CANdito: Improving Payload-based Detection of Attacks on Controller Area Networks

Stefano Longari[1], Carlo Alberto Pozzoli[1], Alessandro Nichelini[1], Michele Carminati[1], and Stefano Zanero[1]

Politecnico di Milano, Milan, Italy
{stefano.longari,michele.carminati,stefano.zanero}@polimi.it,
{carloalberto.pozzoli,alessandro.nichelini}@mail.polimi.it

Abstract Over the years, the increasingly complex and interconnected vehicles raised the need for effective and efficient Intrusion Detection Systems against on-board networks. In light of the stringent domain requirements and the heterogeneity of information transmitted on the Controller Area Network, multiple approaches have been proposed, which work at different abstraction levels and granularities. Among these, RNN-based solutions received the attention of the research community for their performances and promising results. This paper proposes CANdito, an unsupervised IDS that exploits Long Short-Term Memory autoencoders to detect anomalies through a signal reconstruction process. In particular, we improve an RNN-based state-of-the-art IDS for CAN from the detection and temporal performances to comply with the strict automotive domain requirements. We evaluate CANdito by comparing its performance against state-of-the-art Intrusion Detection Systems (IDSs) for in-vehicle network and a comprehensive set of synthetic and real attacks in real-world CAN datasets.

1 Introduction

In the last decades, vehicles have become more complex, especially concerning their electronics [15]. Car manufacturers nowadays implement entertainment and autonomous drive-related technologies. As a result, the number of Electronic Control Units (ECUs) grew to reach more than one hundred units in the most complex vehicles. This evergrowing complexity, however, raises security risks, as firstly demonstrated by Koscher and Checkoway in [3, 11], allowing the attacker to gain control of the vehicle's functionalities, even remotely. To manage such risks, the scientific research community focuses on developing countermeasures and security solutions, amongst which intrusion detection techniques for Controller Area Network (CAN) [1] have proven effective.

CAN security weaknesses are nowadays well known and discussed in multiple works [2, 36]. As demonstrated by Miller and Valasek in [22, 23], one of the most common known vulnerabilities derives from the lack of authentication of

[1]For further details on the CAN specification, we refer the reader to [25].

messages on CAN. A node should not be allowed to send IDs that it does not own, but there is no mechanism to enforce this rule. Therefore, an attacker that takes control of an ECU that has access to a CAN bus can ideally send any ID and payload. In worst-case scenarios, the attacker is also capable of silencing the owner of the packet to avoid conflicts, as presented in [6,16]. Given the IDS nature of the work at hand, we present the capabilities of the attacker through the effects of its actions on the payload and flow of packets on the bus: A weak attacker may **inject** forged packets with one or multiple specific IDs on the bus, without silencing their owner. In this situation, the receivers may or may not consider the attacker's packets valid due to the incongruities on the bus. To solve this conflict, a stronger attacker may silence the owner and then forge packets with its ID, leading to a **masquerade** attack. Such masquerade attack can be implemented with or without consideration of the existence of an IDS checking the bus. If it is considered, the attacker may want to implement a **replay** attack, where she/he does not create a new payload but repeats a payload previously captured on the bus, or a **seamless change** attack, where the attacker drives a signal from its current value to a tampered one by changing it slowly through multiple packets. If it is not considered, an attacker may want to study the effects of various payloads and IDs on the system, implementing a **fuzzing attack**. Finally, an attacker may have the only goal to silence a node without replacing it, creating a **drop** attack. As further discussed in Section 4 when we present our attack tool, we generate datasets that consider these attacks and evaluate the systems against all of them.

Intrusion Detection Systems (IDSs) for vehicular systems analyze the stream of packets and monitor the events on on-board networks for signs of intrusions. Among these, machine learning-based, particularly RNN-based solutions, have proven effective in recognizing anomalous behavior [17,32]. Based on the approach and the results of CANnolo [17], in this paper, we propose CANdito, an RNN-based, unsupervised IDS that exploits Long Short-Term Memory (LSTM) autoencoders to detect anomalies through a signal reconstruction process. After a preprocessing stage, it learns the legitimate signal behavior through an LSTM-based autoencoder. Then, it computes the anomaly score for each CAN ID based on their reconstruction error. In particular, we improve the overall architecture and lighten CANnolo computational requirements to meet real-world timing constraints of the automotive domain. We prove the effectiveness of CANdito by conducting experiments on a real dataset of CAN traffic augmented with a set of synthetic but realistic attacks. With respect to existing works, we consider a broader spectrum of attacks and implement a tool to inject them into real-world CAN traffic logs. This tool is available at `url.to.be.released.once.published`. We demonstrate that CANdito outperforms its predecessor CANnolo, with improved detection rates and a reduction of more than 50% of the timing overheads. Moreover, to provide a fair evaluation of CANdito, we compare its performances against state-of-art Intrusion Detection Systems (IDSs) for in-vehicle network on a public dataset with attack messages. Our experimental results show that

CANdito outperforms state-of-the-art detection methods with a perfect True Positive Ration (TPR) and lower time requirements.

In summary, our contributions are the following:

– We improve CANnolo with CANdito, an RNN-based, unsupervised IDS that exploits LSTM autoencoders to detect anomalies through a signal reconstruction process.
– We design and provide CANtack, a tool to generate and inject synthetic attacks in real datasets, to be used as a benchmarking suite for IDS in the automotive domain.
– An evaluation of CANdito from the point of view of detection and timing performances on a more comprehensive dataset with respect to state-of-the-art works.

