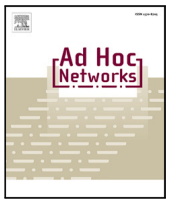
Ad Hoc Networks 169 (2025) 103737

Contents lists available at ScienceDirect

Ad Hoc Networks

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



Enhancing real-time intrusion detection system for in-vehicle networks by employing novel feature engineering techniques and lightweight modeling

Wael Aljabri *, Md. Abdul Hamid, Rayan Mosli

King Abdulaziz University, Faculty of Computing and Information Technology, Jeddah, 21589, Saudi Arabia

ARTICLE INFO

Keywords:
Vehicular networks
Controller area network
Intrusion detection systems

ABSTRACT

Autonomous vehicles are built using a variety of electronic control units (ECUs) that communicate over a controller area network (CAN). A CAN enables the communication of data between ECUs to guarantee safety, assist drivers, and perform different functions. Nevertheless, a CAN lacks built-in security measures, which makes it susceptible to cyberattacks. A significant amount of existing research on intrusion detection systems (IDSs) is aimed at enhancing the security of a CAN by identifying and detecting unauthorized packet injections. However, the majority of machine/deep learning-based IDSs have difficulty sufficiently addressing latency. To address this issue, we propose a novel IDS framework that introduces two distinctive features. The first feature is the utility of data entropy, which is dynamically recalculated as new data arrives to capture unpredictable variations in the data payload. The second feature is an anomaly score, combining data entropy and time interval entropy to detect abnormal patterns in CAN communication. We validated the significance of these features using SHapley Additive exPlanations (SHAP) analysis. These features are integrated into a lightweight deep learning-based IDS model, specifically designed for resource-constrained environments. This integration significantly improves detection accuracy and operational efficiency. Our approach is validated using two well-known public datasets, car hacking: attack & defense challenge and car-hacking datasets. It shows significant detection capabilities with accuracies of 0.9946 and 0.9995 and F1 scores of 0.9945 and 0.9995, respectively. Also, our IDS achieves an effectively low inference latency of only 0.17 milliseconds, surpassing the performance of existing machine/deep learning-based IDSs.

1. Introduction

The rise of intelligent transportation systems (ITSs) marks significant developments in the use of communication and information technology to enhance safety and efficiency in transportation networks. ITSs center around autonomous vehicles (AVs); they have the potential to revolutionize transportation systems by mitigating human errors, thereby decreasing dependence on human drivers, decreasing fatalities caused by traffic accidents, lowering transportation expenses, and facilitating efficient vehicle mobility, thus optimizing traffic flow on the road.

The large number of electronic control units (ECUs), sensors, and actuators in AVs has led to the rise of a wide range of in-vehicle networking technologies. The majority of ECUs depend on signal-based communication using automotive bus technologies, such as a controller area network (CAN), FlexRay, ethernet, or a local interconnect network (LIN). A CAN is a widely used automotive networking technology [1]. It is a serial communications protocol that is well suited for real-time applications that necessitate reliable communication in challenging environments. Due to its affordability and exceptional reliability, this communication protocol has remained popular [2,3]. A CAN is utilized throughout a range of vehicles, including commercial trucks, cars, agricultural vehicles, boats, and even aircraft [1].

Despite its widespread adoption, a CAN protocol possesses vulnerabilities that make it susceptible to exploitation. Real-time performance limitations and restricted payload capacities pose significant obstacles to several in-vehicle networking technologies, including CAN systems. Therefore, traditional security techniques, such as network segmentation, encryption, and authentication, are not effective. These vulnerabilities in AVs make them susceptible to a variety of cyberattacks [4]. Because a CAN protocol transmits all messages in plaintext format and utilizes a CAN ID to identify the source of a message, this gives rise to several security risks. For instance, spoofing is feasible because every participant in a network can create a CAN message using the ID of any other node, hence enabling the spoofing of other ECUs.

A case that clearly demonstrates the actual risks posed by CAN bus vulnerabilities is the documented cyberattack on a Jeep Cherokee [5].

* Corresponding author.
E-mail addresses: waljabri0015@stu.kau.edu.sa (W. Aljabri), mabdulhamid1@kau.edu.sa (Md. Abdul Hamid), rmosli@kau.edu.sa (R. Mosli).

https://doi.org/10.1016/j.adhoc.2024.103737
Received 21 May 2024; Received in revised form 13 October 2024; Accepted 5 December 2024
Available online 12 December 2024

In this breach, attackers remotely gained control of critical vehicle functions, including acceleration, steering, and braking, illustrating the severe safety risks of such exploits. This incident underscores the urgent need for robust defense mechanisms and advanced security measures [5]. Another example is the recent and significant security issue concerning the theft of Toyota RAV4 vehicles in the UK. Thieves took advantage of the weaknesses in a CAN bus system by disconnecting parts of the headlamp to insert harmful devices. These devices sent unauthorized commands to modify the communication network of the vehicle, allowing unauthorized access and the engine to start without using the physical key or remote control [6].

To tackle these issues, it is essential to develop efficient IDSs that have been designed specifically for in-vehicle systems. These IDSs must have the ability to detect and classify attacks in real time. Integrating deep-learning (DL) models into IDSs has been demonstrated to be a highly effective approach for detecting intrusions in in-vehicle networks. The integration highlights the essential role of DL in ensuring AVs' security and reliability [7].

This paper presents a novel multi-class IDS specifically designed for in-vehicle networks. The IDS employs innovative feature engineering, which enables a lightweight DL model to achieve significant performance. The proposed IDS went through an evaluation using multiple datasets to address a wide variety of attacks. The time efficiency of our system was evaluated in a computing environment with limited resources, making it ideal for conducting an efficient evaluation. The main contributions are as follows:

1. We propose novel dynamic detection features that are specifically extracted from the CAN frame to detect cyberattacks in in-vehicle networks. The two novel features are data entropy and an anomaly score derived from the combined entropies of both data and time intervals.
2. We propose a lightweight DL model, making it well suited for use in environments with limited resources. Our study demonstrates that integrating these novel features with the lightweight model leads to a solution that is both effective in detection performance and computationally efficient.
3. We have created a system that allows for real-time intrusion detection for in-vehicle networks. The proposed IDS achieves a detection inference time of only 0.17 ms and a significantly low latency when compared to the current methods.

The rest of this paper is structured as follows: Section 2 explains the CAN, highlighting its important features, and the specific DL model utilized in our research. Section 3 examines the relevant research on techniques for CAN IDSs based on deep learning. Section 4 provides a comprehensive explanation of the proposed IDS framework. It includes a detailed description of the novel feature engineering methodology and the model architecture. Section 5 provides a comprehensive overview of the experimental results obtained from all utilized mechanisms. Additionally, it includes an evaluation of the proposed IDS compared to other existing systems. This section also includes an evaluation of time performance and compares its inference time with existing IDSs. Additionally, SHAP analysis is included to highlight the importance of the proposed novel features. Section 6 concludes our work by providing conclusions and future research directions based on this study's findings.

