs in a vehicle typically transmit frames at a fixed interval [3,12,26,27]. Due to the broadcast property of the CAN bus, all ECUs receive unencrypted messages transmitted through the bus. This makes the CAN bus vulnerable to a sniffing attack, where an attacker can listen to all the messages and records them to analyse the CAN data. Due to the lack of authentication, any node can transmit a frame to the CAN network. This enables an attacker to inject malicious frames using a compromised ECU. In addition to these vulnerabilities, an attacker can use the ID-based priority mechanism to inject higher priority IDs and create a denial-of-service (DoS) attack.

Attacks on IVNs can be mainly classified into three categories as fabrication (injection), suspension and masquerade (impersonation) attacks [14,15]. Fabrication attacks can be launched by inserting new frames into the CAN bus. Some common fabrication attacks on IVNs are DoS, fuzzing, replay and spoofing [10]. DoS attacks try to make communication services unavailable by sending a large number of high-priority IDs. In a fuzzing attack, the attacker sends frames with randomly generated ID, DLC and payload values. Replay attacks record the benign frames of the network and inject them at different times. For instance, a frame that transmits vehicle RPM values can be injected at a later stage making it appear in an unusual context. Spoofing attacks, also known as targeted ID attacks, are somewhat similar to fuzzing attacks but target specific IDs without randomly injecting them. The payload of these frames changes into malicious values. Generally, even though it is easy to implement fabrication attacks, it requires the attacker to continuously inject these frames at a higher rate to override the benign frames [15,26]. Suspension attacks remove benign frames. This requires a weakly compromised ECU, preventing it from transmitting some or all messages [15]. In a masquerade attack, a compromised node impersonates another node. For example, the attacker can monitor and learn about frames transmitted by ECU A. Then, the attacker can stop ECU A by transmitting frames and instead use a compromised

ECU B or new node to transmit ECU A frames with malicious data. However, these sophisticated masquerade attacks require extremely low-level access to the CAN transceiver and therefore more, difficult to perform than simple injection attacks [26]. Masquerade attacks require advanced detection methods as typically these are not changing the ID frequency-related features.

3. Related work

Early experimental attacks [3,6] and more recent experiments such as [28,29] motivated researchers toward implementing countermeasures against cyberattacks on the CAN bus. These attacks change the CAN ID field, or CAN payload field patterns. Therefore, CAN ID-based and CAN payload-based IDSs have been proposed in the literature to detect IVN attacks. This section investigates the most recent and relevant works in IVN attack detection and improved AEs for anomaly detection.

3.1. CAN ID-based IDSs

Collecting attack data is more expensive and risky in vehicle networks than in benign data collection. Supervised learning approaches might have a low generalization capability as these algorithms learn the attack patterns specific to the used training dataset. Due to the nature of this problem domain, repeating the same attack pattern in future attacks is less probable. On the other hand, unsupervised approaches learn the normal behaviour only using benign data and deviation from these patterns identified as anomalous events. Therefore, unsupervised (or one-class) anomaly detection is more appropriate for vehicle networks to identify various attacks, including unknown ones [11].

CAN ID-based IDSs use CAN ID sequences or time-related features to detect attacks on the CAN bus. Long short-term memory (LSTM) based IDS proposed in [30], predicted the next CAN ID and then compared the predicted ID with the actual ID. This only achieved 60% accuracy. Generative adversarial networks (GAN) and convolutional adversarial AE-based model [31] was trained with unlabelled data to learn the normal patterns. Experimental results showed that the proposed model outperforms baseline models for detection rate and latency. However, this work was only limited to message injection attack detection. Context-aware anomaly detector proposed in [32], used the N-gram distributions of CAN ID sequences. N-gram-based methods are capable of capturing the context of CAN sequences. However, this often leads to high computational overhead as N increases. GRU and time-based ensemble model was proposed in [10] to overcome the computational inefficiency of N-gram based models and to detect a wide variety of attacks. The GRU-based model predicted the next CAN ID, whereas the time-based model monitored ID inter-arrival times to detect anomalies. The major limitation of the CAN ID-based IDSs is the inability to detect advanced attacks which only alter the CAN payload field patterns.

3.2. CAN Payload-based IDSs

ML-based classifiers such as one class support vector machine (OCSVM), deep learning-based models such as AEs and LSTM have been used to detect attacks on the CAN payload data. In [33], the authors used one class compound classifier to detect fuzzing attacks in IVN. They only considered three payload values relevant to three CAN IDs. Evaluation results showed that the proposed model produced many false positives. The authors then suggested an ensemble of detection methods to overcome the problems that arise when using only one classifier. In [11], the authors used an algorithm to concatenate the adjacent payload values of each ID based on their value changes. This reduces the dimensionality of the payload of each ID. Pearson correlation was used to cluster the different fields and used the local outlier factor, compound classifier and OCSVM algorithms for the attack detection. However, the results were not acceptable to use in real-world situations due to the high false positive rate.

In [34], the authors used a LSTM model to predict the next CAN payload for each ID. Log loss of each bit was considered to form the anomaly signal. A similar model was proposed by [35] to predict the CAN measurements such as RPM and break positions. Access to these measurements is challenging without having the DBC file. In [36], a separate LSTM model was used to predict the next payload of each ID and concatenated them using a joint latent vector. The authors used a real vehicle dataset with 13 IDs and a synthetic dataset (SynCAN) with 10 IDs for the performance evaluation. The used anomaly score was only feasible with a limited number of signal values. To train LSTM models, 5000 consecutive messages were considered. This will be computationally expensive for modern vehicles with a large number of ECUs. In [22], 81 payload signals were grouped into 32 subgroups based on the signal relationships and trained AE models for each group. However, it is important to note that this approach relied on pre-identified signals as input features, which are typically not accessible without obtaining the CAN DBC file. Similar GRU and LSTM AEs were used in [18,19] to reconstruct the payload values of each ID. In [37], the authors improved the GRU-based IDS [18] by replacing the GRU with a LSTM and introducing a self-attention layer. In [38], the authors introduced an IDS using a temporal convolutional neural network. A decision tree-based classifier was used as the attack detector. All of these models [18,19,37,38] trained separate models for each CAN ID. However, a major limitation common to all of these models was ignoring the interactions from other ID payload values. In particular, this might limit the detection of contextual anomalies. Moreover, these IDSs might require higher memory to store multiple IDSs trained for each ID. It has shown that the association of payload data of different IDs are useful in intrusion detection [19].

CANShield [39], a multiple convolutional AE-based ensemble model only used high-priority signals to reduce the model complexity. This showed a high detection rate for injection and masquerade attacks. However, the signal selection depends on the semantic knowledge of CAN payload and therefore, has low generalization capability for other vehicles without having CAN DBC files. LSTM-based IDS proposed in [40] trained separate models for each IDs. This model utilized the current payload of the ID and the payload of other IDs which belong to a selected time window. There is a risk of ignoring important associations for a small time window due to specific IDs not being included in the window. On the other hand, selecting a larger window might increase the computational complexity and the variable noise due to containing non-associated variables. Larger windows demand a large number of data to learn all the variabilities. In general, AE-based IDSs demonstrated a higher detection rate for payload attacks.

In general, CAN payload-based IDSs use two approaches [15]. The first approach is the black-box approach, where the data frame is considered as a string of bits without decoding the signals they represent. The second approach involves decoding the raw data field into constituent signals and using the identified signal values as inputs. A few IDSs have used the encoded signals as inputs [22,33,39] while the majority of payload-based IDSs [11,18,19,34,35,37,38,40] have employed the black-box approach. A meta-analysis of a number of papers of [10] also reveals that most payload-based IDSs use the payload field without decoding it into signals. IDSs that utilized decoded signal values either used CAN DBC files or reverse engineering approaches. Although these models achieved higher attack detection rates due to selecting only relevant signals, they were not vehicle agnostic [41]. On the other hand, IDSs that used the black-box approach suffered from low attack detection and high computational resource requirements as they considered all features of the payload field. Therefore, there is a clear need for a vehicle-agnostic and lightweight payload-based model to detect advanced attacks such as masquerade attacks. This work focuses on utilizing raw payload values without decoding them into actual signal values, allowing the proposed solution to be readily adaptable across car makes and models. Additionally, by focusing only on important features, the proposed payload-based model is lightweight. Hence, the payload-based model of the proposed IDS is distinct and unique compared to existing payload-based IDSs.

3.3. *Improved autoencoders for anomaly detection*

AEs have been used for anomaly detection in many cybersecurity applications such as IoT security [42] and IVN security [18]. Generally, it is assumed that AEs trained on benign data produce higher reconstruction errors for anomalous data. However, this assumption does not always hold in practice due to the overgeneralization [23,43]. To mitigate this drawback, a few improved AEs are proposed in the literature. In [23], the authors proposed a memory augment AE by adding a memory module to the vanilla AE. The memory module was used to store the prototypical elements of normal data during the model training. Instead of the latent vector produced by the encoder, they used the most relevant memory items of the latent vector as the input to the decoder. This approach requires having separate memory modules when normal datasets have different groups such as CAN IDs in CAN data, due to different data patterns. Also, some datasets might demand a large memory to store prototypical elements [44]. Latent space distribution was used in [45] to detect the anomalies. Usage of k nearest neighbour calculation demands higher computational resources for successful anomaly detection. In [46], a set of multi-layer perceptron (MLP) was used in the latent space to predict each latent space element and used the predicted array as the input to the decoder. None of these approaches is suitable to deploy in IVN for anomaly detection due to the limited computational resources.

4. Methodology

This section explains the selected datasets and attacks, implementation details of CAN ID-based model, CAN payload-based model with the proposed improved AE architecture to improve the anomaly detection of the vanilla AE. Furthermore, we discuss the integration of both CAN ID-based detection and payload detection within the ensemble model.

4.1. *Threat model and datasets*

In this work, we use two publicly available datasets, The Real ORNL Automotive Dynamometer (ROAD) CAN intrusion dataset [15] and SynCAN [36] to evaluate the proposed model. ROAD dataset is the most realistic CAN attack dataset with verified attacks [12,39]. It was collected via OBD-II port under fully compromised ECU mode. This consists of 12 benign datasets encompassing a diverse range of driving activities, including driving, accelerating, decelerating, braking, and reversing, spanning a duration of 3 hours. The details of these datasets are presented in Table 1. ROAD dataset consists of fabrication and masquerade attacks. A fabrication attack uses a strongly compromised ECU to inject malicious frames which alter the ID and payload fields. This includes fuzzing and targeted ID attacks. In contrast, masquerade attack is the most sophisticated type of attack which uses a strongly compromised ECU to inject malicious messages without changing the CAN ID sequences. In ROAD dataset, this was created during the post-processing by removing the legitimate target ID frames relevant to each injected frame. Masquerade attacks were created for each type of targeted ID attack. The attacks shown in Table 2 selects for the performance evaluation.

Unlike the ROAD dataset which has raw CAN data with up to 64 bits payload for each CAN ID, the SynCAN dataset is a synthetic dataset available in normalized signal level. The benign dataset comprises 24 hours of driving data, divided into four separate datasets. It contains 10 CAN IDs and 20 signals. Signals for an ID range from one to four. This includes five types of advanced attacks performed during the post-processing. These attacks are summarized in Table 3. Even though SynCAN dataset and attacks are synthetic, this is a good dataset to evaluate the proposed IDS as each attacks targets multiple signals during different time intervals.

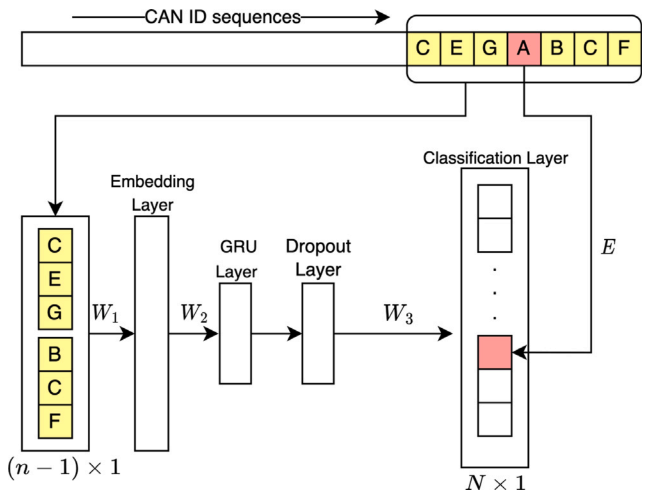


Fig. 2. GRU-based model architecture.

4.2. *CAN ID-based detection*

This section discusses the design details of the CAN ID-based model with the anomaly detection procedure.

4.2.1. *GRU-based model*

This work uses the GRU-based model proposed in [12] as the CAN ID-based model. By utilizing the sequential behaviour of the CAN frames, CAN intrusion detection can be formulated as the next CAN ID prediction task. Given a context of n IDs from the left side (pre-context) of a CAN sequence, it can predict the next CAN ID with a p probability. This is similar to the next word prediction task in a natural language processing (NLP) model. If the predicted next CAN ID's probability is below a certain pre-defined threshold, then it can be identified as an anomalous frame in the given context. Therefore, this can detect contextual anomalies. However, due to the randomness incurred from jitters and ID-based priority, there might be a large number of possible next CAN-IDs for the given context. As a solution, both the left side and right side (post-context) contexts can be used to reduce the possible CAN-IDs. This is equivalent to the continuous bag-of-words (CBOW) model architecture proposed in [47] where it learns the word vectors representing the middle word's meaning and the context words. Given the context from both sides of an ID reduces the number of possible distinct centre IDs. GRU-based model utilized the centre ID prediction to improve the IVN intrusion detection.