2 Related Works

Intrusion detection for automotive onboard networks has drawn vast research in recent years. We refer to Jo et al. [8] for a comprehensive survey of intra-vehicle IDSs, which can be divided into packet-based and hardware-based. Packet-based IDSs can be further divided into flow-based, payload-based, and combined. Flow-based IDSs (e.g., [12,26,27,33]) monitor the CAN bus, extract distinct features as message frequency or packet inter-arrival time, and use them to detect anomalous events without inspecting the payloads of the messages. On the contrary, payload-based IDSs (e.g., [1,7,9,17]) examine the payload of CAN packets (usually only data frames). Finally, combined IDS (e.g., [21,38]) are a combination of the previous two techniques. Many different machine learning techniques have been applied to payload-based CAN intrusion detection, from GANs to CNNs, in both a supervised and unsupervised fashion:

Kang and Kang [9] propose a supervised payload-based IDS based on Deep Neural Network (DNN). The input feature does not use the entire payload, but only the *mode information*, which represents the command state of an ECU, and the *value information*, which represents the value of the mode (e.g., wheel angle or speed). Multiple techniques exploit CNNs. For example, Rec-CNN [5] transforms the detection process into an image recognition one in an attempt to exploit the image recognition capabilities of CNNs, generating so-called recurrence plots that graphically represent the time series of packets. Reduced Inception-ResNet [28] exploits the deep convolutional neural network model of the Inception-ResNet architecture, a supervised model designed for image recognition, but significantly simplifies it in an attempt at lower computational times. CANTransfer [31] instead applies a supervised convolutional LSTM-based model to CAN intrusion detection with the goal of applying transfer learning to simplify the process of training different vehicles and IDs. CNNs are, however, better at processing spatial data, while RNNs and LSTMs are generally better suited for temporal data. CAN-ADF [30] exploits RNNs in a supervised fashion and inserts a rule-based IDS in front of it to detect simpler attacks in an attempt to not only detect attacks but also classify them, while TSP [24] studies the differences between various loss functions in the development of an LSTM-based IDS. HyDL-IDS [14] exploits CNNs and LSTMs to build a supervised system that extracts both temporal and spatial features of each packet stream. Often paired with these techniques, autoencoders have been proposed to predict or reconstruct time series. O-DAE [13] approaches detection by attempting to remove noise from the time series of packets via a supervised DAE autoencoder and then recognizing the attacks through the reconstruction. CANet [7] is one of the few IDSs that elaborates multiple IDs simultaneously, theoretically making it possible to exploit the correlation between the different information shared through the various IDs. It accomplishes this through an LSTM network per CAN ID and an autoencoder that receives the concatenated output of the various LSTMs. CANnolo [17] is an unsupervised IDS based on a LSTM autoencoder and represents the starting point of CANdito. A window of packets is fed to the RNN autoencoder, which attempts to reconstruct it following the trained model. Finally, the capabilities of GANs have also been evaluated, E-GAN [35] uses an unsupervised GAN and the DBC files for a dataset provided by the manufacturer first to comprehend the layout of a packet and then recognizes anomalies inside its various elements.

Motivation. The main limitation of current state-of-the-art approaches is that, while different methods work well on different problems, they can hardly achieve results that are good enough on any anomaly and, at the same time, provide fast enough results to process the network's traffic in real-time. The way this limitation compels a system largely depends on the different types of IDS approach adopted. Generally, flow-based approaches can provide fast predictions while being limited to the detection of specific kinds of vulnerabilities, while payload-based approaches have a broader scope, but it is often a problem to make them work in real-time on the total traffic. Moreover, the traffic on different CAN IDs has different characteristics, but the current state-of-the-art methods rarely consider them to provide better results.

3 CANdito

In this section, we describe CANdito, an improved version of CANnolo [17], a state-of-the-art IDS for CAN that exploits Recurrent Neural Network (RNN)-based autoencoders. RNN-based architectures are effective in modeling time-series and have been successfully proposed for CAN traffic anomaly detection [32]. On the other hand, autoencoders do not require a labeled training dataset since target signals are generated automatically from the input sequence. Moreover, being unsupervised, they learn a model of the legitimate network traffic and not specific anomalies, making them able to potentially recognize novel attacks. Figure 1 shows an overview of the architecture of the system, which works by reconstructing the time series of CAN packets for each ID and computes their anomaly score based on the reconstruction error. The effectiveness of reconstructing the signal (as opposed to predicting the successive one) has been demonstrated by Malhotra et al. [19], which uses LSTM encoder-decoder architectures as recon-

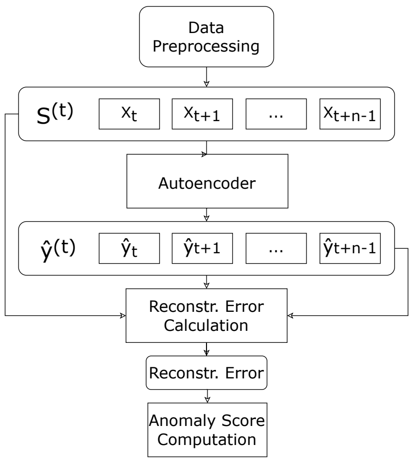

Figure 1: Overview of CANdito's detection process.

structors to detect anomalies in multi-sensor contexts. It comprises three modules: a data preprocessing module, an LSTM-based autoencoder, which learns the legitimate signal behavior, and an anomaly detector, which compares the reconstruction errors.