2. Background

2.1. Controller area network

A CAN utilizes a bus topology, known as a CAN bus, to facilitate communication between ECUs. The CAN specification Version 2.0, which was established by Bosch in 1986 [8], is employed in modern vehicles. A CAN facilitates distributed real-time communication within vehicles. Due to its low cost and excellent reliability, this communication protocol has always maintained its popularity [1].

A CAN is capable of supporting two different formats: standard and extended. The standard format uses an 11-bit identification, but the extended format uses a 29-bit identifier that consists of an 11-bit identifier plus an additional 18-bit extended identifier. Typically, modern commercial vehicles follow the standard format, whereas the 29-bit extended format is mostly utilized for specialty vehicles, such as agricultural machines and semi-trucks.

CAN packets primarily consist of the following fields:

- An arbitration identifier (ID) is responsible for managing the transmission of messages that occur simultaneously in a media access control procedure. In this arbitration, an arbitration ID with a lower value is given higher priority.
- The Remote Transmission Request (RTR) bit is a 1-bit field in the CAN protocol used to distinguish between data frames and remote frames.
- The data length code (DLC) is a 4-bit integer that shows the number of bytes in a data field. The value of a DLC can range from 0 to 8.
- Data field: A data field holds the specific sensor control data of a vehicle that has to be transmitted. It can range from 0 to 8 bytes, depending on the value specified in a DLC field.
- The cyclic redundancy check (CRC) is a method used to calculate and verify the integrity of a message by generating a checksum.

This study aimed to utilize particular components of the CAN frame to improve the intrusion detection algorithm. The utilized components of the CAN frame were the arbitration ID and the data field. To enhance our ability to detect anomalies, we introduced two novel detection features: data entropy and anomaly scores. Data entropy measures the level of unpredictability in the data field. The anomaly score was calculated for every input based on the combined indicators. The purpose of these advanced detection features is to improve the detection of malicious activity occurring within a CAN network.

2.2. Deep learning techniques

DL approaches are characterized by their ability to utilize nonlinear processing of interconnected logical units across several layers. The outputs of one layer are used as inputs for the following layer in a cascading manner. These techniques are compatible with both supervised and unsupervised procedures. Specifically, we employed deep neural networks (DNN), which are neural networks that consist of multiple hidden layers placed between input and output layers. DNNs can apply nonlinear classifiers and construct models via the sequential combination of basic elements [9].

In this study, we examined the utilization of a multilayer perceptron (MLP) classification algorithm in the context of deep learning. The MLP is a type of feedforward artificial neural network. It enables the utilization of spatial separators and introduces one or many hidden layers [10]. The MLP evaluates outputs by aggregating various inputs using a linear weighted transfer function [11]. An output layer provides classification results for the mapped inputs.

3. Related works

Recent studies on IDS for in-vehicle networks have investigated several approaches to achieving a balance between a system's requirements and its ability to detect intrusions. This balance is crucial for the effective implementation of an IDS in AVs. Seo et al. [12] and Song et al. [13] utilized advanced neural network architectures to improve the detection capabilities of in-vehicle IDSs. Seo et al. employed generative adversarial networks (GANs) to differentiate between known and novel attack types [12], while Song et al. adjusted a deep convolutional neural network (DCNN) based on the Inception-ResNet

architecture, which was originally designed for image classification to identify different spoofing attacks [13]. These models are designed to handle and analyze complex patterns of network traffic in vehicles, resulting in a high level of accuracy in detecting hostile activities.

To evaluate the performance of an IDS in terms of processing-delay and power consumption, the study conducted by Khandelwal et al. [14] incorporated a hybrid field-programmable gate array (FPGA) device, specifically the Zynq Ultrascale+. The study examined the actual use of an IDS in real-time vehicular environment. It evaluated the effectiveness of integrating a quad-core ARM processor with programmable logic in efficiently performing IDS operations [14]. Similarly, Cheng et al. [15] developed the temporal convolutional attention network (TCAN-IDS), which integrates a temporal convolutional network with a global attention mechanism. This model improves an IDS's ability to focus on crucial spatial–temporal characteristics inside a vehicular network. This incorporates attention methods employed to enhance data processing, which could affect a system's operational requirements [15].

Agrawal et al. utilized convolutional neural networks (CNNs) and long short-term memory (LSTM) networks to identify anomalies in CAN data [16]. Their approach includes preprocessing data using a sliding window technique, which facilitates the transformation of real-time vehicle network data into sequences that can be inspected for anomalies. This approach prioritizes speeding the model training process while specifically targeting the requirements of embedded systems, such as those present in AVs [16]. Amato et al. explored the utilization of neural networks (NNs) and MLP in the detection and classification of various cyberattacks, including DoS and fuzzy pattern attacks [17]. The methodology employed in this study combines a real-world dataset with simulated attacks, creating a varied environment for the evaluation of an IDS. The study evaluated various configurations of MLP models, with a specific focus on metrics, such as precision, recall, and the Matthews correlation coefficient. The comprehensive evaluation showed different levels of achievement, offering valuable insights into the practical difficulties and possibilities of neural network models in a vehicular IDS [17].

Desta et al. [18] and Hossain et al. [19] focused on employing LSTM networks to effectively manage the temporal and sequential data attributes of a CAN bus. Both studies demonstrated the ability of LSTM models to efficiently handle high-dimensional data. Desta et al.'s solution tackles two primary obstacles: minimizing the need for a deep knowledge of a CAN bus protocol and effectively handling the complex nature of data [18]. The work conducted by Hossain et al. involved the generation of a unique dataset from an experimental vehicle, including both regular operational data and injected cyberattack data [19]. The LSTM-based IDS is specifically designed to capture the dynamic nature of vehicular network communications, hence improving their ability to identify abnormal activities over time [19].

Jeong et al. presented a new method for extracting features from CAN frames that focuses on certain characteristics, such as the inter-frame space and counter information found within CAN packets [20]. These characteristics are then employed in decision tree-based methods, such as Random Forest and XGBoost, to detect malicious injected CAN packets. The primary objective is to boost detection rates by enhancing feature selection while also ensuring that a model remains lightweight and suited for real-time processing [20].

Several studies have shown significant levels of detection performance [21–24]. However, their complexity and significant computation overhead make them challenging to use in AVs. For example, Yang et al.'s IDS involves several million learning parameters [21]. Hence, they are not considered lightweight models. Despite significant progress in IDS research on in-vehicle networks, a crucial need still exists. Achieving an ideal balance between detection performance and time efficiency remains a challenge. Several current approaches are effective at detecting potential attacks, but they lack speed or computational efficiency. This limitation may restrict their practical implementation in real-life situations.

