*smart cities*

MDPI

*Article*

# A Novel Light-Weight Machine Learning Classifier for Intrusion Detection in Controller Area Network in Smart Cars

**Anila Kousar [1], Saeed Ahmed [1], Abdullah Altamimi [2,3,*] and Zafar A. Khan [1,4,*]**

[1] Department of Electrical Engineering, Mirpur University of Science and Technology (MUST), Mipur AJK-10250, Pakistan
[2] Department of Electrical Engineering, College of Engineering, Majmaah University, Al-Majmaah 11952, Saudi Arabia
[3] Engineering and Applied Science Research Center, Majmaah University, Al-Majmaah 11952, Saudi Arabia
[4] Department of Computer Systems Engineering, Mirpur University of Science and Technology (MUST), Mipur AJK-10250, Pakistan
* Correspondence: a.altmimi@mu.edu.sa (A.A.); zafarakhan@ieee.org (Z.A.K.)

**Highlights:**

**What are the main findings?**

- Development of a novel lightweight machine learning classifier for anomaly detection in smart cars.
- Comparison with the baseline methods and existing literature revealed remarkable reduction in computational cost with significant increase in detection rate accuracy.

**What are the implications of the main finding?**

- A decent choice for the imbalanced data and critical applications like spam identification, fraud detection, financial analytics and many others.
- Development of a firmware compatible with the central gateway and the target electronic control unit in smart cars is a key adaption challenge.

**Abstract:** The automotive industry has evolved enormously in recent years, marked by the proliferation of smart vehicles furnished with avant-garde technologies. These intelligent automobiles leverage cutting-edge innovations to deliver enhanced connectivity, automation, and convenience to drivers and passengers. Despite the myriad benefits of smart vehicles, their integration of digital systems has raised concerns regarding cybersecurity vulnerabilities. The primary components of smart cars within smart vehicles encompass in-vehicle communication and intricate computation, in addition to conventional control circuitry. In-vehicle communication is facilitated through a controller area network (CAN), whereby electronic control units communicate via message transmission across the CAN-bus, omitting explicit destination specifications. This broadcasting and non-delineating nature of CAN makes it susceptible to cyber attacks and intrusions, posing high-security risks to the passengers, ultimately prompting the requirement of an intrusion detection system (IDS) accepted for a wide range of cyber-attacks in CAN. To this end, this paper proposed a novel machine learning (ML)-based scheme employing a Pythagorean distance-based algorithm for IDS. This paper employs six real-time collected CAN datasets while studying several cyber attacks to simulate the IDS. The resilience of the proposed scheme is evaluated while comparing the results with the existing ML-based IDS schemes. The simulation results showed that the proposed scheme outperformed the existing studies and achieved 99.92% accuracy and 0.999 F1-score. The precision of the proposed scheme is 99.9%, while the area under the curve (AUC) is 0.9997. Additionally, the computational complexity of the proposed scheme is very low compared to the existing schemes, making it more suitable for the fast decision-making required for smart vehicles.

**Keywords:** controller area network; cyber-security; cyber attacks; intrusion detection system; smart vehicles; machine learning; pythagorean distance



**Citation:** Kousar, A.; Ahmed, S.; Altamimi, A.; Khan, Z.A. A Novel Light-Weight Machine Learning Classifier for Intrusion Detection in Controller Area Network in Smart Cars. *Smart Cities* **2024**, *7*, 3289–3314. https://doi.org/10.3390/smartcities7060127

Academic Editor: Pierluigi Siano

Received: 29 June 2024
Revised: 26 October 2024
Accepted: 31 October 2024
Published: 2 November 2024

## 1. Introduction

The evolution of automotive industry has brought many changes which improved the security and safety of the commuters. However, reports of World Health Organization revealed that there are almost 1.3 million causalities every year by road accidents, i.e., one death in every 24 s [1]. Some real-life examples of the car hacking include "Jeep Cherokee Hack" in 2015 [2], "Tesla Model S Hack" in 2016 [3], and "Nissan Leaf Vulnerability" in 2016 [4]. In the Jeep Cherokee case, the car was remotely hacked by two security researchers, Miller and Valasek, exploiting its infotainment and advanced driver assistance (ADAS) systems. This hijacking attempt was deliberate by the company to test the "Zero-day exploit", a hacking technique. "Tesla Model S Hack" happened when some researchers from the Keen Security Lab remotely hijacked the car's CAN-bus, taking control of the ADAS system both in driving and parking mode. They could control dashboard screen, door locks, car brakes, windscreen wipers, move seats, open boot and sunroof. While being able to remotely access the Nissan Leaf, experts in hacking gained remote access to the car through mobile application using vehicle identification number. This allowed them to operate the car air conditioning system, obtain driving history, and perform climate control functions successfully. During the year 2022, a large number of cars from 16 different makers were hacked by the group of seven researchers who were able to gain access to various car functions, such as honk vehicles, flash headlights, lock/unlock cars, start/stop engine, precisely locate cars, and even alter car ownership remotely. The impacted companies were Hyundai, Honda, Land Rover, Kia, Ford, Ferrari, Nissan, Mercedes-Bens, BMW, Acura, Toyota, Rolls Royce, Porsche, Jaguar, Infiniti, and Genesis [5]. This all shows a serious efficiency and safety concern for the auto-industry which demands a more secure, robust, and efficient self-defense mechanism for cars with less human intervention. Many renowned companies as Google in USA performed road tests of connected intelligent cars in 2009 [6]. The University of Michigan [7] examined the performance of similar cars designed by Tesla [8] in the field of Mcity. In the subsequent years, Mercedes Benz, Audi, and BMW started working on controller area network (CAN), a key element for establishing communication in smart cars (SCs). These intelligent smart cars can perform inter- and intra-vehicular communication and are able to make quick intelligent decisions for collision avoidance.