The model architecture of this method is depicted in Fig. 2. A sliding window of size n is selected to get the context for a CAN ID. Let N be the total number of unique CAN IDs. First, this uses an embedding layer which takes the input of vectorized benign CAN ID sequences of size $n-1$. The objective here is to learn accurate word vectors that encode semantic relationships for all the IDs in the CAN bus. Then GRU layer is used to capture the temporal pattern of the sequential data. GRU is a variant of LSTM with only two gates: reset and update. Therefore, GRU is more computationally efficient with low memory overhead. A dropout layer is used to overcome the model overfitting. Finally, a dense layer of size N is used as the classification layer with the softmax activation function to get the probability for each CAN ID. The overall objective of this model is to learn to predict the centre ID given the

Table 1
Description of ROAD benign datasets.

Dataset	Driving activities	Driving time (s)
Basic long	Basic driving activities	1250
Basic short	Basic driving activities	444
Reverse	Basic reverse activities	51
Benign anomaly	Driving while trying to cause benign anomalies	456
Extended long	Basic driving activities and more complex/one-off activities	657
Extended short	Basic driving activities and more complex/one-off activities	359
Radio infotainment	Playing with radio and infotainment unit while idling and driving	390
Idle radio infotainment	Playing with radio and infotainment unit while idling	660
Drive winter	driving and accelerating in colder conditions	47
Exercise all bits	Trying to exercise full range of all signals	2172
Highway street driving	Drive in parking lots, city streets and highways	469
Highway street driving long	Drive in parking lots, city streets and highways	3764

Table 2
Description of ROAD attack datasets.

Attack	Description
Correlated signal	Inject false speed values to stop the car
Max speedometer	Change a payload byte to display false speedometer value
Reverse light on and off	Change a payload bit to change reverse light status

Table 3
Description of SynCAN attack datasets.

Attack	Description
Plateau	Change signal value into a constant value
Continuous	Change signal value so that it slowly drifts away from its actual value
Playback	Change signal value to a recorded value
Suppress	Prevent an ECU sending messages
Flooding	Inject selected IDs with high frequency

$(n-1)/2$ pre and post-context. This can be achieved by minimizing the objective function of categorical cross-entropy. This is given by:

$$E = -\sum_{i=1}^{N} y_i log(\hat{y}_i) \tag{1}$$

where y_i is the true label and $\hat{y}_i$ is the predicted softmax probability for the ith class. Weights W_1, W_2, and W_3 are learned using backpropagation during the model training GRU-model is trained with a sufficiently large benign dataset to learn benign sequences. During the inference, incorrect classification can be identified as anomalous frames based on a pre-defined anomaly threshold. Counting weak anomalies over a window help to minimize the false positives. Therefore, this approach considers a small observation window of time T to identify anomalous or benign status. This procedure is shown in Algorithm 1. First, this extracts the context IDs (x), centre ID (y) and associated time stamp (t) from streaming CAN data F. Using the trained GRU-based model M, softmax probability for the centre ID is predicted. If the predicted softmax probability for the centre ID is less than a pre-defined threshold ω, the frame is declared as a weak anomaly. These anomalous and benign frames in the observation window are used to detect the window as anomalous or benign based on a window threshold of ψ.

4.2.2. Threshold estimation

CAN ID-based detection uses two thresholds. Since the model training procedure totally depends on benign data, all thresholds must be chosen based on the benign datasets. Therefore, a separate benign dataset is used to estimate both thresholds. For anomaly threshold ω, softmax probabilities were calculated for all IDs in the benign sample. While the minimum values of each ID are ideal as the threshold values to minimize the false positives, there might be unseen benign sequences in the threshold estimation benign dataset, which were not observed in the training dataset. As a result of this, these benign frames tend to get lower probabilities. Therefore, we consider the Nth lowest quantile values as the anomaly thresholds. Fig. 3 shows a softmax probability distribution for a selected ID. Window threshold ψ is defined in a way that it produces a minimum false positive rate for a fixed window size of time T. For example, if benign windows produce an average of two false positives, then at least three anomalous frames should be there to consider the window status as anomalous. This helps to reduce the false positives of the proposed model.

Algorithm 1 CAN GRU-based anomaly detection

```
Input: Streaming CAN data $F$, Anomaly threshold $\omega$, Window threshold $\psi$, Time window $T$, Trained model $M$
Output: Anomaly status for each window
1: while $F$ is not empty do
2:     read $x, y$, time_stamp $t$, $t\_min$
3:     while $t - t\_min \leq T$ do
4:         Init: Benign count $C_b = 0$ , Anomaly count $C_a = 0$
5:         $\hat{y} = M(x)$
6:         if $\hat{y} < \omega$ then                    ▷ for $id$ $y$
7:             Declare $y$ as a weak anomaly
8:             $C_a = C_a + 1$
9:         else
10:            Declare $y$ as a benign
11:            $C_b = C_b + 1$
12:        end if
13:    end while
14:    if $C_a/(C_a + C_b) > \psi$ then
15:        Return Anomaly
16:    else
17:        Return Benign
18:    end if
19:    $t\_min \leftarrow t$
20: end while
```

The GRU-based model which only utilizes the CAN ID field is suitable for detecting injection attacks as CAN frame injection introduces the new CAN ID sequences for observation windows. However, this might not be suitable for detecting some attacks which target the payload field. This is due to some sophisticated attacks such as masquerade attacks might not disrupt the frequency or ID distributions [15]. Instead, such attacks require an IDS which utilizes the payload data.

4.3. CAN Payload-based detection

This section provides a comprehensive overview of the proposed novel AE-based model and its anomaly detection procedure.

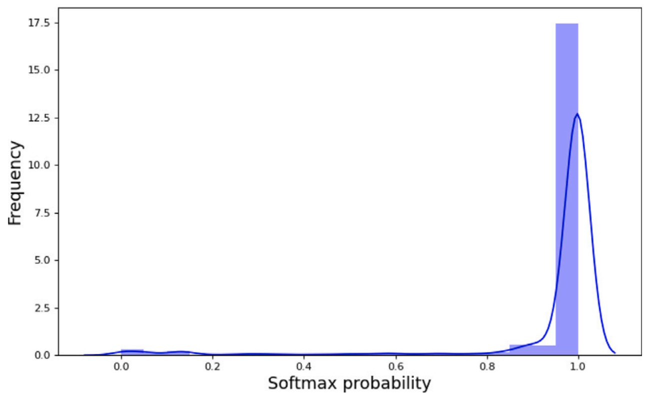


Fig. 3. Softmax probability distribution of ID 580.

4.3.1. *Data pre-processing*

CAN data can be considered as time series data as only one frame transmits at a given time due to the priority-based arbitration mechanism. Sequential behaviour of CAN frames led previous research [18, 34,36] to use deep learning models which can process sequential data. These models include recurrent neural networks (RNNs) such as LSTM and GRU. Despite the high detection rates provided by RNN-based deep learning models, there are a few drawbacks of using these models for CAN intrusion detection. Typically LSTM or GRU nodes have a higher number of trainable parameters and are therefore, computationally expensive than feed-forward neural networks with the same number of nodes. This is one major disadvantage for CAN bus considering the computational resource availability in IVNs. Additionally, it has shown that the association of payload data of other IDs are very important for CAN intrusion detection [19]. In this case, the input sequence should be large enough to capture important associations from other IDs. This also increases the computational complexity of the model and the input frame might include a large number of unassociated variables, which might cause overfitting. This will lead to reduce the detection capability of the model in a real-world deployment.

To address these problems, we transform CAN payload data into a format suitable for anomaly detection. First, hexadecimal 64 bits CAN payload splits into eight bytes and converts into integer values. These eight features might include constant/empty, categorical and numerical discrete variables ranging from 0 to 255. All features are normalized to lie between zero to one using ID feature-wise min–max scaling. This helps to avoid slow, unstable training and the problem of exploding gradients. One dimensional array is used to hold the most recent values of other CAN IDs along with the current ID. This preserves the sequential property of the CAN data and allows to learn the context of payload values. This approach is different to the approach used in [39]. It only considered a few pre-identified variables and created 2-D frames for a specific time window. In contrast, the proposed data structure in this work, consider the context from all other IDs as time based window selection might ignore the association from some IDs. Further, the proposed approach assume no prior knowledge of CAN specification which is not available for open access. Table 4 shows a subset of CAN IDs and features to illustrate the data transformation. Table 4 shows CAN transmission between 77.04383 s and 77.04387 s for four CAN IDs. D1 and D2 represent the first and second features of CAN payload. Each ID has up to eight features (D1...D8). Table 4 shows the snapshot of the transformed data. Each array (row) is updating with the current ID payload values. For example, ID 125 transmission at 77.04387 s, results in an array update for feature 125_D1 and 125_D2 whereas other feature in the array hold the previous array values. These features serve as the contextual information for CAN ID 125, representing the most recent transmitted values on the CAN bus. An array holds the most recent values for all the feature of the current ID and associated variables for the ID.

The ROAD dataset incorporates zero padding to fill empty variables, resulting in a 64-bit payload field. In the transformed data structure, variables that maintain a value of zero across the entire dataset are considered to be instances where zero padding was employed. As a result, these variables are omitted from the data structure. On the other hand, the SynCAN dataset does not require any preprocessing since variable values are already normalized and empty variables are not present. Nevertheless, for realistic CAN data that resembles the format of the ROAD dataset, it is crucial to apply these preprocessing steps. The transformed data structure supports implementing one model to detect the anomalies in all CAN IDs without implementing separate models for each CAN ID.

4.3.2. *Feature selection*

A modern vehicle could include up to 150 IDs, with around 150 ECUs. Based on this, the transformed data structure in Table 4 could have up to 8N features, where N is the total number of IDs. Including all these features can significantly increase the complexity and computational cost of the IDS. Alternatively, focusing on essential features can reduce complexity and offer a practical solution with near real-time detection. However, determining the importance of features for each ID is challenging without knowledge of the CAN specification. One viable approach is to utilize feature associations to identify the important features. Previous works [11,14,48] used the Pearson correlation coefficient to identify important variables clusters while [39,49] employed feature correlation to identify anomalies. In [11], the authors utilized the raw payload values as features, whereas others focused on the decoded signal values. As per the analysis in Section 5.1.2, the majority of payload features have only a limited number of unique categorical values. In the case of nominal categorical features such as the 3rd byte of ID 0D0, the Pearson correlation may not be suitable for estimating feature associations. Consequently, the use of Pearson correlation can restrict the identification of associated features in CAN payload data. For instance, in [48], one of the selected sensor values was the Gear. This particular sensor exhibited the lowest Pearson correlation with other sensor values. Notably, the Gear sensor had only 7 possible values, similar to a nominal categorical variable. Consequently, the Pearson correlation failed to capture the highly associated variables in this case.

Pearson's chi-squared statistic based Cramér's V statistic measures the strength of the association between two discrete variables which have two or more levels [50]. This is calculated using:

$$V = \sqrt{\frac{\varphi^2}{min(k-1, r-1)}} = \sqrt{\frac{\chi^2/n}{min(k-1, r-1)}} \quad (2)$$

where φ is the phi coefficient, χ^2 is the chi-square statistic, n is the total number of observations, k is the number of columns and, n is the number of rows of the contingency table. However, Cramér's V can have a large bias for finite samples. Bias correction was proposed by [51] to mitigate this issue and the corrected value is given by:

$$\hat{V} = \sqrt{\frac{\hat{\varphi}^2}{min(\hat{k}-1, \hat{r}-1)}} \quad (3)$$

where,

$$\hat{\varphi}^2 = max\left(0, \varphi^2 - \frac{(k-1)(r-1)}{n-1}\right)$$

and

$$\hat{k} = k - \frac{(k-1)^2}{n-1}, \hat{r} = r - \frac{(r-1)^2}{n-1}$$

Cramér's V statistic ranges from 0 to 1, with 0 indicating no association between variables and 1 indicating a perfect association between

Table 4
Data transformation from normalized CAN payload to amalgamated CAN payload (N.B. Only a subset of IDs and features are shown).

(a) Snapshot of the normalized CAN payload. This represents only four CAN IDs and two features (D1, D2) out of eight features.

Time	CAN ID	D1	D2
77.04383	125	0.1142	0.0000
77.04384	354	1.0000	0.4481
77.04385	5E1	0.1574	1.0000
77.04386	0A7	0.3333	0.8470
77.04387	125	0.1152	0.3278

(b) Snapshot of the transformed CAN payload. Each row represents the change in the variables over time as each CAN ID transmits. Variable values of current ID is shown in bold.