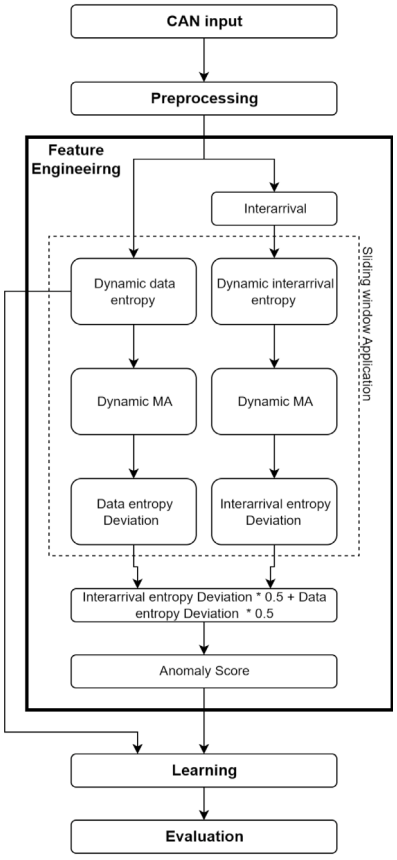


Fig. 1. Proposed IDS framework for CAN bus systems, incorporating dynamic feature engineering.

4. Proposed IDS framework

The IDS framework illustrated in Fig. 1 introduces a novel approach to real-time intrusion detection for CAN bus systems. It incorporates dynamic feature engineering techniques, focusing specifically on data entropy and anomaly score as key detection features. Unlike traditional methods, our framework integrates real-time calculations of data entropy and inter-arrival time entropy, followed by their dynamic moving averages. These novel detection features are dynamically calculated using a sliding window, which significantly improves the system's ability to detect intrusions in resource-constrained environments. Fig. 1 reflects the streamlined methodology, which begins with data preprocessing, progresses through feature extraction, and concludes with model training and comprehensive performance evaluation. Each component is designed to complement the others, establishing a new standard for IDS in in-vehicle networks.

4.1. Novel feature engineering

Feature engineering plays a critical role in extracting valuable insights from CAN bus data to enhance IDS performance in autonomous vehicle systems. The main novelty of our approach lies in the dynamic calculation of data entropy and anomaly score, which are key detection features. These features are dynamically computed using a sliding window technique, making them well-suited for real-time applications.

While the sliding window is the primary method for calculating moving averages (MA), we also assess alternative techniques such as cumulative moving average (CMA) and exponential smoothing to determine their effectiveness in improving detection accuracy. The following parts will explain how each of these metrics is calculated and how they contribute to the overall anomaly detection system.

4.1.1. Time interval calculation

Time intervals can reveal abnormalities in the frequency of messages that are not clear when only considering timestamps. By including temporal analysis, the IDS model gains a dynamic element that improves its ability to detect deviations from typical communication patterns. The following equation is used in its calculation:

$$\Delta t_i = t_i - t_{i-1} \tag{1}$$

where Δt_i is the time interval between the current message i and the previous message $i-1$, and t represents the timestamp.

4.1.2. Sliding window

The sliding window technique is applied to both the entropy calculations and the MA of the data. The sliding window is used to maintain a set of recent traffic values over a fixed window size. For each new incoming traffic entry, data entropy and inter-arrival entropy are recalculated using the sliding window to capture real-time fluctuations in the data. Simultaneously, the sliding window is also employed to compute the MA for these entropy values, smoothing them out and making the system more resilient to sudden fluctuations.

4.1.3. Time interval entropy

The approach we propose involves dynamically calculating the entropy of inter-arrival times for each incoming communication, using a sliding window. The entropy is calculated using the following equation:

$$H(X) = -\sum p(x) \log_2 p(x) \tag{2}$$

where $p(x)$ denotes the probability of each distinct inter-arrival time occurring within the sliding window. Probabilities are calculated by dividing the number of occurrences of a specific inter-arrival time by the total number of previous intervals within the window. This ensures that only the most recent arrival times are considered when calculating entropy. The real-time IDS requires rapid adaptation to new data, which is achieved through the dynamic updating of entropy values as the sliding window moves forward.

4.1.4. Data entropy

Our research utilizes a dynamic approach to compute the entropy of data payloads using a sliding window. The calculation of entropy is specified by the following equation:

$$H(X) = -\sum p(x) \log_2 p(x) \tag{3}$$

where $p(x)$ is the probability of each single payload appearing within the sliding window. As each new data entry is processed, the entropy is recalculated based on the most recent payloads, ensuring continuous updating. Only the data within the sliding window is considered, allowing for real-time adaptation to changes in data patterns.

The key advantage of this method for real-time detection is its ability to continuously update the entropy as new data is received within the window. This makes it particularly well-suited for real-time IDS in dynamic environments where data properties are rapidly changing.

4.1.5. Calculating moving averages

Moving averages have been utilized in various anomaly detection systems due to their ability to smooth data fluctuations and capture trends over time. For instance, ARIMA models are known for capturing time series trends but are often too computationally intensive for real-time applications [25]. Similarly, in smart metering systems, moving averages are paired with neural networks for anomaly detection [26]. However, our approach uses a sliding window technique to dynamically update moving averages, providing fast adaptability for real-time detection in in-vehicle networks. This method maintains a set of recent traffic values and recalculates both data entropy and inter-arrival entropy as new data arrives within the sliding window.

The sliding window technique has been used effectively in real-time anomaly detection systems, such as those applied to hydrological data, where the dynamic nature of traffic makes rapid adaptation crucial [27]. However, unlike prior applications, our approach is specifically used for in-vehicle networks. It focuses on resource constraints and real-time attack detection in automotive environments.

The sliding window moving average calculates the average based on a fixed amount of the most recent incoming traffic within the window. The moving average (MA_t) is calculated at time t using a window size of N for a stream of incoming traffic x.

$$MA_t = \frac{1}{N} \sum_{i=t-N+1}^{t} x_i \tag{4}$$

As new data is received, the sliding window shifts forward by one observation, incorporating the latest traffic and removing the oldest entry from the calculation. This ensures that the moving average always reflects the most recent data. The continuous recalculation of the average makes this approach well-suited for real-time IDS in autonomous vehicles.

Example: In this example, the dynamic calculation of entropy and MA is performed using a sliding window of size 5, applied to incoming traffic data. As shown in Fig. 2, entropy is recalculated for each new traffic entry based on inter-arrival times. Simultaneously, the moving average is updated using the most recent 5 data points. This step-by-step process allows the system to monitor anomalies in real time. Both entropy and the moving averages adjust dynamically to reflect the latest traffic behavior. The figure illustrates how the sliding window moves with each new data point, ensuring the system responds quickly to emerging threats.