Keeping the legacy system, SCs are integrated with advanced technology, connectivity, and digital interface which add intelligent smart features such as keyless entry, automatic parking, traffic sign recognition, lane keeping, emergency braking, obstacle detection, and many others to name. The advanced featuring is by the virtue of embedded electronics which includes the utilization of numerous sensors and electronic control units (ECUs) that build an in-vehicular network in SCs to establish effective communication for efficient and safe drive. The in-vehicular communication is achieved through controller area network which is considered an efficient communication protocol compared to ethernet and FlexRay networks used in SCs [9]. Being message-oriented protocol, CAN allows robust communication between sensors and ECUs, but it lacks proper data authentication and an encryption mechanism. This makes CAN-bus highly vulnerable to cyber attacks arising serious questions on the reliability of a smart car. An unauthorized user can gain access to the system, launch an attack, manipulate the data and thus infiltrate the safe operation of a smart car to gain personal, financial, or any other benefit. This situation demands development of an efficient, reliable and robust intrusion identification and detection system for SCs. The communication between an IDS and the users normally involves (i) Attack Identification and Detection, where the IDS detects an illegal attempt made by the threat actor to establish control of the CAN-bus to send malicious data; (ii) Alert Generation, where alerts are generated by the IDS and are sent to the car dashboard and or to the driver's mobile app; (iii) Automated Response, which is performed by the IDS to disable the OBD-II port restricting external access. In the worst case, (iv) an alert is also sent to the manufacturer for detailed check and debugging. In this study, the primary focus is on Step (i) of the IDS process with the assumption that the proper communication is taking place between the IDS and the users.

### *1.1. Related Works*

Machine learning (ML) presents promising solutions for intrusion and anomaly detection in cyber-physical systems [10–13] . ML techniques primarily learn from data and make predictions based on the data pattern and training. Numerous ML-based studies have been conducted to design an intrusion detection system (IDS) and analyze the behavior of a smart car under different cyber attacks [14–21].

Utilizing SVM-, kNN-, and DT-based ML approaches and real-time data from Chevrolet Spark and Kia Soul, Bari et al. [14] investigated the performance of an ML-based IDS for smart cars. The study conducted experiments using DoS, impersonation and fuzzy attacks. Results showed a remarkable high accuracy of 99.9% with a 1.0 F1-score; however, the computational time including model training and testing time for all the attack types and classifiers is significantly high for all the cases. While evaluating the performance of a machine learning-based IDS proposed by researchers in [15] utilizing RF, DT, MLP, and SVM classifiers, the authors successfully identified and detected the DoS, fuzzy, and impersonation attacks injected on the real-time data from the CAN-bus of KIA Soul. Though the study achieved the accuracy of 98.5269%, the model is very heavyweight with training and testing time of 460.719 s and 14.935 s, respectively, bringing into question its suitability for events requiring instant fast decision. In a related study [16], a generative adversarial network-based anomaly detection scheme is proposed for smart cars. The study presented fairly good results based on different parameters of confusion matrix as hit rate, miss rate, false alarm, and correct rejection rates. Giving no information about anomaly detection time and limited to single attack type, the study fails to provide its effectiveness for multiple attack designs and the smart car used for the data collection is undefined.

Realizing the consequeses of cyber attacks and leveraging the ML techniques, Shahriar et al. [17] proposed a single level deep learning-based anomaly detection system, CANShield for the SCs. The CANShield model comprises the data preprocessing, data analyzer, and ensemble method-based AD modules. The preprocessing module processes and manages the complex data, the analyzer module investigates the time-series data, whereas the detection module offers the final outcome. Despite the comprehendible performance results, the study fails to provide the details about the car type used for data collection, the accuracy measure, and the computational time to make the classification decision. Featured with an LSTM-based ML technique, an IDS for the intrusion detection for a CAN-bus network in smart cars is proposed in [18]. The study investigated the effect of DoS, fuzzy, and spoofing attacks on CAN messages transmitted by different ECUs for normal SCs functionality. The simulation results presented considerably good sensitivity and specificity scores; however, the study lacks details of model accuracy and computational time, which are crucial parameters of an IDS designed for SCs.