Time	CAN ID	125_D1	125_D2	354_D1	354_D2	5E1_D1	5E1_D2	0A7_D1	0A7_D2
77.04383	125	**0.1142**	**0.0000**	0.2000	0.4481	0.1574	1.0000	0.1759	0.9803
77.04384	354	0.1142	0.0000	**1.0000**	**0.4481**	0.1574	1.0000	0.1759	0.9803
77.04385	5E1	0.1142	0.0000	1.0000	0.4481	**0.1574**	**1.0000**	0.1759	0.9803
77.04386	0A7	0.1142	0.0000	1.0000	0.4481	0.1574	1.0000	**0.3333**	**0.8470**
77.04387	125	**0.1152**	**0.3278**	1.0000	0.4481	0.1574	1.0000	0.3333	0.8470

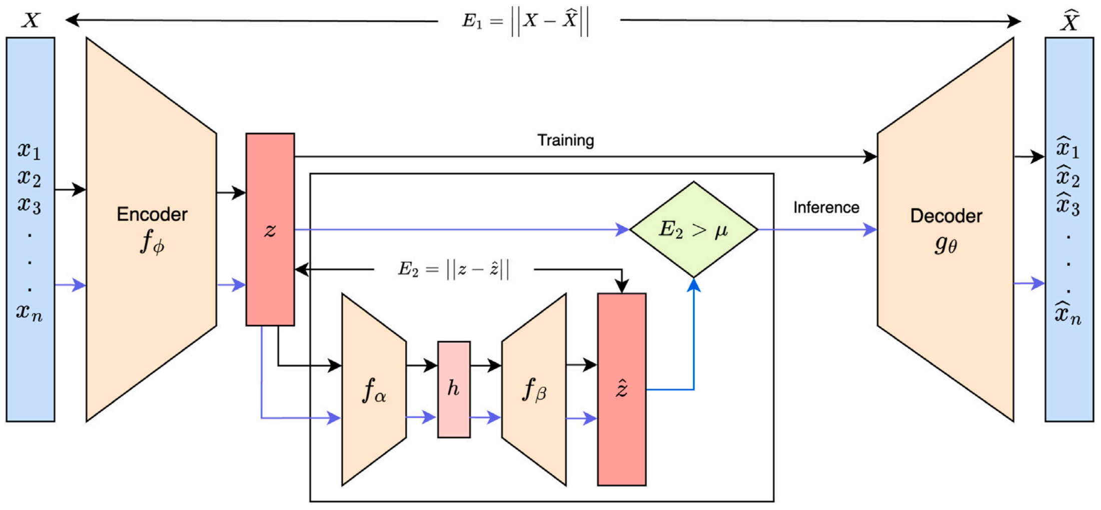


Fig. 4. Overview of the Latent AE.(The black line indicates the training process while the blue line indicates the inference process) (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

variables [52]. This paper uses corrected Cramér's $\hat{V}$ statistic to identify the associated features for all IDs. The associated feature selection procedure is shown in Algorithm 2. First, this algorithm selects the features X_1 of an ID and calculates the strength of associations $\hat{V}$ with the feature X_2 of other IDs using the contingency tables. The objective of this is to remove unassociated features from the other IDs. Therefore, association calculation between features of the same ID is not necessary as all features of the ID is keep in the array without removing them. Threshold λ is used to control the number of feature selections based on the desired strength of associations. If a feature does not have highly associated features with over λ, then the feature with the highest $\hat{V}$ is selected as the associated feature for the particular feature. This ensures that the model considers all features in the dataset with at least one associated feature. Based on this algorithm, unassociated features of the transformed CAN payload (Table 4) are removed using zero padding. This converts the dense array to a sparse array. As a result of this feature selection approach, it reduces the dataset size, which requires learning the benign variable pattern compared to having all associated and non-associated features. In a production environment, previous frame values of the transform CAN payload and the list of associated variables of a ID could be used to update the latest array. Therefore, it only requires storing the latest dense frames and associated feature dictionary in memory.

4.3.3. Vanilla AE

AE is a feed-forward neural network which trains to reconstruct the input as the output. Generally, the vanilla AE consists of two parts, an encoder f_ϕ and decoder g_θ. The encoder maps the input space x to a lower dimensional hidden representation known as the latent space z. The decoder does the opposite by mapping the latent space to the output space $\hat{x}$ by approximating it to the original input space x. This procedure can be formulated as follows:

$$z = f_\phi(x) \tag{4}$$

$$\hat{x} = g_\theta(f_\phi(x)) = g_\theta(z) \tag{5}$$

The objective of the AE is to train encoder f_ϕ and decoder g_θ to minimize the difference between input space x and output space $\hat{x}$ (reconstruction error). This is given by:

$$min_{\phi,\theta}||x - g_\theta(f_\phi(x))|| \tag{6}$$

where $||.||$ denotes the l_2-norm. AE-based anomaly detection assumes that benign data have a smaller reconstruction error due to the learned patterns and anomalous data have a large reconstruction error. Therefore, in vanilla AE, reconstruction error is used as the anomaly score to distinguish benign and anomalous samples. However, as mentioned before, this assumption does not always valid in practice. In some cases, AEs generalize very well that they can reconstruct anomalies. This

Algorithm 2 Associated feature selection procedure

```
Input: CAN ID list $L$, Features $X$, Threshold $\lambda$
Output: Associated feature dictionary $D$, Unassociated feature dictionary $D'$
1: Init: $D = \{ \}$
2: for $id \in L$ do
3:     Init: Feature list $F = [\ ]$
4:     Select $X_1 = [x_1, ..., x_8] \subset X, X_1 \in id$
5:     Select $X_2 = [x_9, ..., x_n] \subset X, X_2 \notin id$
6:     for $i \in X_1$ do
7:         Init: Highest associated feature $X_h = 0$
8:         for $j \in X_2$ do
9:             Create contingency table T
10:            Compute $n, k, r, \chi^2$                ▷ Using T
11:            Compute $\hat{V}$
12:            Update $X_h$
13:            if $\hat{V} > \lambda$ then
14:                $F.append[j]$
15:            end if
16:        end for
17:        if $len[F] = 0$ then
18:            $F.append[X_h]$
19:        end if
20:    end for
21:    $D[id] = F$
22:    $D'[id] = F'$                ▷ $F'$ is the complement of $F$
23: end for
```

causes to have a reconstruction error which is not significantly large enough to detect them as anomalies, leading to many false negatives.

4.3.4. Latent AE - Improved autoencoder architecture

This paper addresses the issue of overgeneralization of vanilla AE by training another small AE model in the latent space. Previous studies [23,45,46] employed the latent space to enhance anomaly detection. However, none of these studies incorporated an additional AE within the latent space, which sets our model apart. The proposed model, Latent AE includes three components: an encoder, a decoder and a latent space AE. As shown in Fig. 4, the black line indicates the training process, whereas the blue line shows the inference process. First, given an input X, the encoder obtains the latent space z. The decoder uses the latent space to reconstruct the input. Parallelly, another small AE trains to reconstruct the obtained latent space z. During training, the encoder, decoder and latent space AE parameters are adjusted to minimize the reconstruction errors through the backpropagation and gradient descent. Both AEs train to reconstruct the benign samples with low reconstruction errors E_1 and E_2. During the inference, shown in the blue line, encoded input z is used as an input to the latent space AE model to reconstruct the latent input. Reconstruction error E_2 tends to be small for the benign samples and large for the anomalous samples. If E_2 exceed a pre-defined latent threshold μ, then reconstructed latent input $\hat{z}$ uses as the input to the decoder g_θ. This enforces reconstructing the original input with a significant reconstruction error for anomalous samples. In contrast, if the E_2 below the pre-defined latent threshold μ, then the original encoded input z uses as the input to the decoder g_θ.

The anomaly detection procedure of the Latent AE is shown in Algorithm 3. This procedure is similar to the CAN GRU-based anomaly detection algorithm. Input frame x, and threshold values are specific to the Latent AE model. First, this calculates the input reconstruction error E_1 using both AEs. It then declares weak anomaly or benign status based on a pre-defined anomaly threshold ω for each frame in the observation window. Then, a pre-defined window threshold ψ and count of anomalous frames over the total number of frames use to identify the window status as benign or anomalous.

Algorithm 3 Latent AE anomaly detection

```
Input: Streaming CAN data $F$, Latent threshold $\mu$, Anomaly threshold $\omega$, Window threshold $\psi$, Time window $T$
Output: Anomaly status for each window
1: while $F$ is not empty do
2:     read $x$, time_stamp $t$, $t\_min$
3:     while $t - t\_min \le T$ do
4:         Init: Benign count $C_b = 0$ , Anomaly count $C_a = 0$
5:         $z = f_\phi(x)$
6:         $\hat{z} = f_\beta(f_\alpha(z))$
7:         Compute $E_2 = ||z - \hat{z}||$
8:         if $E_2 > \mu$ then
9:             $z \leftarrow \hat{z}$
10:        end if
11:        $\hat{x} = g_\theta(z)$
12:        Compute $E_1 = ||x - \hat{x}||$
13:        if $E_1 > \omega$ then
14:            Declare $x$ as a weak anomaly
15:            $C_a = C_a + 1$
16:        else
17:            Declare $x$ as a benign
18:            $C_b = C_b + 1$
19:        end if
20:    end while
21:    if $C_a/(C_a + C_b) > \psi$ then
22:        Return Anomaly
23:    else
24:        Return Benign
25:    end if
26:    $t\_min \leftarrow t$
27: end while
```

4.3.5. Thresholds estimation

Latent AE requires to have three thresholds. Similar to the GRU-based model threshold estimation, a separate benign dataset is used to estimate all three thresholds. The latent threshold μ is the threshold used to distinguish benign and anomalous frames in the latent space. This (μ) can be estimated considering the highest reconstruction errors E_2 for the chosen benign dataset. Similarly, the anomaly threshold ω can be estimated considering the highest reconstruction errors E_1 for the input benign data. Payload values depend on the associated IDs. Therefore, both the latent and anomaly thresholds are estimated for each CAN ID. This helps identify anomalies specific to each ID rather than considering a common threshold for all IDs. While the highest reconstruction errors are ideal as the threshold values to minimize the false positives, there might be few benign frames in the threshold estimation benign dataset which were not observed during the training phase. As a result of this, these benign frames tend to get higher reconstruction errors. Therefore, we consider the N^{th} highest quantile values as the anomaly thresholds μ, ω by allowing a very small percentage for benign anomalies. Window threshold ψ is defined in a way that it produces a minimum false positive rate for a fixed window size of time T.

CAN payload-based IDSs can effectively detect both injections and masquerade attacks as both might change the payload field patterns. However, due to the complexity of the payload field compared to the ID field, it requires selecting important payload features to achieve near real-time detection in a resource-constrained environment. This feature selection might ignore some important features as CAN data specifications are unknown.

4.4. Ensemble IDS

Ensemble models in ML combine the predictions from multiple models and improve the overall performance. Therefore, an ensemble

Table 5
Summary of thresholds used in CAN ID and payload-based models.

Model	Notation	Meaning	Description
ID-based	ω	Anomaly threshold	The trained model is used to compute the softmax probabilities for each CAN ID in a benign sample. Then calculate the Nth lowest quantile values for each CAN ID.
	ψ	Window threshold	The trained model is used to determine the anomalous or benign status of each frame in a benign dataset using the calculated ω. The average false positive rate is then computed for each time window T, and this value is set as the threshold.
Payload-based	λ	Feature selection	This is used to control the number of associated feature selections. Start with a higher association threshold, such as 0.95, and gradually reduce it until the majority of features have at least one highly associated feature.
	μ	Latent threshold	This is used to distinguish benign and anomalous frames in the latent space. The latent space AE is utilized to compute the reconstruction errors for all IDs in a benign dataset. The Nth highest quantile values are then calculated for each CAN ID.
	ω	Anomaly threshold	The reconstruction errors for all IDs are calculated using the Latent AE for a benign dataset. Subsequently, the Nth highest quantile values are calculated for each CAN ID.
	ψ	Window threshold	The trained Latent AE is used to determine the anomalous or benign status of each frame in a benign dataset using the calculated ω. The average false positive rate is then computed for each time window T, and this value is set as the threshold

IDS of CAN ID and payload-based models can be used to overcome the limitation of the individual models. For example, suppose an injection attacks change payload values with a slight deviation compared to benign values. In that case, Latent AE might not detect such changes as these will not create a significant reconstruction error. GRU model can still detect these attacks through the ID sequence change. On the other hand, a sophisticated masquerade attack might not change the ID sequences and therefore the GRU model fails to detect such attacks. In contrast, Latent AE has the capability to detect these attacks. Therefore an ensemble of these two models can increase the attack detection capability.

The proposed CAN ID-based model and payload-based models identify the benign and anomalous windows for streaming CAN data. These predictions can be used to obtain the ensemble prediction as shown in Algorithm 4. Algorithms 1 and 3 run parallelly, and the output status of both algorithms are used with an OR operator to get the prediction for the ensemble IDS.

Algorithm 4 Ensemble IDS anomaly detection

```
Input: Streaming CAN data $F$
Output: Anomaly status for each window
1: while $F$ is not empty do
2:     $output\_1 \leftarrow Algorithm$ 1
3:     $output\_3 \leftarrow Algorithm$ 3
4:     if $output\_1$ or $output\_3$ = Anomaly then
5:         Return Anomaly
6:     else
7:         Return Benign
8:     end if
9: end while
```

CAN ID-based model and payload based model use different thresholds. These are summarized in Table 5.