4.1.6. Alternative moving average techniques

In addition to our sliding window approach, we explored other techniques such as CMA and exponential smoothing. The exponential smoothing technique assigns more weight to recent traffic data while gradually reducing the significance of older data points. This characteristic makes it useful in dynamic environments where recent behavior is critical for detecting anomalies [28]. For instance, exponential smoothing has been successfully applied for anomaly detection in border gateway protocol (BGP) traffic [29]. Exponential smoothing is applied using the following formula:

$$S_t = \alpha x_t + (1-\alpha) S_{t-1} \tag{5}$$

The smoothing factor, denoted as α, is a value between 0 and 1. x_t represents the most recent traffic, while S_{t-1} is the smoothed value from the preceding time step.

On the other hand, CMA calculates the average of all past data points, making it useful for long-term trend analysis. However, in real-time IDS, CMA becomes less effective as the volume of incoming traffic increases. In contrast, the sliding window approach only considers a fixed number of recent data points, allowing it to adapt more quickly to the latest data. The CMA is calculated as follows:

$$CMA_T = \frac{1}{T} \sum_{i=1}^{T} x_i \tag{6}$$

where T represents the cumulative amount of traffic at the current time, while xi is the traffic value at time i. It is useful for identifying slow anomalies in IDS and provides an effective defense against any fluctuations in the data stream.

Both exponential smoothing and CMA are viable techniques. However, the sliding window method in our approach is the most efficient for real-time IDS in resource-constrained environments. In our method, the sliding window focuses on the most recent traffic patterns. This enables the system to adapt quickly and remain responsive to emerging threats in automotive networks.

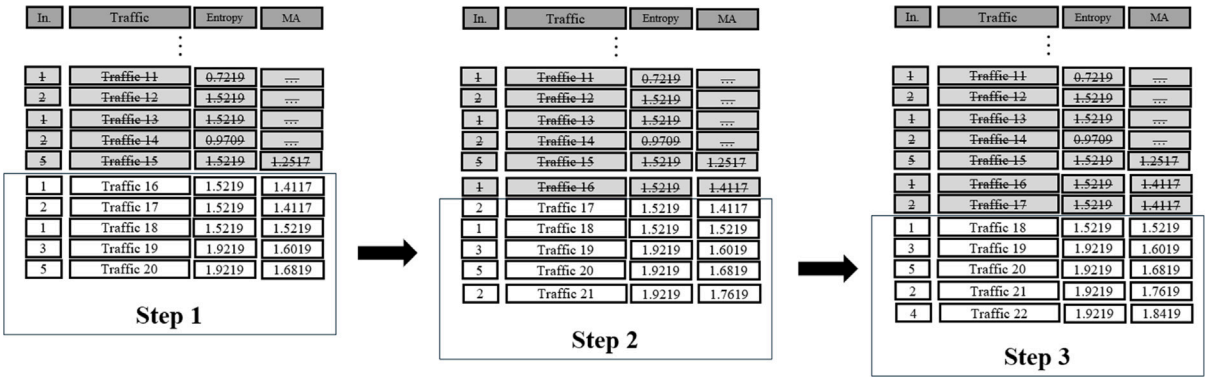


Fig. 2. Dynamic calculation of entropy and MA for incoming traffic using a sliding window of size 5. Each step shows how entropy is recalculated based on interarrival times, and the MA is updated for the most recent 5 data points.

4.1.7. Deviation from the moving average

In addition to our moving average calculations, we measured deviation, which is a crucial indicator of abnormalities in autonomous vehicle networks. Deviation is computed using the mathematical formula below:

$$D_t = |x_t - MA_t| \tag{7}$$

In this context, D_t refers to the deviation observed at a specific time t, x_t represents the value of the incoming traffic, and MA_t denotes the moving average at the same time t. The measure of the difference between the observed values and their predicted pattern enables us to identify deviations that could indicate possible intrusions or system malfunctions. Through continuous monitoring, our IDS is capable of rapidly identifying and reacting to abnormal patterns.

4.1.8. Anomaly score calculation

The anomaly score is the fundamental component of our detection mechanism, representing the heart of our real-time IDS. This score is determined by combining the calculated deviations from the moving average for both interval times and data payloads. The anomalous score is calculated using the following equation:

$$\text{AnomalyScore} = 0.5 \times D_{\text{interval}} + 0.5 \times D_{\text{data}} \tag{8}$$

D_{interval} refers to the deviation of the interval entropy from its moving average value, whereas D_{data} represents the deviation of the data entropy from its corresponding moving average. This scoring process measures the level of abnormality in the incoming traffic, where higher scores indicate a larger probability of an anomaly. The combination of these deviations into a single abnormality score could be a significant measure.

4.1.9. Output features

After feature engineering, our methodology produces two key output features: data entropy and anomaly score. Data entropy captures variations in message payloads, while the anomaly score identifies threats based on deviations in both data entropy and inter-arrival entropy. Additionally, SHAP is used to analyze the importance of these features, enhancing interpretability and securing the network connections in autonomous vehicles.

4.2. Lightweight model

The proposed lightweight neural network model is designed to complement the novel feature engineering techniques detailed in the previous subsections. This model is optimized for resource-constrained environments. As shown in Fig. 3, the architecture consists of three primary components: an input layer, two hidden layers, and an output layer.

The input layer receives the dynamically calculated features, including the novel detection features of data entropy and anomaly score, as well as standard CAN frame features (such as arbitration ID, timestamp, and data fields). These features are passed into the lightweight model for processing. The hidden layers employ ReLU (Rectified Linear Unit) activation functions, which introduce non-linearity while maintaining computational efficiency. The model contains 8 and 4 units in the first and second hidden layers, respectively, which balances model complexity with performance.

The output layer utilizes a softmax activation function to perform multi-class classification, distinguishing between different types of attacks in real time. The model is trained using the Adam optimizer with a learning rate of 0.001, which ensures efficient convergence during training.

By incorporating the novel features of data entropy and anomaly score into this lightweight model, we achieve enhanced detection accuracy without significantly increasing the computational overhead. The dynamic nature of these features allows the model to quickly adapt to incoming traffic patterns, identifying anomalies in real time. This streamlined architecture, combined with the innovative feature engineering process, makes the model highly effective for real-time intrusion detection in resource-constrained environments.

5. Experiments and results

In this section, we present a detailed evaluation of our lightweight IDS model, specifically designed for in-vehicle networks. We start by specifying the datasets and the evaluation metrics used. Next, we evaluate the findings by focusing on the sliding window's effectiveness as the primary method in our framework. We also compare it with CMA and exponential smoothing methods. Afterward, we conduct a comprehensive evaluation to compare our proposed models with state-of-the-art approaches. In the time performance subsection, we detail inference time measurements to show the practicality and competitive advantage of our framework. Finally, we include SHAP analysis to demonstrate the importance of the proposed features in enhancing detection accuracy.