3.1 Data Preprocessing

The first module of CANdito builds the input sequences by applying the READ algorithm [20] and associating to each CAN ID the corresponding ranges of the signals. Using this information, the payloads of each packet are converted into the list of their signals rescaled in the [0-1] range, excluding constant bits, counters, and Cyclic Redundancy Checks (CRCs).

Our resulting input sequence is composed by a matrix n x k, where $n = 40$ is the dimension of the window of CAN packets, k is the number of signals per packet (rescaled in the [0-1] range). Notice that at this step, CANdito discards and flags as anomalous any CAN ID that has not been found in the training set.

3.2 LSTM-based autoencoder

The second module of CANdito is based on an autoencoder whose encoder and decoder layers are implemented with two recurrent LSTMs. The encoder is composed of a dense layer, which consists of 128 units with an Exponential Linear Unit (ELU) [4] activation function. The dense layer is followed by a 20% dropout layer and two recurrent LSTM layers with $L = 64$ units each. The cell and hidden states of the last LSTM layer of the encoder are used to initialize the states of the first LSTM layer of the decoder. The output of the encoder is reversed before being fed into the decoder. Symmetrically, the decoder consists of two recurrent LSTM layers with 64 units each, a dense layer, consisting of 128 units with ELU activation function, and a second output dense layer with k units with sigmoid activation function (to scale the results to the [0,1] interval).

Training and Tuning. The input data of the training and threshold calculation processes are composed by sliding one time step at a time a window of n packets, while for the testing process, in order to have a lightweight detection process, the windows of packets do not overlap. A dataset consisting only of legitimate data sequences has been used as the baseline to establish the 'normal' behavior for any given CAN ID. In particular, we perform training and validation by reconstructing legitimate traffic data and minimizing the reconstruction error between a given source sequence $s^{(t)}$ and a target sequence $y^{(t)}$. Also, we make use of an untampered dataset to perform early stopping and a dataset injected with our attack tool (presented in Section 4) to evaluate the performance of the model and, consequently, tune hyperparameters. The loss function of choice is Mean Squared Error (MSE). The optimizer of choice is Adam [10] with a learning rate of 0.001. The model of each CAN ID has been trained for a maximum of 50 epochs with an early stopping with patience 5 (i.e., training is stopped before the maximum number of epochs if the validation accuracy does not improve for 5 consecutive epochs).

3.3 Anomaly score computation

The third module of CANdito works in an unsupervised fashion by computing the distance between the reconstruction error and the expected normal distribution computed during training. The anomaly score indicates the likelihood of the test sequence to be anomalous.

Each window of n packets is fed into the trained autoencoder and an anomaly score is assigned to each window as the squared l2-norm of the reconstruction error: $e^{(t)} = \|\hat{y}^{(t)} - s^{(t)}\|_2^2$. The chosen detection threshold is, as proposed by Hanselmann et al. [7], the 99.99 percentile of the scores. The windows whose score is greater than the threshold are marked as anomalies.

3.4 Improvements from CANnolo

As previously stated, CANdito is based on CANnolo [17], in light of its promising results in the detection of attacks on CAN. In particular, CANdito required an in-depth study of CANnolo. To do so, we implemented CANnolo from scratch and tested it on our extended dataset. This evaluation brought the improvements described below.

Generalization and computation improvements. The number of artificial neurons of all layers has been halved. This reduction in dimensions has a twofold scope. The networks of several CAN IDs are prone to overfitting, reducing the dimension of the layers mitigated this problem. Moreover, reducing the dimension of the layers has a substantial impact on the improvement of the computation times of the network.

Overfitting and vanishing gradient mitigation. The activation functions of the two symmetric dense layers of the encoder and decoder have been modified from hyperbolic tangent to ELU. This was proven to mitigate the vanishing gradient problem and to improve the generalization capabilities of the network [4, 7].

Input bloat reduction. The inputs of CANnolo are bit-strings composed of the condensed notation of the packets (i.e., excluding constant bits). In our solution, we also exclude signals marked by READ as counters or CRCs. In fact, counters and CRCs do not carry relevant information for the reconstruction module, as demonstrated by the fact that considering them did not improve the effectiveness of the system. Moreover, CANdito's input is not composed of bit-strings but by each signal detected by READ rescaled in the [0-1] range. This significantly reduces the input dimensions, further lowering computing times, while comparative tests with the two input methods did not show meaningful effects on detection performances.

Underfitting mitigation. The output sequence of the encoder is reversed before being fed into the decoder. This operation is meant to help the network reconstruct the target sequence, which is also reversed as suggested by Sutskever et al. [29]. While for some CAN IDs the network results are similar with or without reversing the encoder output, other CAN IDs networks are affected by severe underfitting if the encoder output is not reversed. The same CAN IDs networks perform well after the change.

Lower computational requirements. To lower the computational effort, we do not feed the reconstructed sequence back into the decoder. Evaluations at design time did not show improvements in detection performances between the two implementations. From an implementation standpoint, code optimization and moving from Keras to PyTorch also greatly lowered computation requirements.