Employing feature engineering for the selection of highly significant variables to train a model, the researchers devised a deep learning-based solution for the detection of flaws and anomalies in SCs. They utilized an IVS-Hackathon dataset, which included four different types of cyber attacks. Achieving 95% accuracy, the study provided good comparison with baseline methods and recent studies; however, it neither considered the computational complexity nor the model verified for different datasets. Further, the accuracy value could be increased [19]. Kim et al. [20] determined the performance of an anomaly detection model developed utilizing an LSTM-based autoencoder approach. While presenting the comprehensive simulation results, the study is limited to a single attack type, i.e., a fuzzy attack considering only one dataset. In another similar study, Wang et al. [21] proposed a GAN-based IDS for anomaly identification in a CAN-FD-bus in SCs. The model employed DoS, fuzzy, PRM/Gear Spoofing attack, and normal data for model training. The results showed an outstanding detection rate of 99.93%; however, the study skipped the other performance evaluation metrics (F1-score, precision, recall, ROC) and also the computational time.

Kishoare et al. [22] proposed an intelligent IDS for SCs to identify intrusions, anomalies, and flaws in CAN-bus data utilizing the Bidirectional Long Short-Term Memory technique

employed on the "Car Hacking: Attack and Defense Challenge, 2020" dataset. They found that the proposed scheme achieved a 99.09% detection accuracy with 0.9910 and 0.9901 F1-score and precision, respectively. Aldhyani and Alkahtani [23] evaluated the performance of a deep learning-based security system developed to protecting CAN-bus data from various cyber intrusions and assaults. The study presented a 97.43% accuracy to identify and detect CAN-bus anomalies by the proposed system. He et al. [24] studied the behavior of the anomaly detection model in connected and autonomous cars (CAVs) using decision tree and naive-bayesian machine learning schemes employed on a KDD99-based CAV-KDD dataset. The study concluded that a decision tree is superior to naive bayes in anomaly detection requiring less simulation time. To locate false messages injected into a CAN-bus, Pawar et al. [25] used k-nearest neighbour (kNN)- and decision tree (DT)-based ML models and found 81.48% and 77.99% detection accuracy values for kNN and DT, respectively. Gupta et al. [26] presented a novel graph-based ML scheme to secure smart cars from anomalies. Employing the machine learning and event-triggered interval method, Han et al. [27] detected and identified the abnormalities in CAN-bus data in SCs, and the model achieved up to 99% detection accuracy with 0.990 F1-score and 0.991 precision. While introducing different cyber attacks on a SC in a laboratory setup and collecting real-time data, Onur et al. [28] investigated the performance of an IDS by employing various machine learning techniques. The simulation results showed that the random forest surpassed the other schemes with 0.923 F1-score, 0.925 precision, and 96.1% classification accuracy. Similarly, Alalwany et al. [29] analyzed the performance of an IDS to classify different cyber assaults on CAN-bus data. Their model incorporated machine learning and ensemeble methods including random forest, decision tree, eXtreme gradient boosting, stacking, bagging, and voting with Kappa architecture. The study concluded that outperforming the other methods, the stacking ensemble classifier achieved the best results, i.e., a 98.5% accuracy with 0.985 F1-score and 0.987 precision. In similar studies by Alsaade and Al-Adhaileh [1] and Anand et al. [30], a deep learning method was applied to identify erroneous information and malicious network traffic in smart cars. Reviewing the various studies carried out for anomaly detection in CAN-bus, the authors analyzed different IDSs on the basis of the detection technique, the attack type, and evaluation metrics, and summarized the findings in [31]. Table 1 presents the comparison of the proposed study with some relevant studies in terms of investigating the impact of different cyber attacks on CAN-bus data in smart cars.

**Table 1.** A comparative overview of various machine learning-based studies for anomaly detection in CAN-bus in smart cars and identification of the research gap.

| References | Attack Type | | | | | | No. of Datasets | Detection Method | Research Gap |
|---|---|---|---|---|---|---|---|---|---|
| | DoS | Fuzzy | Flooding | Spoofing | Impersonation | Malfunction | | | |
| [32] | √ | √ | × | √ | × | × | 1 | FFS-ML | × |
| [33] | × | √ | × | √ | × | × | 1 | NN | Heavyweight |
| [34] | √ | √ | × | √ | × | × | 1 | NN, MLP | × |
| [35] | √ | √ | × | √ | × | × | 1 | ML | × |
| [36] | √ | √ | × | × | √ | × | 1 | BN | × |
| [37] | √ | √ | × | × | √ | × | 1 | ML | Heavyweight |
| [38] | √ | √ | × | √ | × | × | 1 | TCAN-ML | Heavyweight |
| [39] | √ | √ | × | √ | × | × | 1 | Deep-CNN | Heavyweight |
| [15] | √ | √ | × | × | √ | × | 1 | DT, SVM, RF, MLP | Heavyweight |
| [40] | × | √ | × | × | × | × | 1 | NN | × |
| [14] | √ | √ | √ | × | √ | √ | 2 | SVM, KNN, DT | Heavyweight |
| [41] | √ | √ | × | √ | × | × | 1 | DNN | × |
| [42] | √ | √ | × | × | √ | × | 1 | EM | Heavyweight |
| [43] | √ | √ | × | × | √ | × | 1 | NB, RF, RT, SGD | Heavyweight |
| This study | √ | √ | √ | √ | √ | √ | 6 | PD-ML | Lightweight |

Acronyms are defined in "Abbreviations" section.

