An Anomaly Detection Method based on Multiple LSTM-Autoencoder Models for In-vehicle Network

Ilsun You *, TaeGuen Kim, Jiyoon Kim

Posted Date: 4 July 2023

doi: 10.20944/preprints202307.0062.v1

Keywords: anomaly detection; vehicular network; Controller Area Network; LSTM-Autoencoder model; intrusion detection system; Vehicular IoT

Article

An Anomaly Detection Method Based on Multiple LSTM-Autoencoder Models for In-Vehicle Network

TaeGuen Kim [1], Jiyoon Kim [2] and Ilsun You [3,*]

[1] Department of Information Security Engineering, Soonchunhyang University, Republic of Korea
[2] School of Computer Science, Gyeongsang National University, Republic of Korea
[3] Department of Financial Information Security, Kookmin University
* Correspondence: ilsunu@gmail.com

Abstract: The CAN protocol is widely adopted for in-vehicle networks due to its cost efficiency and reliable transmission. However, despite its popularity, the protocol lacks built-in security mechanisms, making it vulnerable to various attacks such as flooding, fuzzing, and DoS. These attacks can exploit vulnerabilities and disrupt the normal behavior of the in-vehicle network. One of the main reasons for these security concerns is that the protocol relies on broadcast frames for communication between ECUs within the network. To tackle this issue, this paper presents an intrusion detection system that leverages multiple LSTM-Autoencoders. The proposed system utilizes diverse features, including transmission interval and payload value changes, to capture various characteristics of normal network behavior. By analyzing different types of features separately using the LSTM-Autoencoder model, the system effectively detects anomalies. In our evaluation, we conducted experiments using real vehicle network traffic, and the results demonstrated the system's high precision with a 99% detection rate in identifying anomalies.

Keywords: anomaly detection; vehicular network; Controller Area Network; LSTM-Autoencoder model; intrusion detection system; Vehicular IoT

1. Introduction

As automotive information technology rapidly advances in modern society, vehicles have evolved beyond mere mechanical devices and now encompass numerous sensors and Electronic Control Units (ECUs). The emergence of connected cars and autonomous vehicles, consisting of many complex systems, has led to the rapid development of in-vehicle networks [1,2]. There are many communication protocols such as Controller Area Network (CAN), FlexRay, Local Interconnect Network (LIN). Among these protocols, catoCAN protocol is a most widely used in-vehicle network protocol. The CAN (Controller Area Network) protocol is a serial bus communication system used in the internal network of vehicles. Unlike the traditional point-to-point (P2P) communication, the CAN protocol utilizes a broadcasting method where multiple Electronic Control Units (ECUs) within the vehicle transmit frames through a central CAN bus. This enables fast and efficient communication, leading to cost reduction and improved performance in the vehicle's internal network. The CAN protocol was standardized by ISO (International Standardization Organization) in 1993 under the ISO 11898 standard [3]. However, the CAN protocol for automotive use was not originally designed with security considerations in mind. As a result, several security vulnerabilities exist, such as the transmission of frames in plain text and the absence of proper node authentication [4]. Over the years, researchers have demonstrated various attacks exploiting these security vulnerabilities in the CAN protocol for automotive use [5–8]. These attacks include seizing control of vehicle functions remotely, injecting abnormal frames into the CAN bus to disrupt the normal operation of ECUs, and emphasizing the need for security enhancements in the CAN protocol for automotive use. To improve the security vulnerabilities of the CAN bus, we propose an intrusion detection system utilizing LSTM (Long Short-Term Memory)-autoencoders to detect abnormal frames transmitted within the in-vehicle network. The proposed system utilizes two types of network behavior features, transmission interval and data changes, in combination to detect anomalies. The

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 July 2023 doi:10.20944/preprints202307.0062.v1

input data for each type of feature, comprises individual ID-based frame streams. The ID-based frame stream is a continuous sequence of frames sharing the same Arbitration identifier (ID), and the original frame sequence is re-organized to incorporate the normalcy of network behaviors for different frame types into the LSTM-Autoencoder model.

The paper is organized as follows: Section 2 provides an overview of the CAN protocol used in in-vehicle communication networks, highlighting the security threats that arise due to its vulnerabilities. In Section 3, existing research on in-vehicle CAN Intrusion Detection Systems (IDS) is summarized. Section 4 describes the overall structure of the proposed LSTM-Autoencoder-based CAN IDS, outlining the roles of each component. Detailed experiments and performance evaluations of the intrusion detection models based on the characteristic features are presented in Section 5. Finally, Section 6 highlights our contribution and concludes the paper by summarizing the research, discussing future directions.