5. Experiments

This section presents the CAN bus data analysis, feature association, parameters of the algorithms and performance evaluation. Our experiment code is available at https://github.com/sampathrajapaksha/EnsembleIDS.git

5.1. CAN bus data analysis

We analyse CAN ID and payload field data to understand the benign traffic patterns. To this end, we use the ROAD CAN intrusion dataset [15]

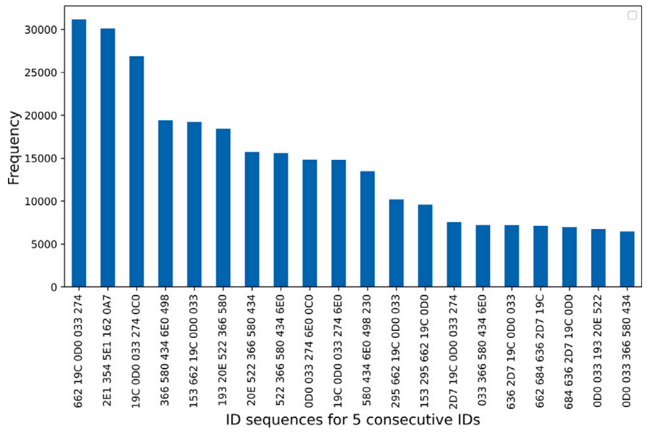


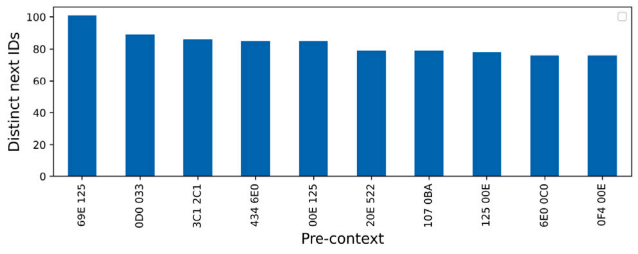
Fig. 5. The top 20 CAN ID sequences for five consecutive IDs.

5.1.1. CAN ID field

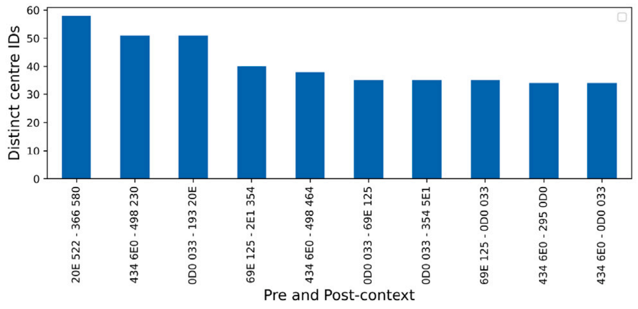
The sequential behaviour of CAN frames creates a finite set of CAN ID sequences for a fixed window size. Fig. 5 shows the frequency for the top 20 sequences for five consecutive IDs. A benign data sample of 30 min drive of the ROAD dataset is selected for this analysis. According to this, some sequences appear in high frequency, whereas others appear less in frequency. During an injection attack, this could create new CAN ID sequences that cannot be observed during benign driving periods. In these new sequences, injected CAN IDs appear in unusual contexts. This property is utilized in the GRU-based model to identify anomalous frames. However, there might be a large number of possible CAN IDs for a given context. This can be explained using Fig. 6(a), which shows the number of possible distinct CAN IDs given two pre-context IDs as the context. For example, given the context as '69E 125', 100 CAN IDs can appear as the next ID. This number can be reduced by giving the context from both sides of the ID. As shown in Fig. 6(b), given the pre-context of '69E 125' and post-context of '2E1 354', there are only 40 CAN IDs that can appear as the centre ID. Increasing the context size further reduces the number of eligible IDs in the centre. This helps achieve better predictability and detect anomalous frames with higher certainty. Therefore, CAN GRU-based model uses the pre and post-context to predict the centre ID.

5.1.2. CAN Payload field

CAN payload supports up to 64 bits of data and decoding information is proprietary. DBC files include details such as signal definition, message transmission frequency, and ECU information [25]. Based on the defined specifications, the payload field might include sensor data,



(a) The top 10 distinct next ID count for the given pre-context



(b) The top 10 distinct centre ID count for the given pre and post-context

Fig. 6. Top 10 distinct next and centre ID counts for the given context.

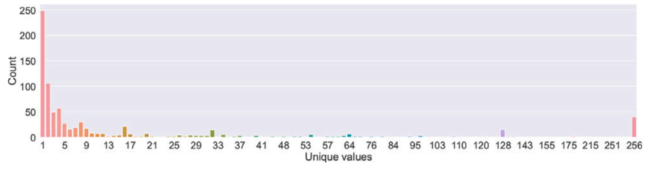
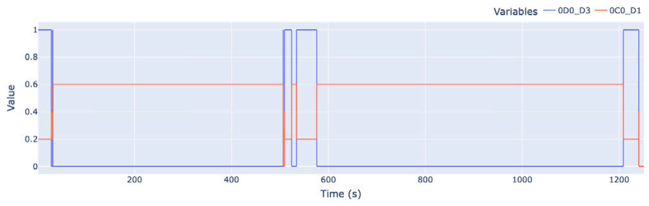
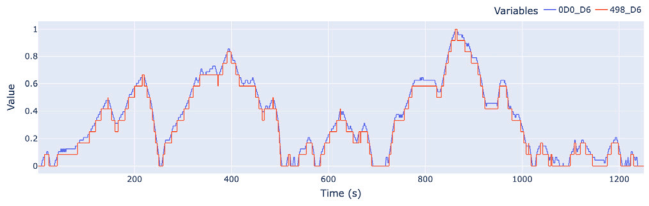


Fig. 7. Unique value distribution for ROAD dataset features.



(a) Association between 0D0_D3 and 0C0_D1



(b) Association between 0D0_D6 and 498_D6

Fig. 8. ROAD dataset variable associations. The x-axis is the time, and y-axis is the normalized variable value.

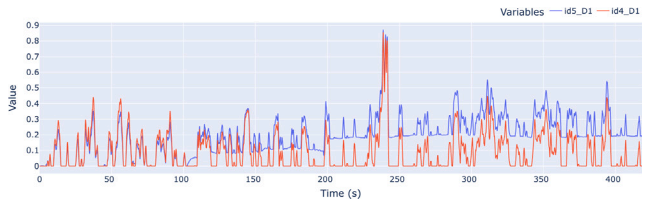


Fig. 9. Association between id5_D1 and id4_D1 in SynCAN dataset. The x-axis is the time, and y-axis is the normalized variable value.

category data, constant data or cyclical counter data [11]. Reverse engineering of CAN payload also identified the physical values, constants, and counter or CRC values [53]. In [54], the authors identified these fields as constant, multi-value and sensor values. Since the boundaries of these fields are unknown, the majority of previous works discussed in Section 3 converted 64 bits CAN payload into 8 bytes and treated each byte as a variable (feature). Each feature value ranges from 0 to 255. IDs with a shorter payload of less than 64 bits have empty features. This paper uses the same conversion and analyse the CAN payload to understand the benign traffic patterns.

The ROAD dataset includes 106 CAN IDs in which 64 bits binary to decimal conversion creates 848 (106 × 8) features. This dataset has used zero padding for empty features. The combined dataset of all the benign datasets listed in Table 1 is utilized for the analysis of payload field data. Fig. 7 shows the unique value distribution of these 848 features. Based on this distribution, 249 (29%) features are either constants or empty, whereas 321 (37%) features have unique values between 1 to 9. In contrast, 40 (0.08%) features have 256 unique values. These are the features which take every discrete value between 0 to 255. Features which are not constant or empty can be treated as either nominal or ordinal categorical features. For example, the 3rd byte of ID 0D0 communicates the signal light status. This has only two possible values of 4 and 12 to indicate the reverse light on and off status. This can be treated as a nominal categorical feature. In contrast, the 6th byte of 0D0 communicates the speedometer signal, which can take any value between 0 to 255. This might have a clear ordering of categories. Therefore, it can be considered as an ordinal categorical feature. Other features also show similar constant, ordinal and nominal variable patterns. However, without having the DBC file or the aforementioned information, it is challenging to distinguish the nominal and ordinal status of the categorical features with higher accuracy. Attackers could target any feature and therefore, the capability to detect attacks on different types of features is an important property of a CAN payload-based IDS.

5.2. Feature association

To identify the highly associated features for a particular feature, threshold λ should be selected carefully. Small λ values select too many features and make the model complex. Larger λ values only select the highly associated features and help to make the model less complex. It is important to identify at least one highly associated features for each feature in the payload. Having a higher number of associated variables for a specific variable enhances the detection capability for attacks on that variable. This is because the presence of additional associated variables can disrupt the multiple expected associations and result in higher reconstruction errors. To determine the appropriate threshold λ, We selected 0.95 as the initial number and then keep reducing it with 0.05 and at λ is 0.8 it identified at least one highly associated variables for 98% variables. For the remaining variables, we selected the highest associated variable below the threshold of 0.8. The majority of variables included two or more associated variables. While it is possible to further decrease the threshold to discover more associated variables, doing so would increase the computational cost of the IDS. Hence, we strike a balance between capturing important associations and maintaining computational efficiency. Therefore 0.8 is used as the λ for the ROAD dataset. Since the SynCAN dataset includes only 20 signals, λ is set to 0.5 based on the above method to capture more associated variables. Reverse light on and off attacks in ROAD dataset targets one bit of the third byte of ID 0D0. As mentioned in Section 5.1.2, feature 0D0_D3 can be considered as a nominal categorical feature as it has only 2 values. Algorithm 2 identifies 4 features that exhibit a high level of association with the feature 0D0_D3, with a correlation coefficient ($\hat{V}$) exceeding 0.99. Specifically, the feature 0D0_D3 shows a strong association of 0.998 with the feature 0C0_D1, which has 4 distinct values. The value change of the feature 0C0_D1 with respect to 0D0_D3 is depicted in Fig. 8(a), clearly illustrating the perfect association between these two nominal categorical features. In contrast, the Pearson correlation coefficient between 0C0_D1 and 0D0_D3 is moderate at 0.59. Furthermore, the feature 0D0_D3 exhibits a strong association of 0.999 with the feature 274_D7, which has 16 unique values. The Pearson correlation coefficient between these two variables is −0.35. Similar patterns can be observed for other nominal categorical variables as well. On the other hand, the feature 0D0_D6 displays an ordinal categorical behaviour with 63 distinct values. It demonstrates the highest association (0.998) with the feature 498_D6. Fig. 8(b) visualizes the association between these two features, while the Pearson correlation coefficient between them is 0.996.

Similarly, corrected Cramér's statistic identifies the associated features in the SynCAN dataset. Fig. 9, depicts the higher association of 0.876 $\hat{V}$ between id5_D1 and id4_D1 features. Similar types of associations can be observed for all features in both datasets. These results indicate that $\hat{V}$ is capable of identifying both ordinal and nominal categorical associations and is more appropriate than Pearson correlation to identify the associated features in the CAN payload. AEs can learn feature associations during the training phase and detect abnormal associations during the testing phase. Without knowing of the exact CAN data specifications, it is extremely difficult for an attacker to manipulate all highly associated features to keep same level of associations during the attack. Therefore, the proposed method is highly effective in identifying attacks on CAN bus.

5.3. Experimental setup

ROAD dataset. Since the dataset includes the different contexts of benign driving behaviours, first, each benign dataset listed in Table 1 except the basic short, splits into two parts: training (70%) and threshold estimation (30%) while preserving the temporal behaviour of the CAN data. Subsequently, the training splits are combined into a single dataset for model training, while the threshold estimation splits are combined into a separate dataset for anomaly and latent threshold estimation purposes. Basic short dataset is used as the benign dataset sample to estimate the window threshold. Using a good representative sample of benign data for the threshold estimation is important as the false positive and negative rate is highly dependent on the selected anomaly thresholds. Symmetric AE architecture is used for both AEs. The transformed data structure includes 655 variables after removing the empty variables. To make the model lightweight, the encoder is restricted to having only two hidden layers, including the latent layer. Grid search is used to find out the optimum nodes for hidden layers. The parameter space for the first hidden layer consists of 64, 128, 256, and 512 nodes, while the latent space parameters include 5, 10, 20, 30, 40, 50 and 60 nodes.

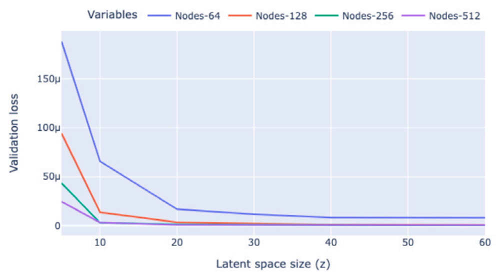
Fig. 10(a) shows the reconstruction error for the validation dataset (validation loss) for different latent sizes for a different number of hidden nodes in the first hidden layer. Validation loss rapidly decreases up to 10 latent sizes for all nodes in the first layer. After that, it slightly reduces up to 50 latent sizes depending on the number of nodes in the first layer. Generally, 10–50 yields with the lowest validation loss. A higher number of nodes in an AE can lead to overgeneralization, making it less suitable for anomaly detection tasks. Simply minimizing the validation loss does not guarantee the highest anomaly detection performance. Conversely, a simplistic AE structure may struggle to capture the variability in the data and may not be powerful enough to accurately reconstruct the inputs. Therefore, it is crucial to carefully choose the number of nodes in an AE. To help determine the optimal latent size for anomaly detection, Principal Component Analysis (PCA) can serve as a guiding factor due to certain similarities between AEs and PCA [55]. PCA aims to identify orthogonal axes that align with the directions of greatest variability in the data [55]. By analysing the PCA variance explained graph, we can identify the number of components that explain different levels of variability in the dataset. The range of 10–50 latent space size of the AE corresponds to the number of principal components (PCs) in PCA that explain the variability of the input data, ranging from 80% to 99%. In other words, if we use 10–50 principle components in PCA, then it can explain 80%–99% variability of the input data. Latent size 10 explains the 80% variability, whereas 20 and 47 explain the 90% and 99% variability, respectively. This is depicted in Fig. 10(b). Considering that 10 principal components only explain 80% of the data variability, using a latent size of 10 in the AE may not be suitable as it may fail to capture the complexity of the data. This is evident in Fig. 10(a), where a large number of nodes in the first layer is required for a latent size of 10 to achieve a small reconstruction loss. In contrast, with 20 principal components explaining 90% of the variability, using 20 nodes in the latent space of the AE only requires 128 nodes in the first layer to achieve a low validation loss. Therefore, considering the model complexity, 128 and 20 are selected as the number of nodes for hidden layers. For the latent space AE, we use a shallow network with only one hidden layer. Since this is a shallow network, the latent size is set to 99% PC variability size allowing fair reconstruction of latent input data. Based on this, the latent space AE includes 18 nodes in latent space with 20 as the input dimension. The latent space AE is much smaller compared to the vanilla AE due to the small input size. Considering the frame transmission rate of around 2000 frames per second, attack datasets split into 25-milliseconds windows to identify attack windows. This can be considered as a smaller window for near real-time prediction. The window threshold is set to 0.03 based on the lowest false positive rate (average) for the benign dataset. Additionally, latent and anomaly thresholds are calculated for each CAN ID considering the reconstruction errors for the input frames. Allowing a small margin to unseen benign data, 99.9th quantile values are considered for these thresholds.