As evident from the literature above, significant contributions have been made by the researchers to propose an anomaly detection model for smart cars. However, limited to the utilization of a single dataset to train an ML-IDS for the detection of a few cyber attacks

injected to CAN-bus in SCs, these IDSs are comparatively heavyweight with computational costs as high as 86.67% [44] compared to this work. Further, the applicability of an IDS should be extended to different makes of the smart car while tackling different types of cyber intrusions. To our knowledge, the literature lacks a simple, fast, lightweight, and efficient intrusion detection system capable of instant decision-making power with zero delay on real-time data while being concurrently applicable to different car models. Additionally, the literature would gain strength by the addition of a combined unified model proposed in this study featuring detection accuracy as high as 99.922% with a 99.9% precision along with detection time reduced to 0.00027 s while becoming equally efficient for various types of smart cars such as KIA Soul, Hyundai Avante CN7, Hyundai YF Sonata, Genesis g80, and CHEVROLET Spark. In pursuit of this, the paper proposes a machine learning-based novel approach to develop a simple, lightweight but efficient intrusion detection classifier for smart cars.

### *1.2. Contributions*

In the real-world scenario, the timely identification of an intrusion is crucial, as delays can lead to severe, threatening consequences. A simple but robust and efficient PD-ML method to detect intrusions, flaws, and anomalies injected by the intruders through the OBD-II port in CAN-bus is proposed in this study. Additionally, the model is evaluated using six different realistic datasets which include 13 different types of attack injections. The main technical contributions to the knowledge are listed below:

- Development of a lightweight novel Pythagorean distance-based machine learning (PD-ML) classifier for intrusion detection in CAN-bus of smart cars;
- Employment of 13 unique cyber attacks categorized into six different classes obtained from six different widely accepted CAN-bus datasets to evaluate the performance of the proposed scheme;
- A comparison of the outcomes by the proposed scheme with distance metrics, existing baseline methods and recent studies. The results show that the proposed scheme outperforms the existing schemes, increasing 2.48%, 2.16%, 2.13%, and 1.75% in accuracy and remarkably reducing by 99.77%, 99.58%, 99.47%, 96.61%, and 86.67% in computational cost.

The remaining paper is organized as follows. Section 2 offers an overview of smart cars with a brief description of CAN-bus protocol, cyber vulnerabilities, cyber attacks on an in-vehicular network and their impacts on car safety and passenger security. The proposed PD-ML scheme for anomaly detection is presented in Section 3, whereas Section 4 discusses the simulation results. Finally, the paper is concluded in Section 5.

## 2. Overview of Smart Cars

Smart cars are the intelligent cars capable of automated multi-tasking performed through an efficient and well-established communication mechanism. It could be vehicle to everything, vehicle to vehicle, and in-vehicle communication. It involves the utilization of various components such as sensors and computing and communication elements which allow data collection, sharing, and processing among different ECUs for efficient and safe driving. Presenting the overview of a SC, Figure 1 highlights various smart features, CAN-bus connectivity to ECUs, and possible cyber attack injections in SCs where sensor data are transmitted through communication modules over the network controlled by the ECUs. The infotainment feature allows navigation and various entertainment activities like music, video playback, and connectivity with mobile phones. Similarly, the human–machine interface enables communication between the car and its occupants through different systems such as voice recognition and gesture controls.