2. Preliminaries

2.1. CAN protocol basics

The CAN protocol facilitates efficient bidirectional communication among Electronic Control Units (ECUs). It operates as a serial bus communication channel within the vehicle's internal network, where frames are broadcasted through a central bus. This enables all connected nodes to transmit and receive frames without relying on source or destination addresses. Instead, each frame includes unique identifiers that indicate the priority and functionality for medium access control [3]. The structure of a CAN data frame is illustrated in Figure 1. Among the frame types in the CAN protocol, including data frame, remote frame, overload frame, and error frame, this explanation will primarily focus on the data frame, which is commonly used for data exchange between ECUs. As shown in the figure, a CAN data frame comprises five main fields: an arbitration field, a control field, a data field, a check field, and an ACK field. The control field, check field, and ACK field serve to accurately interpret the frame format and verify transmission integrity. In contrast, the arbitration field contains the Identifier, which uniquely identifies the frame and determines its priority. The data field carries the payload or actual data of the frame, supporting up to 8 bytes of data.

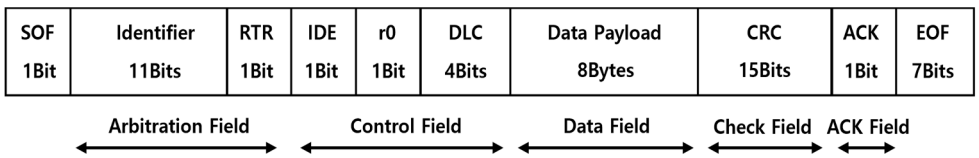


Figure 1. CAN data frame fields.

2.2. Threats on CAN protocol

The CAN frames are transmitted in a broadcast way, so it is not necessary for frames to have source and destination addresses. Besides, the CAN protocol has the problem of its frames not including any authentication data Therefore, it is impossible to identify illegitimate node using any information that the frame itself contains. Additionally, due to the lack of encryption, even legitimate nodes transmit frames in plain text, exposing them to various security threats. Since all nodes connected to a bus can receive broadcasted frames, it is possible to access and inspect all unencrypted frames. There are four kinds of threat presented in many research results [9,10]. Each threat is explained in turn, and our proposed method is designed to prevent the threats by capturing abnormal behaviors.

- Spoofing Attack: Unauthorized attackers can disguise malicious nodes as legitimate nodes and transmit malicious frames.
- Fuzzing Attack: Unauthorized malicious nodes can randomly insert invalid data into the vehicle network, causing confusion in the functionality of legitimate Electronic Control Units (ECUs).

 doi:10.20944/preprints202307.0062.v1

- Sniffing Attack: When a legitimate node transmits frames on the CAN bus without encryption, malicious attackers can eavesdrop on the frames.
- Replay Attack: Malicious attackers can capture frames transmitted in plaintext from an ECU responsible for specific critical functions (e.g., engine shutdown, brake operation) and resend the same frame to a node with a higher priority CAN ID during vehicle operation.

3. Related work

The CAN bus used in vehicles has the advantage of efficiently and rapidly delivering frames transmitted by numerous Electronic Control Units (ECUs) within the vehicle through a broadcasting method. However, due to the lack of security considerations in its design, there are inherent vulnerabilities that expose it to various security threats. As a result, several research efforts have been proposed to protect the CAN bus. In this section, the previous studies related to the CAN security are introduced in turn.

C.Miller and C.Valasek demonstrated CAN bus security vulnerabilities by hacking a Jeep Cherokee vehicle, gaining remote control over critical functions such as brakes, accelerator, and ignition, as well as air conditioning and radio control. This incident emphasized the need for CAN bus security [8]. Wang et al. [11] proposed an entropy-based CAN IDS (Intrusion Detection System) that detects attacks by analyzing the probability and entropy of each CAN ID, detecting significant deviations from normal entropy. Olufowbi et al. [12] developed a timing-based CAN IDS by analyzing transmission intervals and response times of CAN frames, considering frames abnormal if arrival times exceed threshold ranges Jerin et al. [13] designed a hybrid approach CAN IDS using sequential patterns of CAN ID sequences and arrival time intervals to detect attacks.