Anomaly score computation improvement. CANnolo uses the Mahalanobis distance between the reconstruction error of the window under evaluation and the distribution of errors in untampered scenarios. While such distance has been considered for the anomaly scores computation, it has underperformed with our model. Instead, we opted to compute the anomaly score as the squared l2-norm of the reconstruction error, with a 99.99 percentile of the scores as a detection threshold, as suggested in [7].

4 CANtack

We designed CANtack to have an easy, partially automated way to consistently generate different types of anomalies in our datasets while starting from datasets of real CAN traffic. The tool is available at [2] alongside instructions on how to use it. The output of the tool is a dataset structured similarly to the ReCAN

[2]CANtack url: https://github.com/ascarecrowhat/CANtack

dataset [37], but with an additional *isTampered* column, which indicates whether a log entry has been tampered with or not.

The tool allows choosing between the different attack implementations, which enable to deploy all the attacks presented in Section 1. Moreover, all different types of attacks (except for cases in which this does not make sense, i.e., *drop* and *DoS*) can be either performed in an *injection* fashion (i.e., without modifying the original packets of the traffic and specifying an injection rate) or in a *masquerade* fashion (i.e., substituting the original packets with the tampered ones). We proceed to present the list of attacks and their user-defined parameters. Note that for all attacks, the user needs to set the ID to tamper and the beginning time for the attack in seconds.

Basic injection. The user can specify a payload and a number of tampered packets. The tool injects or replaces a number of packets with the new tampered payload.

Progressive injection. As above, the value of every single payload can be specified.

DoS. The network is flooded with packets with ID "0" for the specified duration. It is possible to define the percentage of the bus to fill.

Drop. A set amount of messages from the given ID are deleted from the dataset.

Fuzzy. The payloads are injected or tampered with random values. It is also possible to choose a bit range to fuzzy a value, e.g., simulating the fuzzing only of sensor data.

Replay. The payloads are values sniffed from the dataset. To implement this attack, it is necessary to define the initial sniffing time and whether to partially randomize the position of the first copied packet among the sniffed ones. Moreover, a series of replacements can be specified to modify only some bit-ranges with other values. These values can be set with the following options:

- **payloads** replaces the bit-ranges with explicitly defined data;
- **fuzzy** randomizes the bit-range;
- **min** and **max** respectively find the minimum and maximum value (considered as an integer) registered in the dataset for that bit-range;
- **seamless change** defines a final value to reach and increases or decreases the bit-range from the value read in the last untampered line to the chosen one;
- **counter** increases the values of the bit-range by one per packet, in a counter-like fashion.

5 Experimental validation

First, we compare CANdito with CANnolo [17] from both the point of view of the detection and the temporal performances. In particular, this experiment evaluates the performance of CANnolo and CANdito over different datasets tampered with our novel attack tool 4. CANnolo's implementation has been kept almost untouched, threshold computation criterion included. This consideration must be explicitly made since the authors used as a comparison metric for their experiment just the Area Under Curve (AUC) without focusing over the threshold computation criterion, which was demonstrated to be sub-optimal.

Then, we demonstrate the effectiveness of CANdito by comparing its performances against state-of-the-art solutions on a public dataset with real-world attacks.

We evaluate the detection performances of the systems under analysis by considering the most common metrics used to evaluate unbalanced datasets. Specifically, we make use of Detection Rate (DR), False Positive Ratio (FPR), F1 score, Matthews Correlation Coefficient (MCC). Moreover, to evaluate the timing performances, we use the Testing Time per Packet (TTP).

We evaluate our work through two public datasets: the ReCAN C-1 dataset [37] used for the first set of experiments and the car-hacking dataset [28] used for the comparison with the state of the art. The experiments have been tested by serving datasets split in windows of pre-defined size, as it should happen at runtime instead of testing the entire sequence in one batch; doing so permits to have a measure of the testing time that is more accurate.

5.1 ReCAN Datasets

The ReCAN dataset [37] is a public dataset of CAN logs retrieved in real-world scenarios. We select the C-1 sub-dataset, which has been retrieved from multiple test drives of an Alfa Giulia. In more detail, we use sub-datasets 1, 2, 6, 8, and 9 to train the model, sub-dataset 4 to calculate thresholds, sub-dataset 5 for validation and hyper-parameter tuning, and finally, sub-dataset 7 for testing. We then use our attack tool to generate the attack datasets. All the datasets are generated starting from sub-dataset 7 of the ReCAN C-1 dataset. The datasets have the goal of building attacks as presented in Section 1. All the datasets, their details, and implementation parameters are available on the CANtack webpage[2].

Injection Dataset. This dataset contains generic injection attacks, which consists of added packets on the network, leaving all packets already present in the dataset unchanged. The new packets are sniffed from previous traffic and only one physical signal is modified (recognized through the READ algorithm [20]), although the value changes inside the range of already existing values of the signal. This is done in an attempt to comply as much as possible with the behavior of the ID and increase the difficulty of detection. The packets are added at 20 times the frequency of the original packet and continue for a sequence of 50 packets.

Drop Dataset. This dataset simulates the event where an attacker turns off an ECU or its CAN controller. The attack consists of removing a sequence of valid packets from the original traffic. Twenty-five consecutive packets are removed each time.