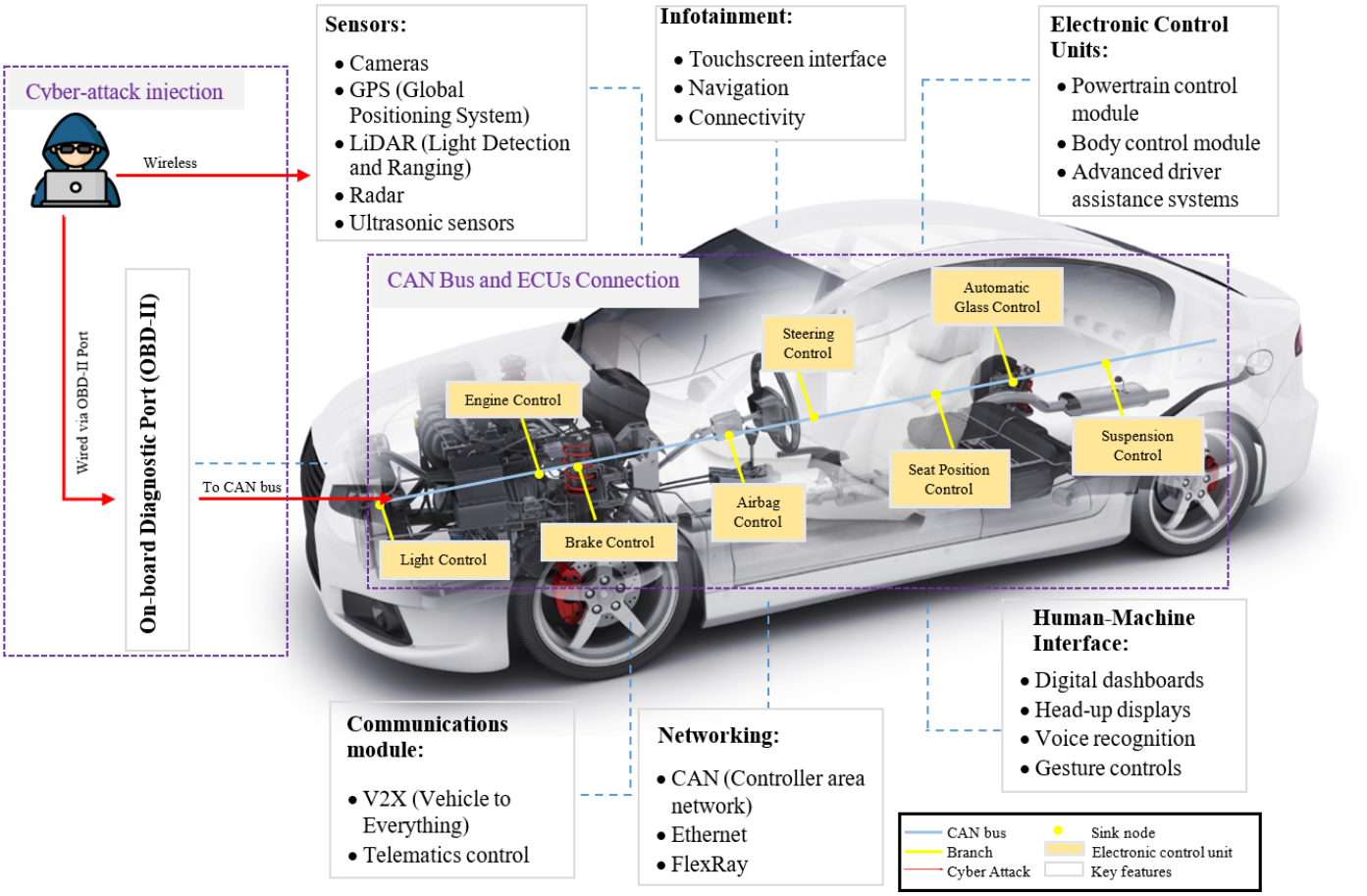


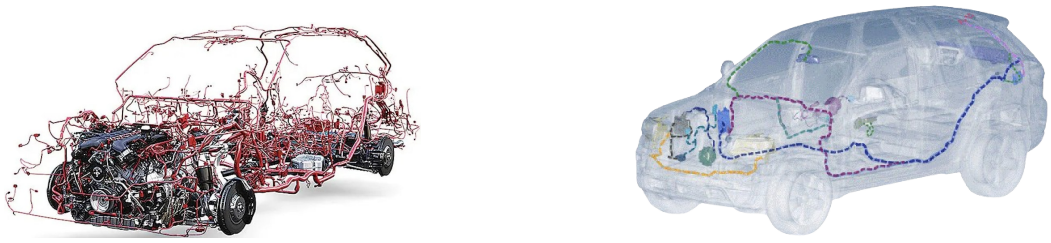

**Figure 1.** An overview of key features in a smart car. Various electronic control units communicate with each other using CAN-bus, connected in bus topology. Cyber attack injection can be wired through OBD-II port to CAN-bus or wireless by means of sensors.

### *2.1. Controller Area Network*

Owing to large number of ECUs and complex dense electrical wiring in the conventional cars which has increased the overall weight of the car and reduced fuel efficiency, there is a need of an efficient lightweight wiring mechanism. Thanks to the controller area network, the building block for infotainment and connectivity not only reduces the car weight but also transformes the concept of a smart car into reality. A general view of wiring in conventional and smart cars is presented in Figure 2.

The development of a controller area network dates back to early 1986 by Robert Bosch GmbH, who aimed to introduce an acceptable weight-saving communication strategy for the auto-industry. With the years of development stages, CAN was recognized by The International Standards Organization (ISO) as *ISO-11898* [45,46], in 1993. The CAN standard continued to evolve according to the growing demands of smart cars, resulting in the *ISO-11898-I* [45,46] flexible data CAN (CAN-FD) standard in 2005 followed by the start of CAN-XL development in 2018. Surprisingly, the independent operation of the new communication protocol like local area network (LIN) is not possible. Depending on CAN, the LIN operates on top of CAN or in its coordination, which signifies the role of CAN in SCs.