D. Stabili et al. [14] proposed a low-complexity intrusion detection algorithm for identifying malicious CAN messages in vehicles' CAN bus. It evaluates payload sequences using the Hamming distance and experiments with real CAN traffic. P.S. Murvery et al. [15] focus on authenticating nodes on the bus by analyzing physical characteristics of frames. They achieve accurate identification through voltage measurements and careful choices of transceivers and frame IDs. K.T. Cho et al. [16] introduced Clock-based IDS (CIDS) for anomaly-based intrusion detection, using periodic message intervals to fingerprint ECUs. CIDS detects abnormal shifts in identification errors using Cumulative Sum (CUSUM) based on a baseline of ECUs' clock behaviors. Researchers also suggest verifying message contents and frequency. Delwar et al. [17] presented an LSTM-based CAN IDS, training LSTM classifiers with normal and attack data features. Their approach utilizes LSTM similar to ours, but it is not an autoencoder-based or unsupervised anomaly detection method. A. Derhab [18] proposed H-IDFS, a Histogram-based Intrusion Detection and Filtering framework. It uses histograms of CAN packets organized into windows and a multi-class IDS classifier to identify malicious traffic windows. S.U. Sagong et al. [19] proposed three voltage-based attacks: overcurrent, denial-of-service, and forced retransmission. To defend against these attacks, they also introduced a hardware-based Intrusion Response System (IRS) that disconnects the VIDS from the CAN bus. The authors of [20] evaluates entropy-based anomaly detection algorithms in modern vehicle networks. Simulated attacks involved injecting various forged CAN messages into traces from a licensed vehicle. The results show that detecting attacks with a high volume of forged CAN messages is possible using entropy-based detection.

4. LSTM-Autoencoder based CAN IDS

The overall structure of the LSTM-Autoencoder based intrusion detection system is illustrated in Figure 2. The proposed system consists of three components: feature sequence extractor, LSTM-Autoencoder model, anomaly detector. The feature sequence extractor begins by separating the entire sequence of frames arranged in chronological order into streams based on their IDs. It then extracts the sequence of time intervals and the sequence of hamming distances for each individual stream. The LSTM-Autoencoder models accept the extracted feature sequences and analyze it for generating the reconstructed sequences. After the reconstructed time interval sequence and the hamming distance sequence are generated by the models, The anomaly detector calculates the dissimilarity

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 July 2023 doi:10.20944/preprints202307.0062.v1

between the original sequences of time intervals and hamming distances, and the corresponding reconstructed sequences. The anomaly detector utilizes dissimilarity to determine whether the given frame sequence is normal or abnormal. The detailed explanations about each component are included in the next subsections.

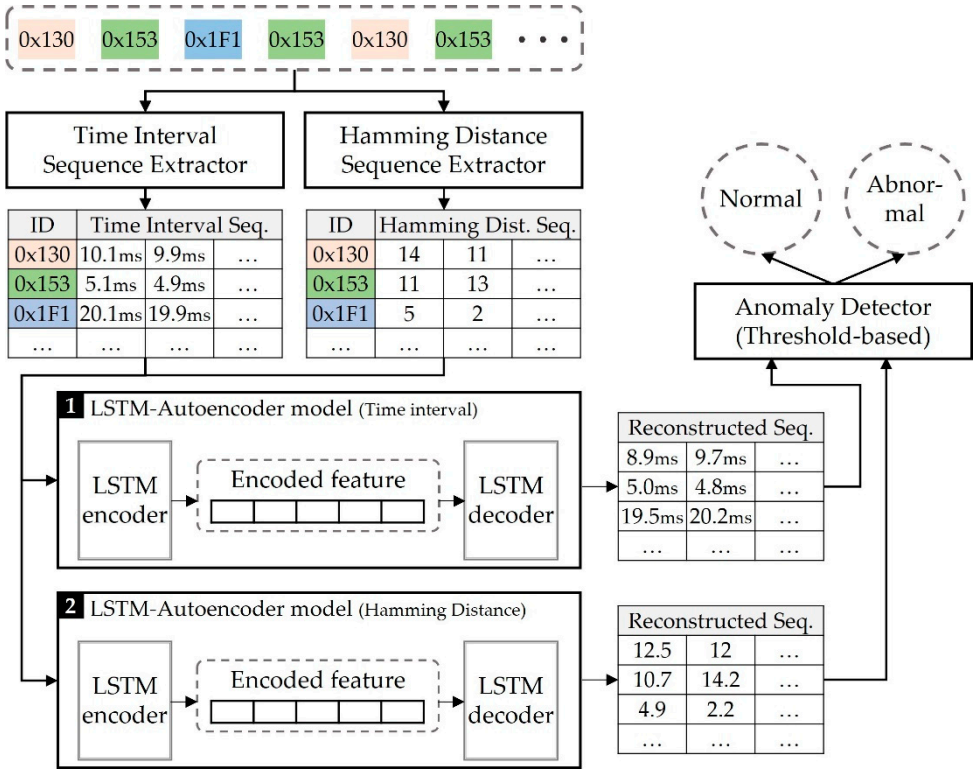


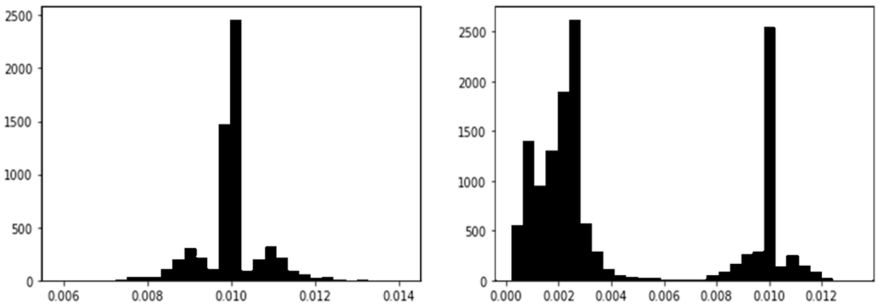
Figure 2. Overall architecture of the proposed system.