Masquerade Dataset. This dataset contains generic masquerade attacks that would not be detectable only through frequency-based or rule-based features.

The modified packets are sniffed from previous traffic and one or more physical signals are modified in the same fashion as the injection dataset. Moreover, the anomalous packets maintain the same timestamps as the packet they replace, alongside its ID. Each anomalous sequence has a length of 25 packets.

Fuzzed Dataset. This dataset represents the event where an attacker is testing random values of signals in order (usually) to trigger unexpected behavior. The attack is made in a masquerade fashion (the original packets are removed and replaced), but only the bits included in some of the signals are fuzzed, while, for example, constant bits are left untampered. As above, 25 consecutive packets are removed each time. As above, each anomalous sequence has a length of 25 packets.

Seamless Change Dataset. This dataset contains masquerade attacks that attempt to evade detection by changing the payload of the packets progressively until the desired value is reached. The physical values in the tampered ID have to be at least 4 bits long. The new packets are sniffed from previous traffic and only one physical signal is modified. As above, each anomalous sequence has a length of 25 packets.

Full Replay Dataset. This dataset contains masquerade attacks that attempt to evade detection by copying exact sequences on the bus. No additional check is made while generating the attacks. Consequently, there is no warranty that the new sequence is taken from a moment where the condition of the car is very different, lowering the detection capabilities but also the actual effects of the attacks. This dataset is primarily interesting to compare the ability of IDSto detect anomalous sequences that are perfectly valid in a different context. As above, each anomalous sequence has a length of 25 packets.

5.2 Car-hacking dataset

The car-hacking dataset [28] is composed of logs of real-time CAN messages via the onboard diagnostic (OBD-II) port of two running vehicles (KIA Soul and Hyundai Sonata) with message attacks. It has four data features, including timestamp, identifier (ID, in hexadecimal format), data length code (DLC, valued from 0 to 8) and data payload (8 bytes), and the label of a CAN message. We refer the reader to [28] for further details on the public dataset under consideration. It contains normal CAN messages (14,237,978) and anomaly messages (2,331,497) belonging to three categories of attacks (for a total of four attacks).

DoS attack. It aims to flood the CAN bus with numerous forged messages with low ID values in a short time interval. Thus, almost all the communication resources are occupied so that messages from other nodes will be delayed or blocked.

Fuzzy attack. Fake messages are sent from malicious ECUs into the CAN bus at a slower rate than the DoS attack.

Impersonation attacks. They realize unauthorized service access by spoofing legitimate authentication credentials, such as **spoofing the drive gear and the RPM gauze**.

5.3 Experimental Results

Table 1: CANnolo vs CANdito performances, tested over the masquerade, fuzzy, seamless change, and full replay datasets. Only CAN IDs recommended for testing by CANnolo's authors have been taken into consideration.

Dataset		DR	FPR	F1	MCC	TTP
Masq.	CANdito	**.9258**	.0081	**.9505**	**.9336**	
	CANnolo	.6477	**.0029**	.7823	.7502	
Fuzzy	CANdito	**.9989**	.0081	**.9886**	**.9844**	**.0700**
	CANnolo	.9541	**.0029**	.9724	.9629	
Seam.	CANdito	**.8972**	.0079	**.9345**	**.9143**	
	CANnolo	.7481	**.0029**	.8518	.8224	1.0630
Replay	CANdito	**.5820**	.0080	.7254	.6909	
	CANnolo	.5801	**.0029**	**.7304**	**.7024**	

Results on the ReCAN dataset As shown in Table 1, our solution is not only twice as fast as CANnolo in providing the predictions, but it is also more effective in almost all the considered attack scenarios.

Focusing on payload-based anomalies, CANdito generally outperforms CANnolo on both the entire dataset and on the subset of the dataset composed of the selected CAN IDs. For the Masquerade dataset, CANdito performs evidently better, with an F1 score and MCC both over .15 point higher than CANnolo. We explain the better performances obtained with the different approaches used to compute the detection threshold. For the Fuzzy datasets, CANdito shows similar performances to CANnolo, with a higher DR, F1-score, and MCC. For the Seamless Change dataset, the better performances of our new architecture are more evident. In fact, both the F1 score and MCC improvements range between .07 and .09. For the Full replayed dataset, the original model is slightly more effective, but the performances of the two systems are comparable. However, in light of the stringent requirements of the automotive domain, where the lack of computational power is critical [18], CANdito is preferable to CANnolo since it provides detection results in less than half of the time.

As expected, both systems perform poorly on flow-based anomalies (i.e., Injection and Drop datasets) since they implement payload-based detection and do not detect changes in the frequency of packets arrivals.

It is interesting to note that CAN ID $0x1E340000$ is responsible for the 18% of overall false positives and features a very different behavior between the train set and the test set with the presence of flipping bits that were static for the entire duration of the training set. This said, it is encouraging to know that a large part

Table 2: Detection Performance Comparison of State-of-the-art IDS against CANdito. In **bold**, the best performance by metric and attack category.