SynCAN dataset. Same approach is used to select the parameters for the SynCAN dataset. Accordingly, 32 and 15 nodes are selected for the vanilla AE with input size of 20. The parameter space for the first hidden layer consists of 8, 32, 64, and 128 nodes, while the latent space parameters include 5, 10, 15, and 20 nodes. For the latent space AE, 11 nodes are selected for the latent layer. Unlike the real ROAD dataset, this dataset includes only 10 CAN IDs with lower transmit rates. Therefore, 100-milliseconds windows are considered with the 0.02 window threshold. This results in around 100 CAN IDs per window and requires to detect at least two anomalous frames to consider the window as anomalous. For model training, three datasets were combined and used as the training datasets. While a separate dataset was employed for threshold estimation.

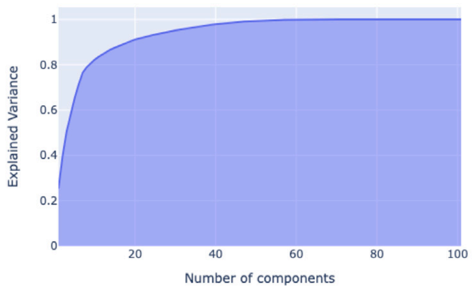
For the performance evaluation, for both datasets, we consider the window as anomalous (ground truth) if at least one frame is anomalous. All AEs train for 100 epochs with a batch size of 128. Early stopping is used to avoid overfitting. The learning rate sets to 0.0001 with the Adam optimizer. The Relu activation function is used as the activation function for all layers except the last layers. The proposed algorithm is implemented using Python 3.8 with Tensorflow and Keras library. KerasTuner is used for the grid search. All experiments run on a MacBook M1 Pro with 16 GB RAM.

5.4. Performance evaluation - CAN Payload-based Detection

We compare the proposed model Latent AE with vanilla AE and two variants of the Latent AE: Latent AE-ND (Non-Decoder) and Latent AE-NT (Non-Threshold). Latent AE-ND only uses the encoder of the vanilla AE and the latent space AE . Latent space AE's reconstruction error E_2 is used to identify anomalies. In contrast, Latent AE-NT removes the latent

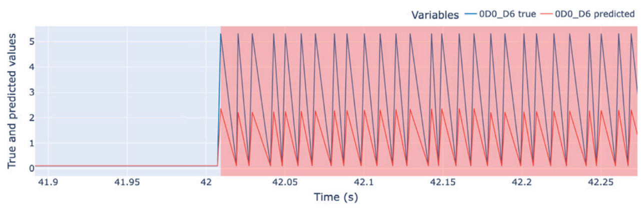


(a) Validation loss for different latent sizes

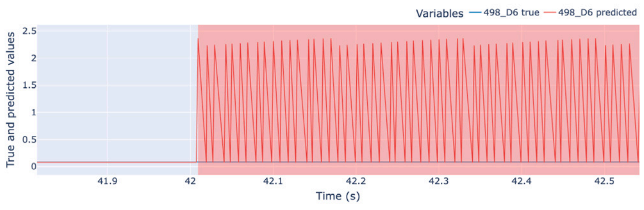


(b) Explained variances of input data for different latent space sizes

Fig. 10. Latent space size selection.

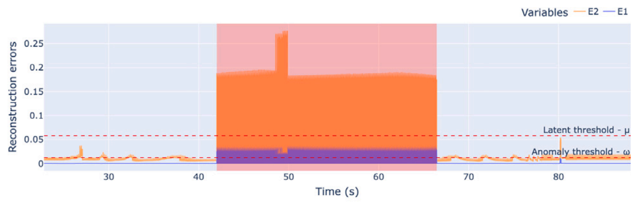


(a) 0D0_D6 true and predicted values

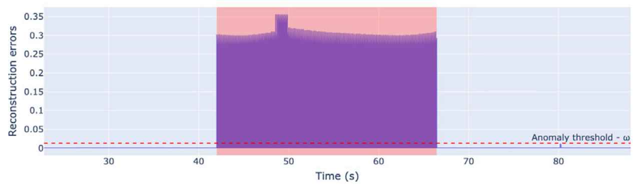


(b) 498_D6 true and predicted values

Fig. 11. Max speedometer attack true and predicted values. Shaded area represents the attack period.



(a) Reconstruction errors for vanilla and latent space AEs



(b) Reconstruction errors for Latent AE

Fig. 12. Max speedometer attack reconstruction errors. E_2 and E_1 represents latent and vanilla AE reconstruction errors respectively.

threshold μ and sends the reconstructed latent space $\hat{z}$ to the decoder of the vanilla AE. These two models are used to evaluate the effectiveness of the latent space AE. Additionally, OCSVM and a RNN-based model are used to compare Latent AE. To this end, we use INDRA [18] as the RNN-based model. This model architecture is similar to the model proposed in [19]. INDRA used a GRU network to reconstruct the input frame of size 20. This paper uses their network architecture to train one model for each CAN ID. The optimized parameter for the ROAD dataset includes sequence of 30 messages as the input frame. Optimized hyperparameters of OCSVM model are 0.0001 gamma and 0.1 nu values for the SynCAN dataset and 0.01 gamma and 0.001 nu values for the ROAD dataset. To evaluate the model performance, this paper use macro averaged F1-Score (F1), true-positive rate (TP), true-negative rate (TN), false-positive rate (FP) and false-negative rate (FN). The evaluation metrics in this approach are calculated based on the observation windows, as outlined in Algorithm 3. Specifically, the benign counts and anomaly count over these observation windows are utilized to derive the evaluation metrics. Considering that our training dataset consists solely of benign data and the test datasets are composed of attack data, employing cross-validation is not feasible in this context. Instead, we conduct multiple independent experiments, averaging the results from 10 different realizations, to ensure unbiased and reliable performance evaluation.

5.4.1. ROAD dataset attack detection

Table 6 shows the detection results of Latent AE and its variants compared to OCSVM and baseline model INDRA. Correlated signal attack targets the ID 6E0 which communicates the four wheels' speeds. This attack changes all payload values into malicious values. Therefore correlated signal attack creates both point and contextual anomalies which can easily be detected by learning the benign ranges. As expected, all models detect this attack with a higher detection rate. Latent AE and its variants achieve a 100% detection rate (TP). The same detection level is observable for the masquerade version of the attack as it only removes benign samples of the targeted ID during the attack. However, OCSVM shows a higher false positive rate for both attacks. OCSVM is highly sensitive to gamma and nu parameters and these can be used to change the decision boundaries. But we observe the higher FN and hence lower F1 score for other gamma and nu values. The large and high dimensionality of the dataset could be a reason for the comparatively low performance of OCSVM model.

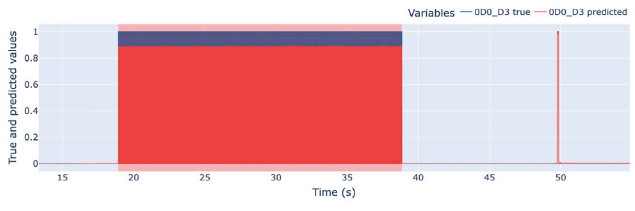
Max speedometer attack changes the 6th byte of ID 0D0 into its maximum value (255). This is a nominal categorical variable. GRU-based INDRA can learn the signal pattern and detect the sudden significant value increment as anomalous because it uses signal level thresholds. OCSVM also shows a better detection performance than correlated signal attack detection. In contrast, Latent AE and its variants can detect this attack in two aspects: significant value change and change of feature associations. This can be explained using Fig. 11 which shows a few-second snapshot of the attack dataset. During the attack period, spikes represent the attack frames whereas normal values represent benign frames. AE fails to reconstruct the spikes with the same magnitude as it has not seen these large spikes in benign training data. Therefore, variable 0D0_D6 creates a large reconstruction error as shown in Fig. 11(a). On the other hand, 0D0_D6 has a higher association with the feature 498_D6. As a result of this learned association,

Table 6
Comparison of Latent AE, Latent AE variants and baseline models detection performance of ROAD dataset.

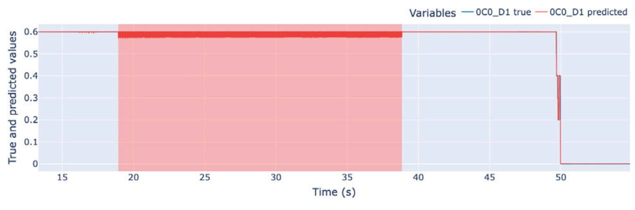
Attack	Model	F1	TP	TN	FP	FN
Correlated signal	OCSVM	89.3%	100%	67.9%	32.1%	0.0%
	INDRA	99.3%	100%	98.8%	1.2%	0.0%
	Vanilla AE	**100%**	100%	100%	0.0%	0.0%
	Latent AE-ND	**100%**	100%	100%	0.0%	0.0%
	Latent AE-NT	**100%**	100%	100%	0.0%	0.0%
	Latent AE	**100%**	100%	100%	0.0%	0.0%
Correlated signal masquerades	OCSVM	89.3%	100%	67.9%	32.1%	0.0%
	INDRA	99.3%	100%	98.8%	1.2%	0.0%
	Vanilla AE	**100%**	100%	100%	0.0%	0.0%
	Latent AE-ND	**100%**	100%	100%	0.0%	0.0%
	Latent AE-NT	**100%**	100%	100%	0.0%	0.0%
	Latent AE	**100%**	100%	100%	0.0%	0.0%
Max speedometer	OCSVM	93.5%	100%	89.6%	10.3%	0.0%
	INDRA	99.6%	100%	99.3%	0.7%	0.0%
	Vanilla AE	**100%**	100%	100%	0.0%	0.0%
	Latent AE-ND	**100%**	100%	100%	0.0%	0.0%
	Latent AE-NT	**100%**	100%	100%	0.0%	0.0%
	Latent AE	**100%**	100%	100%	0.0%	0.0%
Max speedometer masquerades	OCSVM	93.5%	100%	89.6%	10.3%	0.0%
	INDRA	99.6%	100%	99.3%	0.7%	0.0%
	Vanilla AE	**100%**	100%	100%	0.0%	0.0%
	Latent AE-ND	**100%**	100%	100%	0.0%	0.0%
	Latent AE-NT	**100%**	100%	100%	0.0%	0.0%
	Latent AE	**100%**	100%	100%	0.0%	0.0%
Reverse light on	OCSVM	33.1%	0.0%	96.2%	3.7%	100%
	INDRA	33.8%	0.0%	99.1%	0.9%	100%
	Vanilla AE	34.0%	0.0%	100%	0.0%	100%
	Latent AE-ND	**100%**	100%	100%	0.0%	0.0%
	Latent AE-NT	**100%**	100%	100%	0.0%	0.0%
	Latent AE	**100%**	100%	100%	0.0%	0.0%
Reverse light on masquerade	OCSVM	33.1%	0.0%	96.2%	3.7%	100%
	INDRA	33.8%	0.0%	99.1%	0.9%	100%
	Vanilla AE	34.0%	0.0%	100%	0.0%	100%
	Latent AE-ND	**100%**	100%	100%	0.0%	0.0%
	Latent AE-NT	**100%**	100%	100%	0.0%	0.0%
	Latent AE	**100%**	100%	100%	0.0%	0.0%
Reverse light off	OCSVM	38.1%	0.0%	97.1%	2.9%	100%
	INDRA	40.2%	0.0%	99.7%	0.3%	100%
	Vanilla AE	36.0%	0.0%	100%	0.0%	100%
	Latent AE-ND	**100%**	100%	100%	0.0%	0.0%
	Latent AE-NT	**100%**	100%	100%	0.0%	0.0%
	Latent AE	**100%**	100%	100%	0.0%	0.0%
Reverse light off masquerade	OCSVM	38.1%	0.0%	97.1%	2.9%	100%
	INDRA	40.2%	0.0%	99.7%	0.3%	100%
	Vanilla AE	36.0%	0.0%	100%	0.0%	100%
	Latent AE-ND	**100%**	100%	100%	0.0%	0.0%
	Latent AE-NT	**100%**	100%	100%	0.0%	0.0%
	Latent AE	**100%**	100%	100%	0.0%	0.0%

AE tries to reconstruct the feature 498_D6 in a way that it keeps the same level of association with the feature 0D0_D6. However, since the true value of 498_D6 does not change due to this attack, this creates a significant reconstruction error. Since vanilla AE, Latent AE, and its variants consider ID-based message level reconstruction error, collectively, these signals create significant reconstruction error which helps to detect this attack easily. Attack detection is shown in Fig. 12. As shown in Fig. 12(a), both vanilla AE and latent space AEs are capable enough to detect this attack alone. Therefore, vanilla AE, Latent AE-ND and Latent AE-NT show a 100% detection rate. Fig. 12(b) shows that Latent AE increases the anomaly reconstruction error and improves the likelihood of attack detection. Similar performance can be observed for masquerade attack as well.