4.1. Time Interval Sequence Extractor

Among many types of CAN frame transmitted in in-vehicle network, there are many periodically transmitted frames which are used to poll any sensor data or to command some functions, and event-based frames often exhibit a consistent periodicity as well. Therefore, if attacks such as DoS and Fuzzing are performed by injecting malicious frames, there is a high possibility that the predefined periodicity for each CAN frame may be disrupted. Figure 3 shows an example of the counts per time interval, measured from the dataset provided by [21]. Comparing the transmission time intervals of a dataset consisting only of normal data without any attacks, and a dataset containing attacks on frames with CAN ID, 0x316 with an average transmission period of 0.01 seconds, it is observed that the dataset with attacks exhibits transmission time intervals that deviate significantly from the average transmission period. These deviating data points indicate the presence of attack types that insert frames, causing an impact on the normal frame transmission period. In this point of view, we consider transmission time intervals of consecutive frames with same ID as an important feature that defines normal network behavior, and the time interval sequence is used as a kind of feature for the LSTM-Autoencoder model.

The time interval sequence extractor first obtains all time intervals by subtracting timestamps of each pair of consecutive frames with same ID, and It generates a sequence for each ID by listing the time intervals in chronological order.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 July 2023 doi:10.20944/preprints202307.0062.v1



(**a**) (**b**)

Figure 3. An example of the histograms contain frame counts both normal situation and attack situation (**a**): Histogram generated from the dataset that includes only normal frames with 0x316; (**b**): Histogram generated from the dataset that includes normal and malicious frames with 0x316.

4.2. Hamming Distance Sequence Extractor

The hamming distance sequence extractor is responsible to compute hamming distances of pairs of consecutive frames' payloads in each ID-based stream. The hamming distance, which determines the number of differing bits between two bit- sequences of equal length is calculated as presented in (1). In (1), x and y are the bit-sequences, and their length is k.

$$HD = \sum_{i=1}^{k} |x_i - y_i| \quad (1)$$

If the lengths of the two bit-sequences differ, the shorter sequence is padded with zeros at the end to ensure equal length. The computed hamming distances for each ID-based stream are organized in chronological order to form a sequence.

The purpose and necessity of defining the hamming distance of data payloads for each ID are as follows: Many normal frames being transmitted for regular functions have characteristics where the frame's data payload remains unchanged or certain byte values of the data payload remain fixed and unchanged during transmission. This can be attributed to the fact that ECUs responsible for various functions in the in-vehicle network transmit frame data commanding specific functions with unique IDs. In addition, there are frames in which the payload value consistently increases or decreases by a fixed delta. For instance, the CAN protocol utilizes an alive counter value or sequence number to offer End-to-End protection (E2E). The alive counter value alternates between zero and one, or vice versa, while the sequence number consistently increments by one, except when transitioning from the maximum value back to zero.

Due to the property of payload values remaining unchanged or exhibiting a consistent degree of variation in usual situations, we utilize the hamming distance as a crucial feature. When attackers performing reverse engineering for functionality analysis through fuzzing, there are cases where we are compelled to inject arbitrary values into the payload. In such instances, the amount of variation in the payload values often deviates significantly from that observed in normal cases.

Tables 1 and 2 show the changes in payload values caused by a fuzzing attack. These tables present the frame sequences of CAN ID 316, which is utilized for fuzzing attacks, in both normal and attack situations. These examples are extracted from the dataset obtained from [21]. In Table 1, it can be observed that five out of eight bytes in the data payload, excluding the second, third, and fifth bytes, consistently transmit the same data, while in the case of abnormal data containing attack frames in Table 2, completely different data or changed data in fixed bytes are transmitted, which is distinct from the normal data.

Table 1. CAN frames with ID, 0x316 in normal situation.