IDSs	Attacks	Accuracy	Precision	TPR	FPR	F1-score	TTP
Reduced Inception-ResNet [28]	DoS Attack	**0.9993**	**0.9995**	0.9963	**0.0001**	**0.9980**	1.5633
	Fuzzy Attack	0.8730	0	0	0.0002	-	
	Gear Spoofing Attack	0.8223	0	0	**0.0001**	-	
	RPM Spoofing Attack	0.7774	0	0	0.0003	-	
CANTransfer [31]	DoS Attack	0.9991	0.9990	0.9951	0.0002	0.9971	1.3264
	Fuzzy Attack	0.8718	0	0	**0.0001**	-	
	Fuzzy Attack (1-shot)	0.8664	0.9794	0.0309	**0.0001**	0.0599	
	Gear Spoofing Attack	0.8223	0	0	0.0004	-	
	RPM Spoofing Attack	0.7774	0	0	0.0003	-	
CAN-ADF [30]	DoS Attack	0.9938	0.9826	0.9785	0.0033	0.9805	1.4476
	Fuzzy Attack	0.8715	0.0505	0.0002	0.0006	0.0004	
	Gear Spoofing Attack	0.8222	0	0	0.0004	-	
	RPM Spoofing Attack	0.7769	0.1200	0.0005	0.0012	0.0011	
TSP [24]	DoS Attack	0.9802	0.9100	0.9728	0.0183	0.9403	1.1422
	Fuzzy Attack	0.8714	0	0	0.0005	-	
	Gear Spoofing Attack	0.8221	0	0	0.0005	-	
	RPM Spoofing Attack	0.7774	0	0	0.0003	-	
O-DAE [13]	DoS Attack	0.9933	0.9742	0.9843	0.0050	0.9792	1.2130
	Fuzzy Attack	0.8714	0	0	0.0006	-	
	Gear Spoofing Attack	0.8222	0	0	0.0004	-	
	RPM Spoofing Attack	0.7774	0	0	0.0003	-	
LDAN [39]	DoS Attack	0.9806	0.9099	0.9756	0.0184	0.9416	0.9283
	Fuzzy Attack	0.8717	0	0	0.0006	-	
	Gear Spoofing Attack	0.8224	0	0	**0.0001**	-	
	RPM Spoofing Attack	0.7775	0	0	**0.0002**	-	
E-GAN [35]	DoS Attack	0.9806	0.9099	0.9756	0.0184	0.9416	1.0331
	Fuzzy Attack	0.8717	0	0	0.0002	-	
	Gear Spoofing Attack	0.8224	0	0	**0.0001**	-	
	RPM Spoofing Attack	0.7774	0	0	0.0003	-	
HyDL-IDS [14]	DoS Attack	0.9936	0.9819	0.9781	0.0034	0.9800	0.4395
	Fuzzy Attack	0.8715	0.0612	0.0002	0.0005	0.0005	
	Gear Spoofing Attack	0.8221	0	0	**0.0001**	-	
	RPM Spoofing Attack	0.7769	0.1042	0.0005	0.0011	0.0009	
CANet [7]	DoS Attack	0.9993	0.9992	0.9966	0.0014	0.9979	0.3357
	Fuzzy Attack	0.8717	0	0	0.0002	-	
	Gear Spoofing Attack	0.8223	0	0	0.0001	-	
	RPM Spoofing Attack	0.7774	0	0	0.0003	-	
Rec-CNN [5]	DoS Attack	0.9803	0.9097	0.9740	0.0185	0.9408	0.3278
	Fuzzy Attack	0.8714	0	0	0.0006	-	
	Gear Spoofing Attack	0.8221	0	0	0.0005	-	
	RPM Spoofing Attack	0.7774	0	0	0.0003	-	
CANdito	DoS Attack	0.9983	0.9926	**1**	0.0021	0.9963	**0.07**[3]
	Fuzzy Attack	**0.9608**	**0.9915**	**0.8884**	0.0094	**0.9296**	
	Gear Spoofing Attack	**0.9983**	**0.9984**	**0.9934**	0.0004	**0.9959**	
	RPM Spoofing Attack	**0.9996**	**0.9986**	**1**	0.0004	**0.9993**	

of the FPR depends on a small set of CAN IDs because this demonstrates that future works may improve the results of finding a different classification of these "pathological" CAN IDs. Another possible alternative that is not particularly time-consuming, considering the small dimension of this set of CAN IDs is to perform a human-supervised fine-tuning of the model for these specific CAN IDs on top of the automatic classification.

Results on the Car-hacking Dataset In order to provide a comparison with the other various machine learning techniques used in the state of the art, we make use of the systematization done by Wang et al. [34] on the public car-hacking dataset, and follow the same experimental procedure on CANdito. Table 2 contains Wang et al. results followed by ours. CANdito achieves better detection rate on all the datasets, and comparable metrics where it does not win. The significantly lower detection rate on the fuzzy dataset in relation to the others can be attributed by the behavior of the dataset, where the randomized ID sometimes end up being one of the valid ones, but only one malicious packet is inserted in a window of 39 valid ones. The IDS that better stands up against CANdito in terms of detection performances is Reduced Inception-ResNet [28], which, however, is 20 times slower than CANdito, and even more importantly, given the average packet inter-arrival time of the dataset, which is 0.77ms, is not compliant with the real-time requirements of the automotive domain. Finally, CANdito shows overall good detection performances on all the categories of attacks.