The physical structure of a CAN-bus consists of a twisted pair cable with two wires, CAN-low and CAN-high. CAN-low is used for smart operations which require low speed such as power windows, dashboard controls, and displays, whereas for high-speed applications and high data accuracy like engine control, anti-lock braking, and electronic stability control, CAN-high is used. The ends of the wires are connected with terminal resistors to reduce noise disturbance and ensure signal integrity over the bus.



(**a**) Dense, complex and heavy electrical wiring

(**b**) Lightweight 2-wire CAN-bus, an alternate to (**a**)

**Figure 2.** Representation of wiring in a car [47].

In addition, an on-board diagnostic port is also part of the CAN-bus system, which allows interfacing between external devices and ECUs via CAN-bus. Communication through CAN-bus employs the multi-master concept, i.e., any ECU in the network can take control of the bus, send message and request data from any other electronic control unit in the vehicle. In short, it is a message-oriented communication protocol with a data length of eight bytes and an up to 1 Mbps bit rate [48]. Typically, the CAN-bus protocol includes two types of data frames: standard and extended. The standard CAN frame utilizes an 11-bit identifier (2048 message identifier combinations) for the identification of the destination, which was later extended to 29 bits (537 million message identifier combinations) and given the name of an extended CAN frame. Figure 3 provides insight of a typical standard CAN frame.

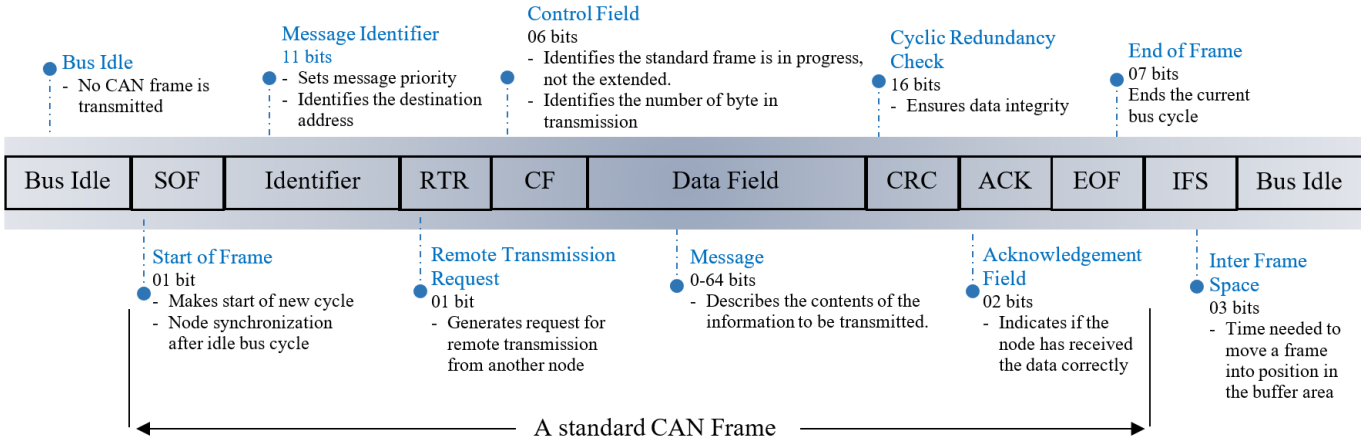



**Figure 3.** Representation of a typical CAN frame.

The functionality of SCs heavily relies on over the cloud connectivity in the cyber space, which opens the door for cyber attacks. In subsequent sections, we see various cyber attack surfaces and attacks in SCs.

### *2.2. Cyber Vulnerability*

By the expanded reliance on the connectivity to ensure intelligent smart functioning of a SC, the vulnerability risk to cyber attacks is increased, presenting various features as a soft lucrative target for the cyber criminals. During the last few years, many key features were identified as attack surfaces serving as entry points for intelligently crafted cyber attacks in smart cars. It includes keyless entry and ignition, airbag, electric window lift, media and infotainment system, and many others to name. Depending on software and sensors, the driver assistance system could be compromised by the remote assailant, disrupting the normal operations accordingly. Access to the infotainment system can be misleading and distracting for the driver. Insecure communication channels for telemetry can be exploited by malicious agents to transmit unauthorized commands. Threat actors can compromise the keyless entry system to unlock and start the car by introducing cyber attacks like a relay

attack. Hackers can gain access to the SCs function by the data breach in cloud services, which provide personalized services and predictive maintenance.