5.1. Dataset

5.1.1. Dataset a

It was obtained from the car-hacking field, focuses on attack categories such as DoS, fuzzy, and spoofing attacks that specifically target drive gear and RPM indicators [12]. The dataset was created by recording CAN traffic from real vehicle during attack simulations conducted via an OBD-II connection. This dataset collected more than 12 million messages. The presence of a lot of injected malicious messages, along with a large volume of normal messages, allows for a comprehensive evaluation of IDS solutions in in-vehicle network [12].

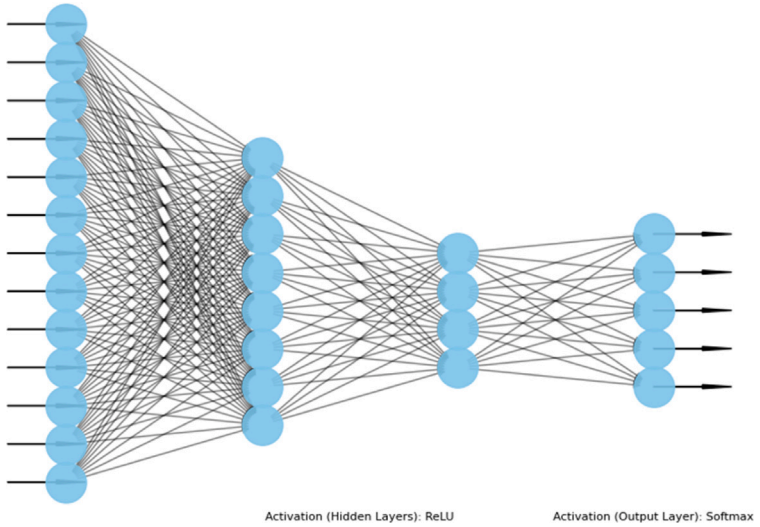


Fig. 3. Architecture of the lightweight neural network model.

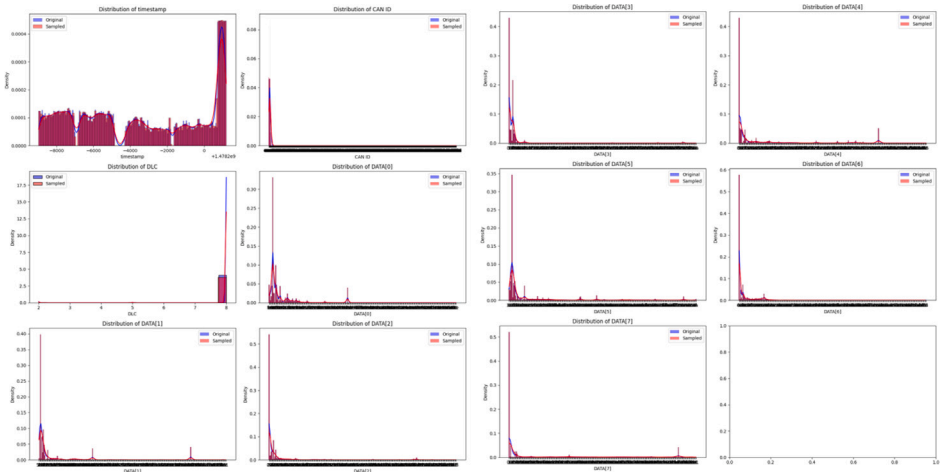


Fig. 4. The distributions of the features for both the original and sampled datasets.

Table 1
Datasets details including total rows, sample size, and label information.

Dataset	Total rows	Sample taken	Labeled
A	17,558,462	295,397	Yes
B	1,270,310	93,528	Yes

5.1.2. Dataset b

It was obtained from the "Car Hacking: Attack and Defense Challenge 2020" [30]. This dataset contains CAN traffic data from the Hyundai Avante CN7. It includes both normal operation messages and various attack vectors. The dataset is divided into two parts: the preliminary-round dataset and the final-round dataset. The final-round dataset includes normal messages and five attack types (four spoofing attacks and one fuzzing attack). The final-round dataset used in our experiments consisted of approximately 1.27 million messages and represents a real-world scenario [30].

Table 1 illustrates the total row count in each dataset, the quantity of samples acquired, and the labeling status of the used datasets. Fig. 4 illustrates the distribution of features for both the original and sampled datasets. This indicates that the samples used are representative of the original datasets. Furthermore, the sampled dataset was split into 80% for training and 20% for testing to ensure a thorough evaluation of the model's performance.

5.2. Experimental setup

The proposed IDS was developed using python (version 3.10.12) with the keras library (version 2.15.0) for training the model. The training was conducted on a system with an Intel Core i7-8750H CPU at 2.20 GHz, 32 GB of RAM, and a 4 GB NVIDIA GeForce GTX 1050 GPU. To test the model's performance in a resource-constrained environment, it was deployed on a Raspberry Pi 3 model B.

5.3. Evaluation metrics

Evaluating the performance of the IDS is crucial to ensure stability in real-life scenarios. The metrics used provide a comprehensive evaluation of the system's ability to classify data across many categories. Accuracy is a metric that quantifies the ratio of correctly identified instances compared with all predictions, serving as a quick measure of the effectiveness of the model. The overall recall score, also known as the detection rate, is the ratio of correctly identified positive examples to the total number of actual positive instances for each class. The F1 score is computed by taking the harmonic mean of precision and recall, providing a balanced and harmonious combination of the two metrics.

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{9}$$

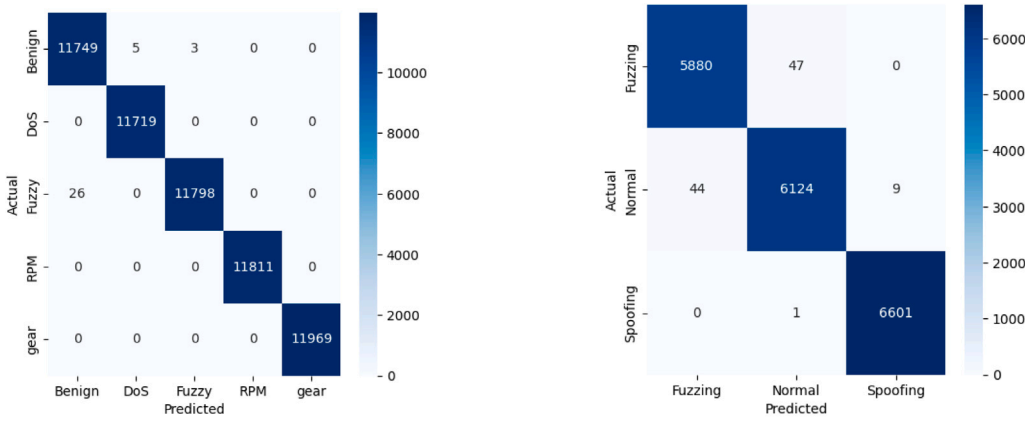


Fig. 5. Confusion matrices for: (a) Dataset A, (b) Dataset B.

Table 2
Example of the impact of window size on sliding window method performance on dataset B.