6 Conclusions

In this paper, we presented CANdito, an improved RNN-based and unsupervised IDS that exploits LSTM autoencoders to detect anomalies through a signal reconstruction process in CAN traffic. We evaluated CANdito from the point of view of the detection and timing performances on a more comprehensive real-world dataset augmented with synthetic attacks generated with CANtack, a tool to generate and inject synthetic attacks in real datasets, which can be used as a benchmarking suite for IDS in the automotive domain. Moreover, we compared its performances against state-of-art Intrusion Detection Systems (IDSs) for in-vehicle network on a public dataset with attack messages, showing that CANdito performs overall better of the current state of the art while requiring significantly less time - up to 1/20 - than the other detection techniques. We plan to overcome CANdito limitation in detecting attacks that work in the frequency domain by complementing the improved detection power of the payload-based detection system presented in this work with the power of frequency-based approaches to building an end-to-end hybrid IDS able to fully exploit all CAN IDs.

References

1. Amato, F., Coppolino, L., Mercaldo, F., Moscato, F., Nardone, R., Santone, A.: Can-bus attack detection with deep learning. IEEE Trans. Intell. Transp. Syst.

22(8), 5081–5090 (2021). https://doi.org/10.1109/TITS.2020.3046974, https://doi.org/10.1109/TITS.2020.3046974

2. Buttigieg, R., Farrugia, M., Meli, C.: Security issues in controller area networks in automobiles. CoRR **abs/1711.05824** (2017), http://arxiv.org/abs/1711.05824
3. Checkoway, S., Mccoy, D., Anderson, D., Kantor, B., Shacham, H., Savage, S., Koscher, K., Czeskis, A., Roesner, F., Kohno, T.: Comprehensive experimental analyses of automotive attack surfaces (08 2011)
4. Clevert, D.A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289 (2015)
5. Desta, A.K., Ohira, S., Arai, I., Fujikawa, K.: Rec-cnn: In-vehicle networks intrusion detection using convolutional neural networks trained on recurrence plots. Veh. Commun. **35**, 100470 (2022). https://doi.org/10.1016/j.vehcom.2022.100470, https://doi.org/10.1016/j.vehcom.2022.100470
6. de Faveri Tron, A., Longari, S., Carminati, M., Polino, M., Zanero, S.: Canflict: Exploiting peripheral conflicts for data-link layer attacks on automotive networks. In: Yin, H., Stavrou, A., Cremers, C., Shi, E. (eds.) Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022. pp. 711–723. ACM (2022). https://doi.org/10.1145/3548606.3560618, https://doi.org/10.1145/3548606.3560618
7. Hanselmann, M., Strauss, T., Dormann, K., Ulmer, H.: Canet: An unsupervised intrusion detection system for high dimensional can bus data. IEEE Access **8**, 58194–58205 (2020)
8. Jo, H.J., Choi, W.: A survey of attacks on controller area networks and corresponding countermeasures. IEEE Trans. Intell. Transp. Syst. **23**(7), 6123–6141 (2022). https://doi.org/10.1109/TITS.2021.3078740, https://doi.org/10.1109/TITS.2021.3078740
9. Kang, M.J., Kang, J.W.: Intrusion detection system using deep neural network for in-vehicle network security. PloS one **11**(6), e0155781 (2016)
10. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
11. Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., et al.: Experimental security analysis of a modern automobile. In: 2010 IEEE Symposium on Security and Privacy. pp. 447–462. IEEE (2010)
12. Lampe, B., Meng, W.: IDS for CAN: A practical intrusion detection system for CAN bus security. In: IEEE Global Communications Conference, GLOBECOM 2022, Rio de Janeiro, Brazil, December 4-8, 2022. pp. 1782–1787. IEEE (2022). https://doi.org/10.1109/GLOBECOM48099.2022.10001536, https://doi.org/10.1109/GLOBECOM48099.2022.10001536
13. Lin, Y., Chen, C., Xiao, F., Avatefipour, O., Alsubhi, K., Yunianta, A.: An evolutionary deep learning anomaly detection framework for in-vehicle networks-can bus. IEEE Transactions on Industry Applications (2020)
14. Lo, W., AlQahtani, H., Thakur, K., Almadhor, A., Chander, S., Kumar, G.: A hybrid deep learning based intrusion detection system using spatial-temporal representation of in-vehicle network traffic. Veh. Commun. **35**, 100471 (2022). https://doi.org/10.1016/j.vehcom.2022.100471, https://doi.org/10.1016/j.vehcom.2022.100471