### *2.3. Impacts of Cyber Attacks*

Different types of cyber attacks can have serious impact, from disabling various key functions to disrupting car control and safety features. Primarily, we consider the CAN-bus spoofing attack where the hacker impersonates the legitimate user and sends manipulated data into the bus, which results in loss of critical control features such as steering, braking, and acceleration commands. Passenger security is compromised when the hackers successfully spoof the control mechanism and deactivate the car windows or doors, hijack and trap the riders. Compromised by the DoS attack, the communication and control network, among various ECUs facilitating sending and receiveng vital signals such as engine, adaptive cruise, and collison avoidance control, fails. It also disturbs airbags, lane keeping assistance, emergency braking system, traction control, and many other safety functions. Further, passenger safety is at risk when the DoS attack floods the network by sending enormous amounts of idle data prohibiting authentic ECU from sending valid data results in delayed emergency response and rescue measures. ADAS systems, being highly dependent on sensor data, may misfire the obstacle detection by the spoofed sensors sending misleading data values. Falsified sensor values can delude the driver guiding to locations where risks of injuries, collisions, or other perils to security are enormous. A compromised GPS system could drive the car to unwanted dangerous routes and sites. Infotainment systems infiltrated by intruders can change the infotainment display to flashing images ditracting the driver, which can increase the risk of possible accidents. Malicious remote disabling of the car electronic control system can leave commuters in dangerous situations, i.e., stranding in unsafe areas, losing control on the crucial security systems, and putting them at high risk of collisions especially during driving. Among all impacts, compromised keyless entry by impersonation attacks is the most dire. Once becoming successful in key fob cloning, the adversary can gain physical access to the car, allowing malevolent use of the car, and perform privacy invasion [49] which can result in sensitive data loss and leakage, change in the firmware, alteration in its cofiguration, alteration in the personal preferences and theft of driver identity to perform fraudulent activities. In short, the hacktavist gains full access to the car, enjoying the rights of the legal owner of the car. A summary of the various cyber attacks and their impacts on vehicle safety and passenger security is given in Figure 4.

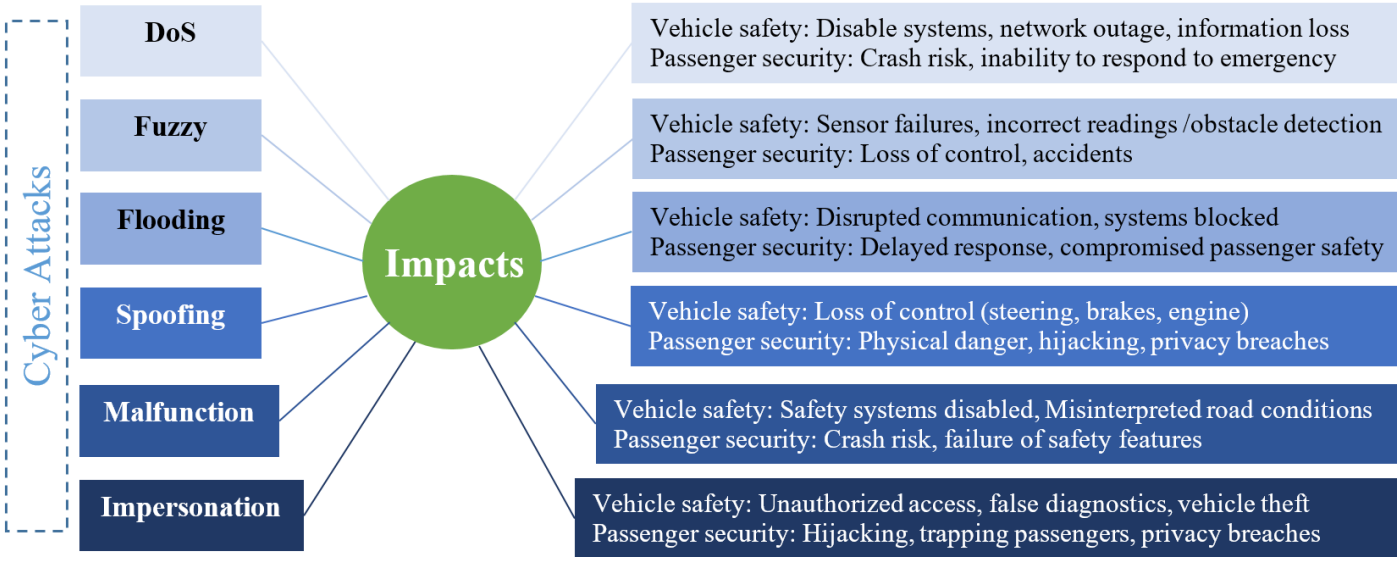



**Figure 4.** An overview of various cyber attacks and their impact on vehicle safety and passenger security.

In short, these attack surfaces can experience numerous intrusions initiated by the intruders to take control of the car. These flaws, intrusions and attacks can delay the data transmission, infuse false contents in the message, resend the same information multiple times, and so forth. Table 2 highlights different types of cyber attacks introduced in the CAN-bus data.