Window size	Metric	Result
5	Accuracy	0.9925
	F1-score	0.9922
	Recall	0.9923
	FPR	0.0037
10	Accuracy	0.9946
	F1-score	0.9945
	Recall	0.9944
	FPR	0.0027
20	Accuracy	0.9937
	F1-score	0.9936
	Recall	0.9936
	FPR	0.0031
30	Accuracy	0.9918
	F1-score	0.9915
	Recall	0.9916
	FPR	0.0041

$$Recall = \frac{TP}{(TP + FN)} \tag{10}$$

$$Precision = \frac{TP}{(TP + FP)} \tag{11}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{12}$$

where TP is true positives, TN is true negatives, FN is false negatives, and FP is false positives.

5.4. Evaluation results

We first conducted an experimental analysis to evaluate the impact of the sliding window size on the algorithm's performance, as it is the optimal technique in our approach. As shown in the Table 2, window sizes like 5 and 10 achieved significant detection accuracy and low FPR. Increasing the window size to 20 slightly decreased accuracy and recall, but further increasing to 30 resulted in a marginal rise in FPR. This highlights the need to select a suitable window size to balance detection accuracy with real-time performance in in-vehicle networks.

Fig. 5 present the confusion matrices for a sliding window size of 10, demonstrating the classification results for dataset A and dataset B. Both datasets had misclassified fuzzy samples. However, the approach shows overall efficiency in identifying major attack categories.

Table 3
The effectiveness of feature engineering techniques compared with the baseline strategy without feature engineering in both datasets.

Dataset	Techniques	Accuracy	F1-score	Detection Rate	FPR
A	Without FE	0.9703	0.9699	0.9700	0.0074
	Sliding window	0.9995	0.9995	0.9995	0.0001
	Exponential Smoothing	0.9992	0.9992	0.9992	0.0002
	CMA	0.9997	0.9997	0.9997	0.0001
B	Without FE	0.9436	0.9423	0.9428	0.0245
	Sliding window	0.9946	0.9945	0.9944	0.0027
	Exponential Smoothing	0.9888	0.9886	0.9884	0.0056
	CMA	0.9925	0.9924	0.9923	0.0037

Additionally, we evaluated the performance of two other techniques: CMA and exponential smoothing. The results in Table 3 show that all feature engineering strategies outperformed the baseline (without feature engineering) across both datasets A and B. This improvement provides evidence of the effectiveness of the designed features in capturing the complex nature of network traffic and facilitating the detection of cyber threats.

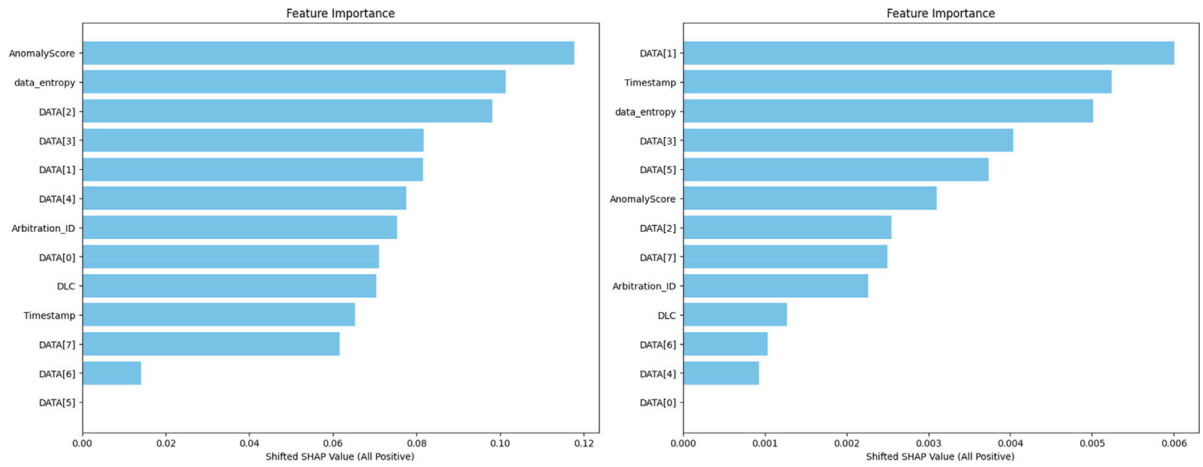
5.5. Comparison results

Our system demonstrates an effective balance between accuracy, F1-score, and detection rate. As presented in Table 4, our DMLP demonstrates an effective performance compared with the range of approaches mentioned. However, high-quality detection metrics may require significant computational resources, which is an important factor to consider for real-time IDS in-vehicle systems. The practical application of any IDS, especially in the computationally constrained environment, will be determined by the balance between speed and accuracy. Our model has a significantly low latency, as described in the following subsection.

5.6. Time performance

The training time for the proposed IDS model was recorded as 113.52 s on the system described earlier. To further evaluate its performance in a resource-constrained environment, the model was deployed on a Raspberry Pi 3. The characteristics of the Raspberry Pi can be found in Table 5.

In the analysis of the timing performance comparison presented in Table 6, our model demonstrated a significant latency of only 0.17 ms, beating other techniques by a wide margin. For example, MLIDS and LSTM, which have reported latencies of 275 and 147.054 ms [18,19], respectively, are significantly less efficient. The NovelADS technique



(a) (b)

Fig. 6. SHAP feature importance plot for: (a) Dataset A, (b) Dataset B.

Table 4
The comparative effectiveness of our IDS compared with other systems.

Ref	Technique	Dataset used	F1-score	Detection Rate	Classifier
[12]	GAN	A	0.9828	0.9865	Binary
[13]	DCNN	A	0.9991	0.9984	Binary
[18]	LSTM	A	0.9995	0.9998	Multi-class
[17]	MLP	A	0.9600	0.9826	Multi-class
[16]	CNN-LSTM	A	0.9993	0.9991	Multi-class
[14]	DCNN	A	0.9959	0.9939	Binary
[15]	TCN	A	0.9996	0.9995	Multi-class
[20]	Random Forest	B	0.9799	0.9831	Multi-class
[20]	XGBoost	B	0.9790	0.9835	Multi-class
Ours	DMLP	A	0.9995	0.9995	Multi-class
Ours	DMLP	B	0.9945	0.9944	Multi-class

Table 5
The characteristics of the Raspberry Pi used to evaluate inference time.

Specification	Detail
Model	Raspberry Pi 3 Model B
Processor	1.2 GHz Quad-Core ARM Cortex-A53
RAM	1 GB LPDDR2
Networking	10/100 Ethernet, 2.4 GHz 802.11n wireless
Storage	MicroSD card slot
Power Input	5 V/2.5 A DC via micro USB

Table 6
A comparison of the delay, measured in milliseconds, of several intrusion detection methods, highlighting the effectiveness of our DMLP approach.