ID	DLC	Data Payload							
316	8	5	1F	84	9	1F	1D	0	7B

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 July 2023 doi:10.20944/preprints202307.0062.v1

316	8	5	1E	84	9	1E	1D	0	7B
316	8	5	1E	88	9	1E	1D	0	7B
316	8	5	1F	88	9	1F	1D	0	7B
316	8	5	1E	7C	9	1E	1D	0	7B
316	8	5	1F	7C	9	1F	1D	0	7B
316	8	5	1F	70	9	1F	1D	0	7B
316	8	5	1F	70	9	1F	1D	0	7B
316	8	5	1F	70	9	1F	1D	0	7B

Table 2. CAN frames with ID, 0x316 in fuzzing attack situation.

ID	DLC	Data Payload							
316	8	8	5	1E	7C	9	1E	1D	0
316	8	8	5	1F	7C	9	1F	1D	0
316	8	8	5	1F	70	9	1F	1D	0
316	8	8	45	29	24	FF	29	24	0
316	8	8	45	29	24	FF	29	24	0
316	8	8	5	1F	84	9	1F	1D	0
316	8	8	45	29	24	FF	29	24	0
316	8	8	5	1E	84	9	1E	1D	0
316	8	8	45	29	24	FF	29	24	0

4.3. LSTM-Autoencoder

The feature extractor is responsible for extracting time interval sequences and hamming distance sequences from ID-based streams within the original frame sequence. To accommodate the fixed size of input sequences required by our LSTM-Autoencoder model, each time interval sequence and hamming distance sequence are divided into smaller subsequences. It is important to note that analyzing same length of sequence in real-time detection with that of model training usually perform better than the other cases. However, gathering a large number of frames within a short time for real-time detection can be challenging, potentially leading to reduced responsiveness. Furthermore, network attacks tend to occur rapidly. Consequently, the feature sequences are segmented into arbitrary time steps for effective utilization.

Our LSTM-Autoencoder model accepts three-dimensional data in the format of (Timestep, The number of features (IDs), Batch size). In our model, the number of features corresponds to the number of IDs, while the time steps represent the length of the sequence, which aligns with the number of cells in the input layer of the LSTM-Autoencoder. Lastly, the batch size refers to the number of data inputs into the LSTM network.

Figure 4 shows the overall architecture of the LSTM-Autoencoder models used for analyzing time-interval sequences and hamming distance sequences. Each LSTM-Autoencoder consists of encoder, repeat vector layer, decoder, fully-connected layer. The first part of the LSTM-Autoencoder is the encoder. It takes an input sequence and encodes it into a lower-dimensional representation. In the case of the LSTM-Autoencoder, the encoder is typically composed of LSTM layers. Our encoder consists of two LSTM, and each LSTM layer captures temporal dependencies and encode its input into an encoded feature.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 July 2023 doi:10.20944/preprints202307.0062.v1