Reverse light on attack targets one bit of the 3rd byte of ID 0D0. This attack turns on the reverse light while the vehicle is in the drive gear. Unlike other attacks mentioned above, reverse light on attack does not change the feature value into an unseen value as 0D0_D3 takes only two values. Hence, this attack can only be detected by identifying the mismatch of feature associations. INDRA has individual models for each CAN IDs and does not consider the associated features from other

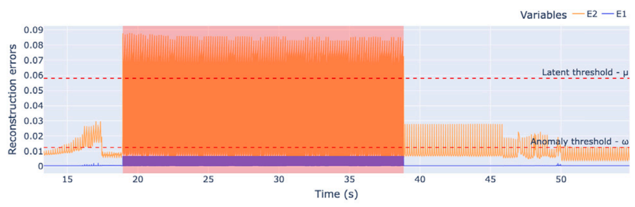


(a) 0D0_D3 true and predicted values

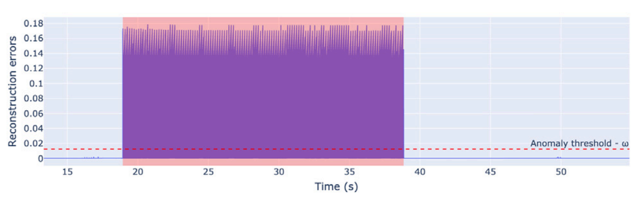


(b) 0C0_D1 true and predicted values

Fig. 13. Reverse light on attack true and predicted values.

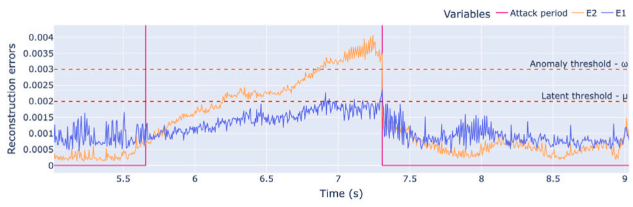


(a) Reconstruction errors for vanilla and latent space AEs

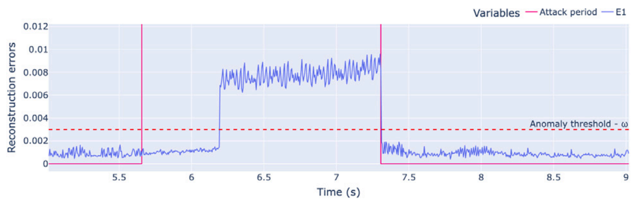


(b) Reconstruction errors for Latent AE

Fig. 14. Reverse light on attack reconstruction errors.



(a) Reconstruction errors for vanilla and latent space AEs



(b) Reconstruction errors for Latent AE

Fig. 15. Plateau attack reconstruction errors.

IDs. Therefore, it cannot exploit the feature dependencies and fails to identify such sophisticated attacks. OCSVM also has limited capability to detect these attacks. On the other hand, vanilla AE which utilized our transformed data structure, should be capable enough to detect the attack. However, this also fails to detect signal light on attack.

Fig. 13 explains the reason for this. Signal light on attack creates a very small reconstruction error for attack feature 0D0_D3 (Fig. 13(a)) and associated feature 0C0_D1 (Fig. 13(b)). A similar pattern can be observed for all other associated features of 0D0_D3. This could be due to the vanilla AE suffering from the well-known problem of overgeneralization and reconstructing anomalous data as well. We limit our model architecture to a simple architecture by using a limited number of nodes in hidden layers. 0D0_D3 and all its associated features have a limited number of unique values in the range of 2–10. Therefore, it is likely that vanilla AE still generalizes so well for features which have a limited number of unique values. We observe this behaviour for different simple and complex model architectures. In contrast, Latent AE and both of its variants detect reverse light on and masquerade attacks with 100% detection rate. Fig. 14(a) illustrates the reconstruction errors for vanilla AE and latent space AEs. vanilla AE creates a small reconstruction error that is not significant enough to detect anomalous messages. However, the latent space AE creates a large reconstruction error and thus detects the attacks. Latent AE further increases this reconstruction error due to anomalous input to the decoder of vanilla AE as shown in Fig. 14(b). This result indicates that the vanilla AE is overgeneralized during the decoding phase and these attacks can still be detected in the latent space using the latent space AE. A limited number of layers and nodes in the decoder of vanilla AE does not prevent overgeneralization as it also fails to reconstruct benign frames. During the attack periods, unassociated features reconstruct the inputs well similar to benign periods. Therefore, non of the features create FPs. Similar level of detection is achieved for the reverse light off and its masquerade attack version.

CAN payload data analysis in Section 5.1.2 shows that the majority of CAN features include only a limited number of unique values (Fig. 7). As shown above, vanilla AE fails to detect attacks on these features due to overgeneralization. Therefore, Latent AE is highly effective in detecting a wide variety of attacks on the CAN bus. Additionally, regardless of the precise overlapping with the actual payload features, Latent AE detected all attacks with 100% detection rate with the selected eight-byte features.

5.4.2. Syncan dataset attack detection

As opposed to the ROAD dataset which targeted one ID payload during a particular attack, the SynCAN dataset targets different ID payloads in different attack periods that lasted 2–3 s with various magnitudes. Furthermore, the majority of these attacks are very similar to the true signal values. These make detecting the majority of these attacks extremely difficult. Flooding and suppress attacks are simple injection attacks which change the transmission rate of IDs. Table 7 presents the results for the SynCAN dataset. OCSVM fails to identify the majority of these attacks. This might be due to the slight change of values not lie outside the decision boundaries. INDRA also shows a low F1 and detection rate for all attacks. This is mainly because INDRA uses signal-level intrusion scores and cannot utilize the feature associations. Due to this, only the targeted signal creates a small reconstruction error which might not be enough to exceed the defined threshold. Vanilla AE outperforms both OCSVM and INDRA for all attacks, exploiting the feature association and creating higher reconstruction errors for attacks. Latent AE outperforms all other models, including its variants for all attacks. This detection improvement is explained in Fig. 15, which shows a few seconds snapshot of the plateau attack. As shown in Fig. 15(a), none of the reconstruction errors of vanilla AE exceeds the anomaly threshold. However, the latent space AE detects around 70% of the attack period. Therefore, Latent AE detects the majority of the attack window (Fig. 15(b)). However, Latent AE variants do not show a promising detection for the synCAN dataset similar to ROAD dataset attack detection. This is because, for some attack periods, vanilla AE creates a higher reconstruction error, whereas, for other attack periods, latent space AE creates a higher reconstruction error. This depicts in Fig. 16. This prevents Latent AE variants from outperforming vanilla AE. These results show that the Latent AE has a better generalization capability than its variants. Even though Latent AE outperforms all models, it still misses some anomalies. This is expected because manipulated signals are almost similar to the actual signal values for some attacks and still labelled as anomalies. These might not create any anomalous status in real vehicles as they do not change any benign CAN payload data. Additionally, SynCAN labelled all frames within the attack period as anomalous even though it only includes limited anomalous frames.

Fig. 17 depicts how variable association of the SynCAN dataset helps to create higher reconstruction errors. Fig. 17(a) shows the true and predicted values of feature id3_D3 during an attack period. During this period, it targets the id3_D3 with a continuous attack which slowly drifts away from its true value. Therefore, AE fails to create this signal and predicts the value with a higher reconstruction error towards the end of the attack period. Feature id0_D0 is a highly associated feature for id3_d3. Therefore, to keep the learned association, AE reconstructs the id0_D3 by drifting away from its true value. This is shown in

Table 7
Comparison of Latent AE, Latent AE variants and baseline models detection performance of SynCAN dataset.

Attack	Model	F1	TP	TN	FP	FN
Plateau	OCSVM	57.0%	19.6%	92.4%	7.5%	80.3%
	INDRA	70.2%	39.7%	94.8%	5.1%	60.2%
	Vanilla AE	88.1%	70.5%	99.3%	0.7%	29.4%
	Latent AE-ND	88.1%	66.6%	99.7%	0.2%	33.3%
	Latent AE NT	84.3%	72.4%	94.8%	5.1%	27.5%
	Latent AE	**92.6%**	76.4%	99.3%	0.7%	23.5%
Continuous	OCSVM	53.7%	12.0%	93.6%	6.3%	87.9%
	INDRA	82.0%	56.2%	98.7%	1.2%	43.7%
	Vanilla AE	93.0%	81.7%	99.0%	0.9%	18.2%
	Latent AE-ND	91.4%	72.3%	99.8%	0.1%	27.6%
	Latent AE-NT	90.2%	85.8%	96.8%	3.2%	14.2%
	Latent AE	**95.4%**	84.1%	99.1%	0.8%	15.8%
Playback	OCSVM	49.8%	4.3%	97.0%	2.9%	95.6%
	INDRA	81.2%	48.5%	98.4%	1.6%	51.4%
	Vanilla AE	96.3%	91.7%	93.3%	0.6%	8.2%
	Latent AE-ND	95.4%	84.3%	99.8%	0.1%	15.6%
	Latent AE-NT	95.7%	93.2%	98.2%	1.7%	6.7%
	Latent AE	**98.2%**	92.5%	99.4%	0.5%	7.8%
Suppress	OCSVM	49.7%	6.6%	95.1%	4.8%	93.3%
	INDRA	74.3%	38.7%	96.3%	3.7%	61.2%
	Vanilla AE	84.3%	59.9%	99.9%	0.1%	40.0%
	Latent AE-ND	76.0%	41.3%	99.8%	0.1%	58.6%
	Latent AE-NT	85.4%	66.6%	97.5%	2.4%	33.3%
	Latent AE	**87.6%**	64.2%	99.9%	0.1%	35.7%
Flooding	OCSVM	48.8%	4.1%	96.2%	3.7%	95.8%
	INDRA	74.6%	44.2%	96.6%	3.4%	66.7%
	Vanilla AE	90.0%	74.5%	99.9%	0.1%	25.4%
	Latent AE-ND	87.6%	62.8%	99.8%	0.1%	37.1%
	Latent AE-NT	89.0%	77.3%	97.2%	2.7%	22.6%
	Latent AE	**91.3%**	75.9%	99.9%	0.1%	24.0%

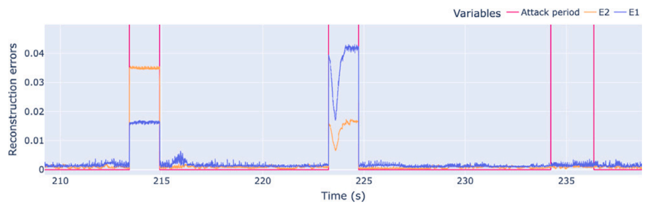


Fig. 16. SynCAN reconstruction errors.

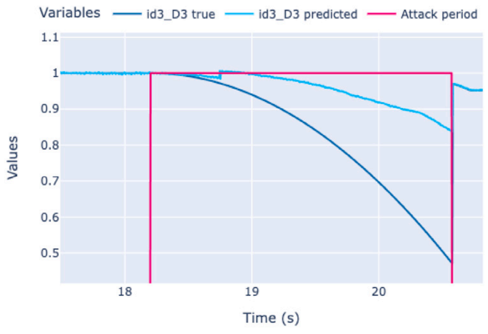
Fig. 17(b). Both of these errors aid in creating a higher reconstruction error at the message level. Moreover, this attack shows the anomalous value similarity to the true signal value at the beginning of the attack period which causes the higher FN rate for point level and small window-level attack detection. These experimental results show the effectiveness of the proposed feature selection, transformed data structure and improved AE, Latent AE for attack detection in the CAN bus.

5.5. Performance evaluation - Ensemble IDS

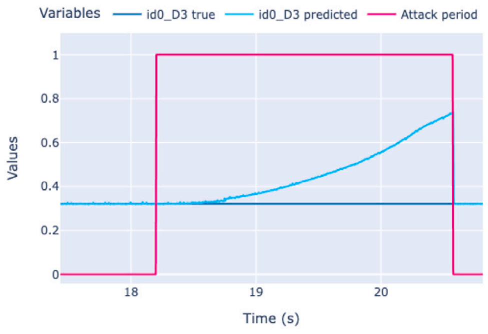
The ensemble IDS integrates GRU model and Latent AE model to improve the overall attack detection. GRU model used the same experimental settings which used in [12] with 25-milliseconds windows to align with the Latent AE model. Same model parameters is used for the SynCAN dataset with 100-milliseconds windows.

5.5.1. ROAD and SynCAN Attack Detection

Table 8 shows the detection performance of the ensemble IDS compared to GRU and Latent AE models for the ROAD dataset attacks. GRU model fails to detect the correlated signal and correlated signal masquerade attacks with a high detection rate. This is likely because this attack targets the second most frequent ID, which has a slightly random transmission rate compared to other IDs. Therefore, it creates



(a) id3_D3 true and predicted values



(b) id0_D3 true and predicted values

Fig. 17. SynCAN feature association.