Ref	Method	Latency (ms)
[18]	MLIDS	275
[19]	LSTM	147.054
[16]	NovelADS	128.7
[17]	MLP	38.03
[20]	Random Forest	33.744
[20]	XGBoost	33.599
[12]	GIDS	5.89
[13]	DCNN	5
[15]	TCN	3.4
[14]	DCNN	0.43
Ours	DMLP	0.17

shows a latency of 128.7 ms, illustrating the significant benefit of our concept in terms of operating speed.

In addition, Seo et al. [12] and Song et al. [13] used GTX 1080 and Tesla K80, respectively, to measure the time it takes for inference. These techniques are generally unsuitable to evaluate time performance because they do not accurately represent the technology utilized in autonomous vehicles. It is crucial to consider that certain methods, including those mentioned by Seo et al. [12], Song et al. [13], and Cheng et al. [15], employ block-based detection. Although this approach can decrease the average time it takes to process each message, it requires collecting messages before they can be processed, resulting in a delay. For example, in the study of Cheng et al. [15], it took approximately 8.3 ms to accumulate 64 messages on a 1-Mbps CAN network with 8-byte messages. On the other hand, our solution delivers low latency without the requirement for such accumulation, allowing for real-time detection and quick response, which are crucial for ensuring the security of in-vehicle networks.

Furthermore, the most competitive technique was that used by Khandelwal and Shreejith [14], which employs DCNN and achieves an inference time of 0.43 ms. Although the time performance of DCNN is impressive, the authors' evaluation was focused on binary classification and restricted to only two types of attacks. On the other hand, our approach provides an efficient multiclass detection ability.

5.7. Feature importance using SHAP

SHapley Additive exPlanations (SHAP) is a popular method in the field of explainable artificial intelligence (XAI) introduced by Lundberg and Lee [31]. It aims to make the results of machine learning models more understandable. SHAP gives each feature an importance value for individual predictions. Unlike other XAI methods, SHAP is based on strong theoretical foundations and can explain the outputs of any machine learning model [32].

Our novel features provide dynamic and effective detection across various attack scenarios. Anomaly score and data entropy play key roles in identifying attacks in both dataset A and dataset B. As shown in Fig. 6(a), anomaly score stands out as the most important feature, effectively capturing deviations from normal behavior.

Data entropy, the second most important feature, is effective in identifying randomness within CAN traffic, which makes it useful against attacks like fuzzing. As shown in Fig. 6(b), data entropy proves valuable in detecting random noise generated by fuzzing attacks. It adapts dynamically to changes in data patterns caused by injected errors. At the same time, anomaly score plays a critical role in detecting attacks like spoofing by identifying irregularities in message flows that other methods might miss.

The difference in the importance of timestamp between dataset A and dataset B can be explained by the nature of the attacks and how they are injected into the CAN traffic. The higher importance of timestamp in dataset B is due to attack methods that directly affect message

timing. In contrast, dataset A focuses more on data manipulation, where features like data entropy and specific CAN data bytes take precedence over timing-related features.

6. Conclusion and future work

As AVs continue to offer effortless driving functions for drivers and passengers, ensuring automotive security to protect driver safety is becoming increasingly crucial. Nevertheless, a CAN bus, the communication backbone for an in-vehicle network, is susceptible to vulnerabilities that include a lack of authentication for access control. This security vulnerability enables an attacker to inject attack packets into a system and thereby gain control over a vehicle's safety-critical operations, including acceleration, braking, and more. To resolve this issue, two new detection features, data entropy and anomaly score, were designed via this current study. We applied them to a lightweight DL model. The performance evaluation was conducted on two commonly used public datasets. The IDS we propose demonstrated significant detection capability in the context of multi-class classification. Additionally, we evaluated the time it took for the inference process to be completed in computing environments with limited resources. The latency of our model is impressively low, with an inference time of only 0.17 ms. This outperforms other machines and DL IDSs currently available. The feature engineering processes outlined in this paper can serve as the foundation for building an IDS that can be utilized for real-time detection. Our goal for the future is to implement this model in a real-life system in which adjustments for these calculations can be executed on the fly when new traffic is received.

CRediT authorship contribution statement

Wael Aljabri: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Data curation, Conceptualization. **Md. Abdul Hamid:** Writing – review & editing, Supervision. **Rayan Mosli:** Writing – review & editing, Supervision.

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

We used public datasets which are cited in the paper.

References

[1] A.M. Nasser, Automotive Cybersecurity Engineering Handbook, Packt Publishing Ltd, 2023.

[2] L. Pan, X. Zheng, H. Chen, T. Luan, H. Bootwala, L. Batten, Cyber security attacks to modern vehicular systems, J. Inf. Secur. Appl. 36 (2017) 90–100.

[3] E. Aliwa, O. Rana, C. Perera, P. Burnap, Cyberattacks and countermeasures for in-vehicle networks, ACM Comput. Surv. 54 (1) (2021) 1–37.

[4] M. Bozdal, M. Samie, S. Aslam, I. Jennions, Evaluation of can bus security challenges, Sensors 20 (8) (2020) 2364.

[5] C. Miller, C. Valasek, Remote exploitation of an unaltered passenger vehicle, Black Hat USA 2015 (S 91) (2015) 1–91.

[6] Toyota, Toyota GB statement on vehicle theft, 2024, https://mag.toyota.co.uk/toyota-gb-statement-on-vehicle-theft/, *Accessed: 2024-05-09.

[7] W. Tong, A. Hussain, W.X. Bo, S. Maharjan, Artificial intelligence for vehicle-to-everything: A survey, IEEE Access 7 (2019) 10823–10843.

[8] C. Specification, Bosch, Robert Bosch GmbH, Postfach 50 (1991) 15.

[9] S. Srinivas, R.V. Babu, Deep learning in neural networks: An overview, Comput. Sci. (2015).

[10] G. Villarrubia, J.F. De Paz, P. Chamoso, F. De la Prieta, Artificial neural networks used in optimization problems, Neurocomputing 272 (2018) 10–16.

[11] I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, Adv. Neural Inf. Process. Syst. 27 (2014).

[12] E. Seo, H.M. Song, H.K. Kim, GIDS: GAN based intrusion detection system for in-vehicle network, in: 2018 16th Annual Conference on Privacy, Security and Trust, PST, IEEE, 2018, pp. 1–6.

[13] H.M. Song, J. Woo, H.K. Kim, In-vehicle network intrusion detection using deep convolutional neural network, Veh. Commun. 21 (2020) 100198.

[14] S. Khandelwal, S. Shreejith, A lightweight multi-attack CAN intrusion detection system on hybrid FPGAs, in: 2022 32nd International Conference on Field-Programmable Logic and Applications, FPL, IEEE, 2022, pp. 425–429.

[15] P. Cheng, K. Xu, S. Li, M. Han, TCAN-IDS: intrusion detection system for internet of vehicle using temporal convolutional attention network, Symmetry 14 (2) (2022) 310.