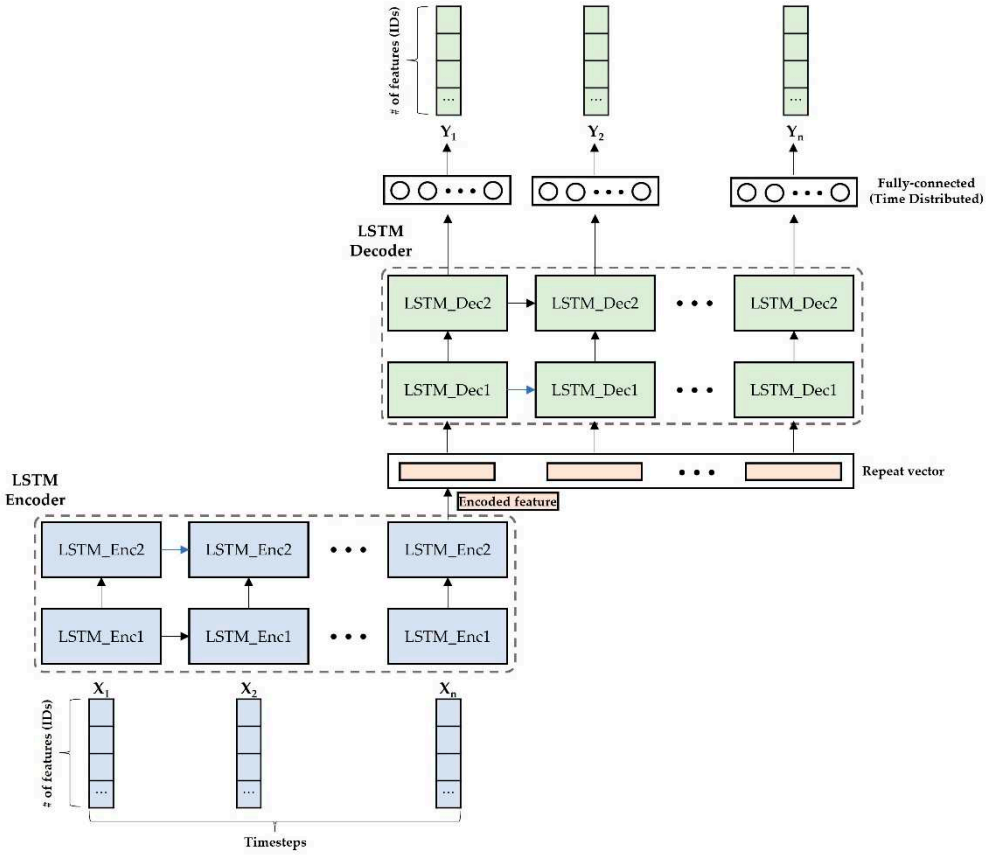


Figure 4. The architecture of the LSTM-Autoencoder model.

After the encoder compress the input to into the encoded feature, the repeat vector layer takes the encoded feature and repeat the feature timestep times. Then, the decoder takes the repeat vector that contains same data of the encoded feature and aims to reconstruct the original input sequence. The decoder is composed of two LSTM layers as the encoder is. The detailed information for hyperparameter setting is included in Table 3.

Table 3. The hyperparameters of our LSTM-Autoencoder (Timestep and # of IDs are variable).

Layer		Activation	Dropout rate	Output	Others
Encoder	LSTM	Tanh/Sigmoid	-	(Timestep, 128)	Optimizer: Adaptive Moment Estimation
	LSTM	Tanh/Sigmoid	0.25	(1,64)	
Repeat vector		-	-	(Timestep, 64)	
Decoder	LSTM	Tanh/Sigmoid	-	(Timestep,64)	Loss: Mean Squared Error
	LSTM	Tanh/Sigmoid	-	(Timestep,128)	
Fully-connected		Identity	0.25	(Timestep, # of IDs)	

As shown in the Table 3, it can be observed that the number of features decreases from 128 to 64 in the encoder and increases from 64 back to 128 in the decoder. To ensure that the shape of the reconstructed data, which is the output of the model, matches the original data, the repeat vector and fully-connected layers restore the timestep and feature size to their original values. The model's optimizer is set to Adaptive Moment Estimation (Adam), enabling automatic adjustment of the learning rate and momentum-based optimization. For calculating the reconstruction error between the reconstructed data and the original data, Mean Absolute Error (MAE) is utilized, which uses absolute values to compute errors, giving equal weight to all errors regardless of their magnitude.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 July 2023 doi:10.20944/preprints202307.0062.v1

4.4. Anomaly Detector

The anomaly detector performs the role of deciding whether the given CAN frame sequence is normal or not. The anomaly detector accepts the reconstructed time interval sequence and hamming distance sequence from the LSTM-Autoencoders. Afterwards, the reconstructed sequences are decomposed into stream units. Each divided sequence is then compared with the original sequence of the corresponding ID-based stream. The Mean Absolute Error (MAE) is utilized to compare the two sequences. When calculating the MAE value for each pair of sequences with the same ID, it is compared against a predefined threshold assigned to that specific ID. Only if all the MAE values do not exceed their respective thresholds, it can be concluded that there is no abnormal behavior detected.

5. Evaluation

In this section, we provide a description of the CAN dataset utilized in this study, and the experimental setup. Additionally, we present the experimental results and performance evaluation of two detection methods based on time interval feature and hamming distance changes in payload.

5.1. Dataset description and experimental environment

In this study, we analyzed CAN network traffic datasets obtained from [21]. The datasets can be categorized into two types. The first dataset comprises CAN frames collected under normal operating conditions, while the second dataset includes both CAN frames used for attacks and normal frames. As mentioned in [21], the dataset is publicly available, and the CAN frames were collected during the operation of a Hyundai YF Sonata 2010 model. Table 4 presents the number of CAN data frames for each dataset.

Table 4. Experimental dataset configuration.

Data Type	# of Normal Frames	# of Abnormal Frames
Normal set	117,173	0
Abnormal set	116,677	15,974

The normal dataset consists of 117,173 normal frames, while the abnormal dataset is composed of 116,677 normal frames and 15,974 abnormal frames that involve message injection attacks targeting CAN IDs 0x316 and 0x43F. During the training phase of the LSTM-Autoencoder models, 70% of the normal dataset, specifically the frame data, was used. For the testing phase, we utilized 0x316 stream and 0x43F stream from 30% of the normal dataset and the entire abnormal dataset. This allowed us to evaluate the models' performance on both normal and abnormal instances.