Table 8
Comparison of GRU, Latent AE and Ensemble IDS detection performance of ROAD dataset.

Attack	Model	F1	TP	TN	FP	FN
Correlated signal	GRU	82.1%	83.8%	100%	0.0%	16.2%
	Latent AE	100%	100%	100%	0.0%	0.0%
	Ensemble IDS	100%	100%	100%	0.0%	0.0%
Correlated signal masquerades	GRU	86.7%	85.6%	100%	0.0%	14.4%
	Latent AE	100%	100%	100%	0.0%	0.0%
	Ensemble IDS	100%	100%	100%	0.0%	0.0%
Max speedometer	GRU	100%	100%	100%	0.0%	0.0%
	Latent AE	100%	100%	100%	0.0%	0.0%
	Ensemble IDS	100%	100%	100%	0.0%	0.0%
Max speedometer masquerades	GRU	100%	100%	100%	0.0%	0.0%
	Latent AE	100%	100%	100%	0.0%	0.0%
	Ensemble IDS	100%	100%	100%	0.0%	0.0%
Reverse light on	GRU	99.1%	99.0%	100%	0.0%	1.0%
	Latent AE	100%	100%	100%	0.0%	0.0%
	Ensemble IDS	100%	100%	100%	0.0%	0.0%
Reverse light on masquerade	GRU	99.4%	99.3%	100%	0.0%	0.7%
	Latent AE	100%	100%	100%	0.0%	0.0%
	Ensemble IDS	100%	100%	100%	0.0%	0.0%
Reverse light off	GRU	100%	100%	100%	0.0%	0.0%
	Latent AE	100%	100%	100%	0.0%	0.0%
	Ensemble IDS	100%	100%	100%	0.0%	0.0%
Reverse light off masquerade	GRU	100%	100%	100%	0.0%	0.0%
	Latent AE	100%	100%	100%	0.0%	0.0%
	Ensemble IDS	100%	100%	100%	0.0%	0.0%

Table 9
Comparison of GRU, Latent AE and Ensemble IDS detection performance of SynCAN dataset.

Attack	Model	F1	TP	TN	FP	FN
Plateau	GRU	46.0%	0.0%	100%	0.0%	100%
	Latent AE	92.6%	76.4%	99.3%	0.7%	23.5%
	Ensemble IDS	92.6%	76.4%	99.3%	0.7%	23.5%
Continuous	GRU	47.1%	0.0%	100.0%	0.0%	100%
	Latent AE	95.4%	84.1%	99.1%	0.8%	15.8%
	Ensemble IDS	95.4%	84.1%	99.1%	0.8%	15.8%
Playback	GRU	47.4%	0.0%	100%	0.0%	100%
	Latent AE	98.2%	92.5%	99.4%	0.5%	7.8%
	Ensemble IDS	98.2%	92.5%	99.4%	0.5%	7.8%
Suppress	GRU	99.9%	99.9%	99.9%	0.1%	0.0%
	Latent AE	87.6%	64.2%	99.9%	0.1%	35.7%
	Ensemble IDS	99.9%	100%	99.8%	0.1%	0.0%
Flooding	GRU	100%	100%	100%	0.0%	0.0%
	Latent AE	91.3%	75.9%	99.9%	0.1%	24.0%
	Ensemble IDS	99.9%	100%	99.9%	0.1%	0.0%

more sequences, which results in more valid sequences being created, even for attack frames. However, Latent AE detects all attacks with 100% detection rate. Since none of the individual models produces false positives, the ensemble model of GRU and Latent AE achieves the best performance of the Latent AE model.

Table 9 shows the detection performance of the ensemble IDS compared to GRU and Latent AE models for the SynCAN dataset attacks. Unlike ROAD attacks, masquerade attacks of the SynCAN dataset do not change the CAN ID sequences. Thus, as expected, the GRU model fails to detect these attacks. Flooding and suppress attacks change the CAN ID sequences due to frame injections and frame suspension. As a result of this change, the GRU model detects these two attacks with 100% detection rate (TP). Latent AE fails to achieve a high detection rate for these two attacks. This might be due to very small changes to signal values which do not create significant reconstruction errors. The ensemble IDS achieves the performance of the best individual model for all attacks. It is impotent to set anomaly and window thresholds to optimal values using benign datasets to minimize false positives. This is important to outperform the individual models.

These results on ROAD and SynCAN attack datasets show the ensemble IDS's effectiveness in detecting a wide range of attacks with a higher detection rate. Ensemble IDS increases the overall attack detection while decreasing the weakness of individual models.

5.5.2. *Comparison with baseline models*

We compare the proposed ensemble IDS with two baseline models, namely INDRA [18], which is RNN-based, and CANShield [39], which is a deep AE-based model designed with prior knowledge of the CAN specification. However, due to the lack of details to reproduce CANShield, we relied on comparing our results using the area under the curve (AUC) score reported in their paper. In the case of the ROAD dataset, masquerade attacks were not discussed in their results, so we did not include a comparison for those attacks. Based on the results, for the ROAD dataset, both the Ensemble IDS and CANShield successfully detected all attacks, except for the reverse light off attack, where CANShield achieved a slightly lower AUC score of 0.99. For the SynCAN dataset, CANShield showed higher detection rates for the plateau attack, while the ensemble IDS outperformed CANShield for other attacks, except for the flooding attack, where both models exhibited the same level of detection. INDRA, did not perform well for either dataset, as it lacked the capability to detect contextual anomalies. This comparison is presented in Table 10.

5.5.3. *Model implementation on Raspberry Pi*

To analyse the overhead in a computationally constrained environment for the ensemble IDS, we deployed the GRU and Latent AE models on a Raspberry Pi 4. The Raspberry Pi is a small single-board

Table 10
Comparison with baseline models — AUC score.

Dataset	Attack	Ensemble IDS	CANShield	INDRA
ROAD	Correlated signal	**1.00**	**1.00**	0.98
	Max speedometer	**1.00**	**1.00**	0.99
	Reverse light on	**1.00**	**1.00**	0.67
	Reverse light off	**1.00**	**0.99**	0.74
SynCAN	Plateau	0.93	**0.96**	0.72
	Continuous	**0.93**	0.87	0.79
	Playback	**0.97**	0.94	0.84
	Suppress	**0.99**	0.98	0.75
	Flooding	**0.99**	**0.99**	0.77

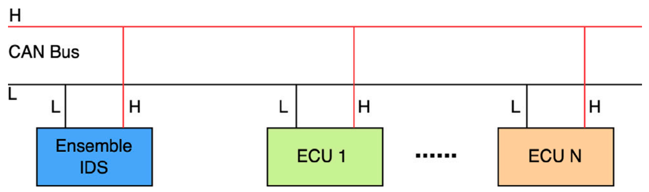


Fig. 18. The proposed ensemble IDS deployment in the CAN bus.

computer with limited memory and processing power. For this purpose, we used the Raspberry Pi 4 Model B 8 GB version, along with a 16 GB micro SD card. In order to optimize for on-device machine learning, we converted the trained GRU and Latent AE models into TensorFlow Lite (TFLite) versions. During this conversion, we also applied quantization to reduce detection latency. Quantization is an optimization technique that reduces the precision of model parameters by using 16 or 8-bit numbers instead of the default 32-bit floating-point numbers. We opted for 16-bit quantization as 8-bit quantization slightly reduced accuracy in both models. The converted TFLite models were named GRU-TFLite and Latent AE-TFLite. This conversion significantly reduced the model size, with the Latent AE model going from 1145 KB to 745 KB, and the GRU model decreasing from 233 KB to 49 KB. These TFLite models were then deployed on the Raspberry Pi for overhead analysis.

The Raspberry Pi is capable of running both the Latent AE-TFLite and GRU-TFLite lightweight models in parallel. However, when running them simultaneously, each model incurs a small additional time compared to running only one model. During the inference process, streaming CAN data is stored in a buffer, as streaming CAN data is faster than data preprocessing. For ID-based model data preprocessing, we use the Python double-ended queue (deque) due to its efficiency in data append and pop operations. On the other hand, the payload-based model utilizes NumPy arrays to optimize data preprocessing. The ID-based model requires less time for data preprocessing and inference compared to the payload-based model. Since both models use the same window size T, the Ensemble IDS predicts the window status by considering the predictions from both the ID-based and payload-based models. The Raspberry Pi device can be integrated into the CAN bus through the OBD2-II or central gateways, acting as an additional ECU as depicted in Fig. 18. This enables continuous monitoring of CAN messages. This deployment strategy preserve the privacy of CAN data since the data is not shared with a server for making predictions.

5.5.4. Overhead analysis

In addition to the detection rate, detection latency and memory are also critical aspects of a CAN IDS. Table 11 shows the comparison of payload-based and ID-based models for the number of trainable model parameters, model size (KB) and average inference time (ms). ROAD dataset is used for this analysis. As mentioned earlier, these are evaluated on a MacBook M1 Pro with 16 GB RAM. The number of parameters and model sizes are presented for one model.

In the case of the INDRA model, each CAN ID requires a separate model, resulting in the largest memory requirement and the highest inference time. Consequently, it is not suitable for a CAN IDS. Among the payload-based models, Latent AE-ND demonstrates the best performance in terms of memory and inference time. This is achieved by removing the decoder component from the vanilla AE model. Latent AE only requires an additional 0.05 ms for prediction compared to the vanilla AE, with an average inference time of 0.18 ms per frame. The Latent space AE model is much smaller than the vanilla AE due to its limited number of input features and shallow model architecture. On the other hand, the GRU model is significantly smaller and more efficient than the payload-based models, as it solely utilizes the CAN ID as the input feature.

Table 12 presents the average inference overhead, including CPU utilization, memory usage (RAM), and inference time, for the deployed TFLite models on Raspberry Pi. The ensemble IDS utilizes 82% of the CPU and 94MB of RAM, with an inference time of 0.5 ms per CAN frame. This is a small additional time compared to the Latent AE-TFLite model. However, despite this slight increase in inference time, the ensemble IDS remains a practical and deployable in-vehicle IDS due to its comprehensive attack detection capabilities. The outputs of the TFLite models on Raspberry Pi are comparable to the values obtained from TensorFlow models, with negligible accuracy differences. Therefore, the deployed models on Raspberry Pi exhibit the same detection capabilities without any loss in accuracy. Overall, a 25 ms window includes around 50 CAN frames in which ensemble IDS takes only 25 ms to give the window prediction. Typically, the average driver's response time ranges from 0.7 s to 1.5 s [56]. Thus, the 25 ms detection time is minimal compared to the driver's response time. This enables the driver or the vehicle itself to take appropriate countermeasures in near real-time. Additionally, during a CAN bus attack, continuous message injections are required to overwrite legitimate CAN frames [26]. This increases the likelihood of detecting malicious frames even before the attack has a significant impact. Therefore the proposed ensemble IDS is suitable for detecting wide range of attacks on the CAN bus in near real-time.

5.6. Limitations

Despite the superior attack detection capabilities exhibited by the ensemble IDS, there are certain limitations associated with both the GRU-based and payload-based models. Both the GRU-based and payload-based models require a large benign dataset to effectively capture a diverse range of benign driving behaviours to minimize the occurrence of unseen data. For the payload-based model, each variable must be normalized based on the observed minimum and maximum values. However, if the training dataset does not include the true minimum and maximum values for each variable, any value falling below the minimum or above the maximum during the inference stage could potentially result in false positives. This limitation can be minimized by utilizing a large and diverse training dataset. Nevertheless, there may still exist some variable values such as engine temperature that do not reach the maximum values during normal driving conditions.

A potential limitation of black-box approach-based models, compared to signal value-based models, is that the payload features may not precisely align with the signal values. Depending on the CAN data

Table 11
Average detection latency and memory requirement.

Features	Model	Parameters	Model size (KB)	Inference time (ms)
Payload	INDRA	190728	3200	0.44
	Vanilla AE	173731	882	0.13
	Latent AE-ND	87306	697	0.11
	Latent AE-NT	174489	1145	0.18
	Latent AE	174489	1145	0.18
ID	GRU	16945	233	0.08

Table 12
Average inference overhead on Raspberry Pi.

Model	CPU (%)	Memory (MB)	Inference time (ms)
Latent AE-TFLite	38	58	0.4
GRU-TFLite	27	3	0.2
Ensemble IDS	82	94	0.5

specification for a particular vehicle model, signal values can span multiple bytes with different byte ordering or may be represented by a single bit. This means that a payload feature could include multiple signal values or multiple payload features may represent a single signal value. However, the Latent AE approach is designed to learn feature association patterns for benign data and detect deviations from these patterns as anomalies. Even with byte-level feature splitting, the majority of these patterns can still be learned using Latent AE. Attacks on the CAN payload disrupt these learned patterns, leading to high reconstruction error that can be detected. Therefore, if a payload feature has at least one highly associated feature, the impact of not precisely selecting the actual signal boundaries is minimized in the proposed method for attack detection. In other words, even if a payload feature includes multiple signal values, as long as it has a highly associated feature, the method can identify the mismatch in association during an attack. Similarly, if a signal is represented by multiple payload features and any of these features has a highly associated feature, the method can detect the association mismatch. For the ROAD dataset, 98% of the features have at least one highly associated feature. However, if a payload feature does not have a highly associated feature, it may result in false negatives for the respective signals.