[16] K. Agrawal, T. Alladi, A. Agrawal, V. Chamola, A. Benslimane, NovelADS: A novel anomaly detection system for intra-vehicular networks, IEEE Trans. Intell. Transp. Syst. 23 (11) (2022) 22596–22606.

[17] F. Amato, L. Coppolino, F. Mercaldo, F. Moscato, R. Nardone, A. Santone, Can-bus attack detection with deep learning, IEEE Trans. Intell. Transp. Syst. 22 (8) (2021) 5081–5090.

[18] A.K. Desta, S. Ohira, I. Arai, K. Fujikawa, MLIDS: Handling raw high-dimensional CAN bus data using long short-term memory networks for intrusion detection in in-vehicle networks, in: 2020 30th International Telecommunication Networks and Applications Conference, ITNAC, IEEE, 2020, pp. 1–7.

[19] M.D. Hossain, H. Inoue, H. Ochiai, D. Fall, Y. Kadobayashi, LSTM-based intrusion detection system for in-vehicle can bus communications, Ieee Access 8 (2020) 185489–185502.

[20] Y. Jeong, H. Kim, S. Lee, W. Choi, D.H. Lee, H.J. Jo, In-vehicle network intrusion detection system using CAN frame-aware features, IEEE Trans. Intell. Transp. Syst. (2023).

[21] Y. Yang, G. Xie, J. Wang, J. Zhou, Z. Xia, R. Li, Intrusion detection for in-vehicle network by using single GAN in connected vehicles, J. Circuits Syst. Comput. 30 (01) (2021) 2150007.

[22] Q. Zhao, M. Chen, Z. Gu, S. Luan, H. Zeng, S. Chakrabory, CAN bus intrusion detection based on auxiliary classifier GAN and out-of-distribution detection, ACM Trans. Embedded Comput. Syst. (TECS) 21 (4) (2022) 1–30.

[23] A.K. Desta, S. Ohira, I. Arai, K. Fujikawa, Rec-CNN: In-vehicle networks intrusion detection using convolutional neural networks trained on recurrence plots, Veh. Commun. 35 (2022) 100470.

[24] T.-N. Hoang, D. Kim, Supervised contrastive ResNet and transfer learning for the in-vehicle intrusion detection system, Expert Syst. Appl. 238 (2024) 122181.

[25] V. Kozitsin, I. Katser, D. Lakontsev, Online forecasting and anomaly detection based on the ARIMA model, Appl. Sci. 11 (7) (2021) 3194.

[26] S. Maitra, S. Kundu, A. Shankar, A real-time anomaly detection using convolutional autoencoder with dynamic threshold, 2024, arXiv preprint arXiv:2404.04311.

[27] L. Kulanuwat, C. Chantrapornchai, M. Maleewong, P. Wongchaisuwat, S. Wimala, K. Sarinnapakorn, S. Boonya-Aroonnet, Anomaly detection using a sliding window technique and data imputation with machine learning for hydrological time series, Water 13 (13) (2021) 1862.

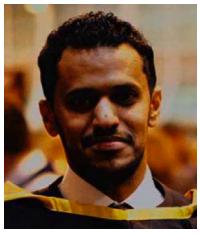
[28] J. Yu, S.B. Kim, J. Bai, S.W. Han, Comparative study on exponentially weighted moving average approaches for the self-starting forecasting, Appl. Sci. 10 (20) (2020) 7351.

[29] P. Moriano, R. Hill, L.J. Camp, Using bursty announcements for detecting BGP routing anomalies, Comput. Netw. 188 (2021) 107835.

[30] H. Kim, Car hacking: Attack & defense challenge 2020 dataset. IEEE, 03 feb 2021, 2021.

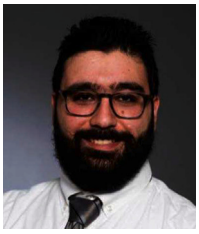
[31] S.M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, Adv. Neural Inf. Process. Syst. 30 (2017).

[32] M. Sarhan, S. Layeghy, M. Portmann, Evaluating standard feature sets towards increased generalisability and explainability of ML-based network intrusion detection, Big Data Res. 30 (2022) 100359.



Wael Aljabri is now a PhD candidate at the Department of Information Technology at King Abdulaziz University. He is a senior trainer and co-founder of PT Cybersecurity, a company focused on protecting small and medium-sized enterprises from digital attacks. He has taught computing, programming, and network security at a top Technical College and currently works as a cyber defense Engineer. His academic path concluded with a Master of Engineering in Information and Network Security from the University of Limerick, providing him with a strong basis to address the difficulties in the industry. At PT Cybersecurity, he has applied theoretical knowledge to develop real solutions by directly tackling weaknesses. His dedication to cybersecurity is motivated by the recognition of its crucial significance and the aspiration to provide new ideas in order to remain proactive against advancing dangers.

Md. Abdul Hamid was born in the Village Sonatola, Pabna, Bangladesh. His education life spans over different countries in the world. He completed his high school and college (1989- 1995) graduation from Rajshahi Cadet College, Bangladesh. He earned his Bachelor of Engineering degree in Computer & Information Engineering (1996-2001) from International Islamic University Malaysia (IIUM). He earned the Master-Ph.D. (2004-2009) degree from Kyung Hee University, South Korea, in August 2009 from the Computer Engineering department majoring in Information Communication. He has been in the teaching profession throughout his life, which also spans over different parts of the globe. 2002-2004: Lecturer in the Computer Science & Engineering department, Asian University of Bangladesh, Dhaka. 2009-2012: Assistant professor in the department of Information & Communications Engineering at Hankuk University of Foreign Studies (HUFS), South Korea. 2012-2013: Assistant professor in the Department of Computer Science and Engineering, Green University of Bangladesh. 2013-2016: Assistant professor in the Department of Computer Engineering at Taibah University, Madinah, Saudi Arabia. 2016-2017: Associate Professor of Department of Computer Science, Faculty of Science & Information Technology at American International University-Bangladesh, Dhaka, Bangladesh. 2017-2019: Associate Professor & professor in the Department of Computer Science and Engineering at University of Asia Pacific, Dhaka, Bangladesh. 2019-present: Professor in the Department of Information Technology, King Abdulaziz University, Jeddah, KSA. His research interest includes network/cyber-security, natural language processing, machine learning, wireless communications, and networking protocols etc. He has served as a program committee member contributing for the curriculum development and new program development in undergraduate and graduate disciplines.



Rayan Mosli obtained his PhD in 2020 from the Rochester Institute of Technology in the State of New York. His research interests include malware detection using machine learning, adversarial machine learning, natural language processing, IoT, and blockchain. Dr. Mosli is currently an assistant professor in the Faculty of Computing and Information Technology at King Abdulaziz University where he also served in multiple administrative roles such as the deputy director of the High Performance Computing Center, the deputy director of the Cybersecurity Center, and as a member of multiple other committees within KAU.