5.2. Performance metrics

The performance evaluation of the detection model in this study was conducted using various metrics, including accuracy, precision, recall, and F_1-score. These metrics are fundamental measures for assessing the effectiveness of the model in detecting anomalies. The calculations for these metrics are outlined in (2-5), which provide a quantitative assessment of the model's performance based on the detected anomalies and the ground truth labels. The accuracy metric measures the overall correctness of the model's predictions, while precision indicates the proportion of correctly identified anomalies out of the total anomalies predicted. Recall, also known as sensitivity or true positive rate, represents the proportion of correctly detected anomalies out of the actual anomalies present in the dataset. Lastly, the F_1 score combines precision and recall into a single metric, providing a balanced measure that considers both the false positives and false negatives.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (2)$$

$$Precision = \frac{TP}{TP + FP} \quad (3)$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 July 2023 doi:10.20944/preprints202307.0062.v1

$$Recall = \frac{TP}{TP + TN} \quad (4)$$

$$F_1 score = \frac{2(Precision \times Recall)}{Precision + Recall} \quad (5)$$

Each performance metric is calculated using True Positive (TP), False Negative (FN), False Positive (FP), and True Negative (TN). TP measures the correct detection of anomalies, while FN represents misclassifying anomalies as normal instances. FP denotes predicting normal instances as anomalies, and TN signifies the accurate identification of normal instances.

5.3. Anomaly detection accuracy

Table 5 displays the performance measurements for our anomaly detection system. Our system utilizes two LSTM-Autoencoders, one analyzing the transmission timing and the other analyzing data changes, for anomaly detection. Therefore, we evaluated the performance metrics of each LSTM-Autoencoder individually. According to the table, On average, the precision was 0.99, recall was 0.86, F-measure was 0.91, and accuracy was 0.87. In addition, the LSTM-Autoencoder that utilizes the time interval sequence feature achieved a remarkably high detection accuracy of 0.99. However, the LSTM-Autoencoder that uses the hamming distance sequence feature demonstrated relatively lower accuracies of 0.70 and 0.78.

Table 5. Performance measurement value for each LSTM-Autoencoder models.

Feature	ID	TN	FP	FN	TP	Precision	Recall	F_1 score	Accuracy
Time	0x316	5,941	2	0	13,876	0.99	1	0.99	0.99
interval	0x43F	5,939	14	3	13,873	0.99	0.99	0.99	0.99
Hammin	0x316	5,925	18	4563	9,313	0.99	0.67	0.80	0.70
g distance	0x43F	5,694	249	3212	10,664	0.98	0.77	0.86	0.78

While the detection rate of the hamming distance-based LSTM-Autoencoder itself may not be very high, the low number of false positives implies that even if some attack segments are missed, sustained attacks over a certain period can still be detected. For example, in a 10-second attack, if an attack segment of approximately 3 seconds is not detected, it can be considered a missed detection. However, since the attack segments are not continuously consecutive but rather scattered in time, it is still meaningful and possible to detect them adequately.

A more critical factor to consider is the false positive (FP) rate, which is a very low value in this case. If normal frames in the vehicle's internal network are mistakenly classified as attacks and trigger warnings or blocking mechanisms, it can lead to unexpected vehicle malfunctions. Therefore, the false positive rate is a crucial metric. In our model, the detection threshold is set to minimize false positives. Specifically, the maximum of the average absolute error calculated for each ID-based stream is used as the detection threshold to distinguish between normal and abnormal messages for detection purposes.

6. Discussion

Through this research, we have discovered that utilizing an unsupervised learning model called LSTM-Autoencoder allows us to define normalcy based solely on the vehicle's internal network traffic and perform anomaly detection effectively. This approach proves to be an efficient method by our experiments. The study introduces a method for processing input data, preserving sequential information. Based on the inherent characteristics of the protocol where each CAN frame is assigned an Arbitration Identifier (ID), we propose a method for decomposing the entire frame sequence into streams consisting of frames with the same ID. For each stream, we extract features and arrange them in chronological order. This approach allows us to preserve the structural properties of the frames while analyzing the behavior of each stream.

There are still unresolved issues in our research. One of them is the existence of various and numerous feature information beyond the characteristics of the transmission timing and payload

data rate. For sequentializable information, it seems possible to apply them to the LSTM-Autoencoder approach we are proposing. For example, features such as the variance or entropy representing the dispersion or uncertainty of frame IDs within a certain time period, or the correlation between data, can be defined. Therefore, we plan to continue our research by devising methods to extract multiple features that are useful in capturing abnormal network behavior in vehicle internal communication and applying them to existing detection models.