6. Conclusion

The CAN is the most widely used in-vehicle network due to several benefits such as low cost and simplicity. Despite these benefits, CAN bus is vulnerable to cyberattacks that directly affect vehicle passengers' safety. Therefore, there is a clear need to implement an effective detection mechanism against these attacks. IDSs that utilize only the CAN ID field effectively detect injection attacks. However, time-series CAN payload data are critical for detecting advanced attacks such as masquerade attacks. Therefore, identifying a wide variety of attacks on the CAN bus is challenging and requires an IDS that employs multiple methods to cover a wide range of attacks with limited computing resources.

Thus, we propose an ensemble IDS by integrating a GRU-based model and a novel AE model. The GRU-based lightweight model uses the CAN ID field, whereas the AE model uses the CAN payload field to detect attacks. Latent AE, an improved AE-based model uses a novel feature selection method based on Cramér's $\hat{V}$ statistics and a transformed CAN payload data structure to handle complex CAN data. Latent AE addresses the issue of high false negatives of vanilla AEs due to overgeneralization by introducing a small latent space AE. CAN bus payload includes a higher number of categorical features with a limited number of unique values. Therefore, CAN payload-based vanilla AEs are prone to overgeneralization and miss detecting attacks on those variables. Results from the experiment show the effectiveness of Latent AE for detecting sophisticated attacks on CAN payload in near real-time by overcoming the limitation of vanilla AEs. Experiment results show that the ensemble IDS improves attack detection while decreasing the weakness of individual models. The inference overhead of the proposed model is minimum and therefore, suitable to deploy in a real vehicle to detect various attacks in near real-time. For future works, we plan to deploy the ensemble IDS in a real vehicle and evaluate it with real attacks with the help of project's industrial partner, HORIBA MIRA Ltd on their proving grounds.

CRediT authorship contribution statement

Sampath Rajapaksha: Conceptualization, Methodology, Software, Data curation, Writing – original draft. **Harsha Kalutarage:** Conceptualization, Supervision, Writing – review & editing. **M. Omar Al-Kadri:** Conceptualization, Supervision, Writing – review & editing. **Andrei Petrovski:** Conceptualization, Supervision, Writing – review & editing. **Garikayi Madzudzo:** Conceptualization, Supervision, Writing – review & editing.

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

Data will be made available on request.

Acknowledgment

This work was supported by the School of Computing, Robert Gordon University and HORIBA MIRA Limited.

References

[1] Xun Y, Zhao Y, Liu J. Vehicleeids: A novel external intrusion detection system based on vehicle voltage signals. IEEE Internet Things J 2021.
[2] Bi Z, Xu G, Xu G, Wang C, Zhang S. Bit-level automotive controller area network message reverse framework based on linear regression. Sensors 2022;22(3):981.
[3] Miller C, Valasek C. Adventures in automotive networks and control units. Def Con 2013;21(260–264):15–31.
[4] Checkoway S, McCoy D, Kantor B, Anderson D, Shacham H, Savage S, et al. Comprehensive experimental analyses of automotive attack surfaces. In: 20th USENIX security symposium. 2011.
[5] NasrEldin A, Bahaa-Eldin AM, Sobh MA. In-vehicle intrusion detection based on deep learning attention technique. In: 2021 16th international conference on computer engineering and systems. IEEE; 2021, p. 1–7.
[6] Miller C, Valasek C. Remote exploitation of an unaltered passenger vehicle. Black Hat USA 2015;2015(S 91).
[7] Rajapaksha S, Kalutarage H, Al-Kadri M, Petrovski A, Madzudzo G, Cheah M. AI-based intrusion detection systems for in-vehicle networks: A survey. ACM Comput Surv 2023;55(11). http://dx.doi.org/10.1145/3570954
[8] Müter M, Asaj N. Entropy-based anomaly detection for in-vehicle networks. In: 2011 IEEE intelligent vehicles symposium. IEEE; 2011, p. 1110–5.
[9] Aliwa E, Rana O, Perera C, Burnap P. Cyberattacks and countermeasures for in-vehicle networks. ACM Comput Surv 2021;54(1):1–37.
[10] Rajapaksha S, Kalutarage H, Al-Kadri MO, Petrovski A, Madzudzo G, Cheah M. AI-based intrusion detection systems for in-vehicle networks: A survey. ACM Comput Surv 2023;55(11). http://dx.doi.org/10.1145/3570954.

[11] Tomlinson A, Bryans J, Shaikh SA. Using internal context to detect automotive controller area network attacks. Comput Electr Eng 2021;91:107048.

[12] Rajapaksha S, Kalutarage H, Al-Kadri MO, Madzudzo G, Petrovski AV. Keep the moving vehicle secure: Context-aware intrusion detection system for in-vehicle CAN bus security. In: 2022 14th international conference on cyber conflict: Keep moving! vol. 700. IEEE; 2022, p. 309–30.

[13] Rajapaksha S, Kalutarage H, Al-Kadri MO, Petrovski A, Madzudzo G. Improving in-vehicle networks intrusion detection using on-device transfer learning. In: Symposium on vehicles security and privacy. 2023, http://dx.doi.org/10.14722/vehiclesec.2023.23088.

[14] Moriano P, Bridges RA, Iannacone MD. Detecting CAN masquerade attacks with signal clustering similarity. 2022, arXiv preprint arXiv:2201.02665.

[15] Verma ME, Iannacone MD, Bridges RA, Hollifield SC, Kay B, Combs FL. Road: The real ornl automotive dynamometer controller area network intrusion detection dataset (with a comprehensive can ids dataset survey & guide). 2020, arXiv preprint arXiv:2012.14600.

[16] Stabili D, Marchetti M, Colajanni M. Detecting attacks to internal vehicle networks through hamming distance. In: 2017 AEIT international annual conference. IEEE; 2017, p. 1–6.

[17] Number of automotive ECUs continues to rise. 2022, https://www.eenewsautomotive.com/en/number-of-automotive-ecus-continues-to-rise/. [Accessed 08 August 2022].

[18] Kukkala VK, Thiruloga SV, Pasricha S. Indra: Intrusion detection using recurrent autoencoders in automotive embedded systems. IEEE Trans Comput-Aided Des Integr Circuits Syst 2020;39(11):3698–710.

[19] Longari S, Valcarcel DHN, Zago M, Carminati M, Zanero S. CANnolo: An anomaly detection system based on LSTM autoencoders for controller area network. IEEE Trans Netw Serv Manag 2020;18(2):1913–24.

[20] Zhou W, Fu H, Kapoor S. CANGuard: Practical intrusion detection for in-vehicle network via unsupervised learning. In: 2021 IEEE/ACM symposium on edge computing. 2021, p. 454–8. http://dx.doi.org/10.1145/3453142.3493514.

[21] Lokman SF, Othman AT, Musa S, Abu Bakar MH. Deep contractive autoencoder-based anomaly detection for in-vehicle controller area network (CAN). In: Progress in engineering technology: Automotive, energy generation, quality control and efficiency. Springer; 2019, p. 195–205.

[22] Novikova E, Le V, Yutin M, Weber M, Anderson C. Autoencoder anomaly detection on large CAN bus data. In: Proceedings of DLP-KDD. 2020.

[23] Gong D, Liu L, Le V, Saha B, Mansour MR, Venkatesh S, et al. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In: Proceedings of the IEEE/CVF international conference on computer vision. 2019, p. 1705–14.

[24] Bergsma W. A bias-correction for Cramér's V and Tschuprow's T. J Korean Stat Soc 2013;42(3):323–8. http://dx.doi.org/10.1016/j.jkss.2012.10.002, Available: https://www.sciencedirect.com/science/article/pii/S1226319212001032.

[25] Verma M, Bridges R, Hollifield S. ACTT: Automotive CAN tokenization and translation. In: 2018 international conference on computational science and computational intelligence. IEEE; 2018, p. 278–83.

[26] Miller C, Valasek C. Can message injection: Og dynamite edition. Tech. rep., 2016.

[27] Suda H, Natsui M, Hanyu T. Systematic intrusion detection technique for an in-vehicle network based on time-series feature extraction. In: 2018 IEEE 48th international symposium on multiple-valued logic. IEEE; 2018, p. 56–61.

[28] Dürrwang J, Braun M, Kriesten R, Pretschner A. Enhancement of automotive penetration testing with threat analyses results. SAE International Journal of Transportation Cybersecurity and Privacy 2018 1(11-01-02-0005):91–112.

[29] Cai Z, Wang A, Zhang W, Gruffke M, Schweppe H. 0-days & mitigations: roadways to exploit and secure connected BMW cars. Black Hat USA 2019;2019:39.

[30] Desta AK, Ohira S, Arai I, Fujikawa K. ID sequence analysis for intrusion detection in the can bus using long short term memory networks. In: 2020 IEEE international conference on pervasive computing and communications workshops. IEEE; 2020, p. 1–6.

[31] Hoang T-N, Kim D. Detecting in-vehicle intrusion via semi-supervised learning-based convolutional adversarial autoencoders. Veh Commun 2022;38:100520. http://dx.doi.org/10.1016/j.vehcom.2022.100520, Available: https://www.sciencedirect.com/science/article/pii/S2214209622000675.

[32] Kalutarage HK, Al-Kadri MO, Cheah M, Madzudzo G. Context-aware anomaly detector for monitoring cyber attacks on automotive CAN bus. In: ACM computer science in cars symposium. 2019, p. 1–8.

[33] Tomlinson A, Bryans J, Shaikh SA. Using a one-class compound classifier to detect in-vehicle network attacks. In: Proceedings of the genetic and evolutionary computation conference companion. 2018, p. 1926–9.

[34] Taylor A, Leblanc S, Japkowicz N. Anomaly detection in automobile control network data with long short-term memory networks. IEEE; 2016, p. 130–9.

[35] Tanksale V. Anomaly detection for controller area networks using long short-term memory. IEEE Open J Intell Transp Syst 2020;1:253–65.

[36] Hanselmann M, Strauss T, Dormann K, Ulmer H. Canet: An unsupervised intrusion detection system for high dimensional CAN bus data. IEEE Access 2020;8:58194–205.

[37] Kukkala VK, Thiruloga SV, Pasricha S. LATTE: L STM self-attention based anomaly detection in embedded automotive platforms. ACM Trans Embed Comput Syst (TECS) 2021;20(5s):1–23.

[38] Thiruloga SV, Kukkala VK, Pasricha S. TENET: Temporal CNN with attention for anomaly detection in automotive cyber-physical systems. In: 2022 27th Asia and South Pacific design automation conference. IEEE; 2022, p. 326–31.

[39] Shahriar MH, Xiao Y, Moriano P, Lou W, Hou YT. CANShield: Signal-based intrusion detection for controller area networks. 2022, arXiv preprint arXiv:2205.01306.

[40] Balaji P, Ghaderi M. NeuroCAN: Contextual anomaly detection in controller area networks. In: 2021 IEEE international smart cities conference. IEEE; 2021, p. 1–7.

[41] Verma ME, Iannacone MD, Bridges RA, Hollifield SC, Moriano P, Kay B, et al. Addressing the lack of comparability & testing in CAN intrusion detection research: A comprehensive guide to CAN ids data & introduction of the ROAD dataset. 2020, arXiv preprint arXiv:2012.14600.

[42] Yin C, Zhang S, Wang J, Xiong NN. Anomaly detection based on convolutional recurrent autoencoder for IoT time series. IEEE Trans Syst Man Cybern 2020;52(1):112–22.

[43] Cheng Z, Wang S, Zhang P, Wang S, Liu X, Zhu E. Improved autoencoder for unsupervised anomaly detection. Int J Intell Syst 2021;36(12):7103–25.

[44] Park H, Noh J, Ham B. Learning memory-guided normality for anomaly detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020, p. 14372–81.

[45] Angiulli F, Fassetti F, Ferragina L. LatentOut: an unsupervised deep anomaly detection approach exploiting latent space distribution. Mach Learn 2022;1–27.

[46] ElMorshedy MM, Fathalla R, El-Sonbaty Y. Feature transformation framework for enhancing compactness and separability of data points in feature space for small datasets. Appl Sci 2022;12(3):1713.

[47] Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. 2013, arXiv preprint arXiv:1301.3781.

[48] Ganesan A, Rao J, Shin K. Exploiting consistency among heterogeneous sensors for vehicle anomaly detection. Tech. rep., SAE Technical Paper; 2017.

[49] Li H, Zhao L, Juliato M, Ahmed S, Sastry MR, Yang LL. Poster: Intrusion detection system for in-vehicle networks using sensor correlation and integration. In: Proceedings of the 2017 ACM SIGSAC conference on computer and communications security. 2017, p. 2531–3.

[50] Acock AC, Stavig GR. A measure of association for nonparametric statistics. Soc Forces 1979;57(4):1381–6.

[51] Bergsma W. A bias-correction for Cramér's V and Tschuprow's T. J Korean Stat Soc 2013;42(3):323–8.

[52] Akoglu H. User's guide to correlation coefficients. Turk J Emerg Med 2018;18(3):91–3.

[53] Marchetti M, Stabili D. READ: Reverse engineering of automotive data frames. IEEE Trans Inf Forensics Secur 2018;14(4):1083–97.

[54] Markovitz M, Wool A. Field classification, modeling and anomaly detection in unknown CAN bus networks. Veh Commun 2017;9:43–52.

[55] Ladjal S, Newson A, Pham C-H. A PCA-like autoencoder. 2019, arXiv preprint arXiv:1904.01277.

[56] Droździel P, Tarkowski S, Rybicka I, Wrona R. Drivers 'reaction time research in the conditions in the real traffic. Open Eng 2020;10(1):35–47. http://dx.doi.org/10.1515/eng-2020-0004.