15. Longari, S., Cannizzo, A., Carminati, M., Zanero, S.: A secure-by-design framework for automotive on-board network risk analysis. In: 2019 IEEE Vehicular Networking Conference (VNC). pp. 1–8 (2019). https://doi.org/10.1109/VNC48660.2019.9062783
16. Longari, S., Penco, M., Carminati, M., Zanero, S.: Copycan: An error-handling protocol based intrusion detection system for controller area network. In: Cavallaro, L., Kinder, J., Holz, T. (eds.) Proceedings of the ACM Workshop on Cyber-Physical Systems Security & Privacy, CPS-SPC@CCS 2019, London, UK, November 11, 2019. pp. 39–50. ACM (2019). https://doi.org/10.1145/3338499.3357362, `https://doi.org/10.1145/3338499.3357362`
17. Longari, S., Valcarcel, D.H.N., Zago, M., Carminati, M., Zanero, S.: Cannolo: An anomaly detection system based on lstm autoencoders for controller area network. IEEE Transactions on Network and Service Management (2020)
18. Maffiola, D., Longari, S., Carminati, M., Tanelli, M., Zanero, S.: Goliath: A decentralized framework for data collection in intelligent transportation systems. IEEE Transactions on Intelligent Transportation Systems (2021)
19. Malhotra, P., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., Shroff, G.: Lstm-based encoder-decoder for multi-sensor anomaly detection. arXiv preprint arXiv:1607.00148 (2016)
20. Marchetti, M., Stabili, D.: Read: Reverse engineering of automotive data frames. IEEE Transactions on Information Forensics and Security **14**(4), 1083–1097 (2018)
21. Marchetti, M., Stabili, D., Guido, A., Colajanni, M.: Evaluation of anomaly detection for in-vehicle networks through information-theoretic algorithms. In: 2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI). pp. 1–6. IEEE (2016)
22. Miller, C., Valasek, C.: Adventures in automotive networks and control units. Def Con **21**(260-264), 15–31 (2013)
23. Miller, C., Valasek, C.: Remote exploitation of an unaltered passenger vehicle. Black Hat USA **2015**(S 91) (2015)
24. Qin, H., Yan, M., Ji, H.: Application of controller area network (CAN) bus anomaly detection based on time series prediction. Veh. Commun. **27**, 100291 (2021). https://doi.org/10.1016/j.vehcom.2020.100291, `https://doi.org/10.1016/j.vehcom.2020.100291`
25. Robert Bosch GMBH: Can specification, version 2.0. Standard, Robert Bosch GmbH, Stuttgart, Germany (1991)
26. Seo, E., Song, H.M., Kim, H.K.: Gids: Gan based intrusion detection system for in-vehicle network. In: 2018 16th Annual Conference on Privacy, Security and Trust (PST). pp. 1–6. IEEE (2018)
27. Song, H., Kim, H., Kim, H.K.: Intrusion detection system based on the analysis of time intervals of can messages for in-vehicle network. pp. 63–68 (01 2016). https://doi.org/10.1109/ICOIN.2016.7427089
28. Song, H.M., Woo, J., Kim, H.K.: In-vehicle network intrusion detection using deep convolutional neural network. Veh. Commun. **21** (2020). https://doi.org/10.1016/j.vehcom.2019.100198, `https://doi.org/10.1016/j.vehcom.2019.100198`
29. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. arXiv preprint arXiv:1409.3215 (2014)
30. Tariq, S., Lee, S., Kim, H.K., Woo, S.S.: CAN-ADF: the controller area network attack detection framework. Comput. Secur. **94**, 101857 (2020). https://doi.org/10.1016/j.cose.2020.101857, `https://doi.org/10.1016/j.cose.2020.101857`

31. Tariq, S., Lee, S., Woo, S.S.: Cantransfer: transfer learning based intrusion detection on a controller area network using convolutional LSTM network. In: Hung, C., Cerný, T., Shin, D., Bechini, A. (eds.) SAC '20: The 35th ACM/SIGAPP Symposium on Applied Computing, online event, [Brno, Czech Republic], March 30 - April 3, 2020. pp. 1048–1055. ACM (2020). https://doi.org/10.1145/3341105.3373868, https://doi.org/10.1145/3341105.3373868
32. Taylor, A.: Anomaly-based detection of malicious activity in in-vehicle networks. Ph.D. thesis, Université d'Ottawa/University of Ottawa (2017)
33. Taylor, A., Japkowicz, N., Leblanc, S.: Frequency-based anomaly detection for the automotive can bus. In: 2015 World Congress on Industrial Control Systems Security (WCICSS). pp. 45–49. IEEE (2015)
34. Wang, K., Zhang, A., Sun, H., Wang, B.: Analysis of recent deep-learning-based intrusion detection methods for in-vehicle network. IEEE Transactions on Intelligent Transportation Systems pp. 1–12 (2022). https://doi.org/10.1109/TITS.2022.3222486
35. Xie, G., Yang, L.T., Yang, Y., Luo, H., Li, R., Alazab, M.: Threat analysis for automotive CAN networks: A GAN model-based intrusion detection technique. IEEE Trans. Intell. Transp. Syst. **22**(7), 4467–4477 (2021). https://doi.org/10.1109/TITS.2021.3055351, https://doi.org/10.1109/TITS.2021.3055351
36. Young, C., Zambreno, J., Olufowobi, H., Bloom, G.: Survey of automotive controller area network intrusion detection systems. IEEE Des. Test **36**(6), 48–55 (2019). https://doi.org/10.1109/MDAT.2019.2899062, https://doi.org/10.1109/MDAT.2019.2899062
37. Zago, M., Longari, S., Tricarico, A., Carminati, M., Pérez, M.G., Pérez, G.M., Zanero, S.: Recan–dataset for reverse engineering of controller area networks. Data in brief **29**, 105149 (2020)
38. Zhang, L., Shi, L., Kaja, N., Ma, D.: A two-stage deep learning approach for can intrusion detection (2018)
39. Zhao, R., Yin, J., Xue, Z., Gui, G., Adebisi, B., Ohtsuki, T., Gacanin, H., Sari, H.: An efficient intrusion detection method based on dynamic autoencoder. IEEE Wirel. Commun. Lett. **10**(8), 1707–1711 (2021). https://doi.org/10.1109/LWC.2021.3077946, https://doi.org/10.1109/LWC.2021.3077946