Author Contributions: Conceptualization, T.G.K. and I.S.Y.; methodology, T.G.K. and I.S.Y.; validation, I.S.Y.; investigation, T.G.K. and J.Y.K; writing—original draft preparation, T.G.K.; writing—review and editing, T.G.K. and I.S.Y.; visualization, J.Y.K.; supervision, I.S.Y.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: This work was supported by the Soonchunhyang University Research Fund

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Leen, G.; Heffernan, D. Expanding automotive electronic systems. Computer 2002, 35, 88-93.
2. Huang, S. C.; Chen, B. H.; Chou, S. K.; Hwang, J. N.; Lee, K. H. Smart car [application notes], IEEE Computational Intelligence Magazine 2016, 11, 46-58.
3. HPL, S. C. Introduction to the controller area network (CAN). Application Report SLOA101 2002, 1-17.
4. Carsten, P.; Andel, T. R.; Yampolskiy, M.; McDonald, J. T. In-vehicle networks: Attacks, vulnerabilities, and proposed solutions. In Proceedings of the 10th Annual Cyber and Information Security Research Conference, pp. 1-8.
5. Koscher, K.; Czeskis, A.; Roesner, F.; Patel, S.; Kohno, T.; Checkoway, S.; Savage, S. Experimental security analysis of a modern automobile, In Proceedings of the 2010 IEEE symposium on security and privacy, pp. 447-462. IEEE.
6. Hoppe, T.; Dittman, J. Sniffing/Replay Attacks on CAN Buses: A simulated attack on the electric window lift classified using an adapted CERT taxonomy. In Proceedings of the 2nd workshop on embedded systems security (WESS), pp. 1-6.
7. Checkoway, S.; McCoy, D.; Kantor, B.; Anderson, D.; Shacham, H.; Savage, S.; Kohno, T. Comprehensive experimental analyses of automotive attack surfaces. In Proceedings of the 2011 USENIX security symposium pp. 2021, pp. 447-462.
8. Miller, C.; Valasek, C. (2015). Remote exploitation of an unaltered passenger vehicle. In Proceedings of the 2015 Black Hat, USA, 2015, pp. 1-91.
9. Sun, H.; Chen, M.; Weng, J.; Liu, Z.; Geng, G. Anomaly detection for in-vehicle network using CNN-LSTM with attention mechanism. IEEE Transactions on Vehicular Technology 2021, 70, pp. 10880-10893.
10. Song, H. M.; Kim, H. R.; Kim, H. K. Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network. In Proceedings of the 2016 international conference on information networking (ICOIN), pp. 63-68.
11. Wang, Q.; Lu, Z.; Qu, G. (2018, September). An entropy analysis based intrusion detection system for controller area network in vehicles. In Proceedings of the 31st IEEE International System-on-Chip Conference (SOCC), pp. 90-95.
12. Olufowobi, H.; Young, C.; Zambreno, J.; Bloom, G. SAIDuCANT: Specification-based automotive intrusion detection using controller area network (CAN) timing. IEEE Transactions on Vehicular Technology 2019, 2, pp. 1484-1494.
13. Sunny, J.; Sankaran, S.; Saraswat, V. A hybrid approach for fast anomaly detection in controller area networks. In Proceedings of the 2020 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), pp. 1-6.
14. Stabili, D.; Marchetti, M.; Colajanni, M. Detecting attacks to internal vehicle networks through Hamming distance. In Proceedings of the 2017 AEIT International Annual Conference, pp. 1-6.
15. Murvay, P. S.; Groza, B. Source identification using signal characteristics in controller area networks. IEEE Signal Processing Letters 2014, 4, pp. 395-399.
16. Cho, K. T.; Shin, K. G. Fingerprinting electronic control units for vehicle intrusion detection. In Proceedings of the 2016 USENIX Security Symposium, 40, pp. 911-927.
17. Hossain, M. D.; Inoue, H.; Ochiai, H.; Fall, D.; Kadobayashi, Y. (2020, July). Long short-term memory-based intrusion detection system for in-vehicle controller area network bus. In 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC) (pp. 10-17). IEEE.

18. Derhab, A.; Belaoued, M.; Mohiuddin, I.; Kurniawan, F.; Khan, M. K. Histogram-based intrusion detection and filtering framework for secure and safe in-vehicle networks. IEEE Transactions on Intelligent Transportation Systems 2021, 3, pp. 2366-2379.
19. Sagong, S. U.; Ying, X.; Poovendran, R.; Bushnell, L. Exploring attack surfaces of voltage-based intrusion detection systems in controller area networks. In Proceedings of the 2018 ESCAR Conference, pp. 1-13.
20. Müter, M.; Asaj, N. Entropy-based anomaly detection for in-vehicle networks. In Proceedings of the 2011 IEEE Intelligent Vehicles Symposium, pp. 1110-1115.
21. Han, M. L.; Kwak, B. I.; Kim, H. K. Anomaly intrusion detection method for vehicular networks based on survival analysis. Vehicular communications 2018, 14, 52-63.