**Table 2.** An overview of various cyber attacks launched in CAN-bus messages to negatively impact the communication between different ECUs.

| Attack Type | Description |
| --- | --- |
| DoS (Denial of Service) | Transmission is interrupted by sending messages of the highest priority at unusually high speed. |
| Fuzzy | Actual data are manipulated by injecting random bits in arbitrarily selected identifiers. |
| Flooding | Authorized user is disconnected from the network by the congestion of network traffic with malicious data. |
| Spoofing | False messages are injected by the hacker pretending to be the authorized user. |
| Malfunction | Normal data are replaced with malicious data to impact data integrity. |
| Impersonation | A threat actor impersonates as a legitimate user to steal sensitive data. |

## 3. Proposed Methodology

While the development of smart cars facilitates the driving experience with connectivity and real-time information, the security risk associated with the car control and communication in the cyber space is tremendous. Numerous studies have been conducted to develop an attack detection system in smart cars, but still there is a gap to realize a simple, fast but lightweight and efficient self-defense mechanism. In this study, we propose a machine learning-based intrusion detection classifier employing six distinct real-time CAN-bus datasets collected from different cars including KIA Soul, Hyundai Avante CN7, Hyundai YF Sonata, CHEVROLET Spark, and Genesis g80. The datasets differ in attack types, car status, car model, and the collected time. The datasets and the proposed classifier are explained in detail in the following subsections.

### *3.1. Datasets*

While learning of an ML model highly depends on the information embedded in the data, selection of the dataset is crucial to build an ML-based intrusion detection classifier for critical applications like smart cars. In pursuit of this, the current study utilizes real-time data as a more realistic approach to propose an IDS. While working in the Hacking and Countermeasure Research Lab, South Korea, the data were collected by capturing the controller area network traffic through the OBD-II port connected to Raspberry pi3 coupled with a laptop through wireless connection. To obtain more accurate data, several different smart cars including Hyundai YF Sonata, KIA Soul, and Genesis g80 were tested with different types of cyber attacks introduced in the CAN-bus data [50]. Depending on the car model, type of attacks, injection frequency, and collection time, there were six independent datasets. These were: (1) Car Hacking: Along normal values, it contains intrusions launched on CAN-bus for 3 to 5 s in Hyundai YF Sonata; (2) Car Hacking: Attack and Defense Challenge 2020: Targeting the Hyundai Avante CN7, attacks were introduced with different injection frequencies; (3) In-vehicle Network Intrusion Detection Challenge: It includes CAN-bus data from CHEVROLET Spark, KIA Soul, and Hyundai Sonata collected while the car was in a stationary position; (4) CAN-Intrusion-Dataset (OTIDS): The CAN-bus data with intrusions were recorded from KIA Soul; (5) M-CAN Intrusion Dataset: M-CAN is a variant of CAN-bus where different multimedia devices and navigation modules communicate. Genesis g80 was used in this case; (6) CAN-FD Intrusion

Dataset: CAN-FD is an extension of CAN-bus with a high data rate. The CAN-bus data were collected from the smart cars released in 2021. Table 3 highlights the various features including attack types, dataset publication year, and a total number of samples for the mentioned datasets.

**Table 3.** An overview of 6 different CAN-bus datasets used in this study.

| Dataset | No. of Samples | Attack Types | Publication Year |
|---|---|---|---|
| 1 | 17,558,462 | DoS, Fuzzy, Spoofing (Gear), Spoofing (RPM) | 2018 |
| 2 | 8,694,507 | Flooding, Spoofing, Fuzzing, Reply | 2020 |
| 3 | 9,367,773 | Flooding, Fuzzy, Malfunction, Replay | 2018 |
| 4 | 4,613,909 | DoS, Fuzzy, Impersonation | 2017 |
| 5 | 2,952,620 | DDoS, Fuzzing | 2022 |
| 6 | 7,120,602 | Flooding, Fuzzing, Malfunction | 2022 |

### *3.2. Data Preprocessing*

In machine learning-based model development, data are normally preprocessed for better model training to improve performance and enhance convergence with better interpretation of data features. Preprocessing can be standardization- and normalization-based. Standardization is primarily used for the preprocessing of Gaussian distribution data, while for non-Gaussian data and where the impact of attacks and outliers is to preserve, normalization is used. The data used in this study are highly non-Gaussian, along with a significant impact of cyber attacks; therefore, normalization was used to preprocess the CAN-bus data. Given the varying range of data sent among different ECUs, the normalization helps in reducing the repercussions of differing sclaes. More specifically, Min–Max normalization is employed to scale down the data between 0 and 1. Avoiding the biasness and enhancing the model learning, this allows efficient convergence of the data required to optimize the detection accuracy and computational cost of the proposed classifier. Equation (1) explains the general normalization process, while Equation (2) is used to transform it to the 0 and 1 scale.

$$X = \frac{x - x_{min}}{x_{max} - x_{min}}, \tag{1}$$

$$X = \frac{x - x_{min}}{x_{max} - x_{min}}(N_{max} - N_{min}) + N_{min} \tag{2}$$

where $X$ and $x$ are normalized and acutal data points, $x_{max}$ and $x_{min}$ are the maximum and minimum values of the data, and $N_{max}$ and $N_{min}$ are the new maximum and minimum values when scaled down to 1 and 0.

### *3.3. Attack Design*

With the assumption that the attacker is fully aware of the network structure, an attack is launched which disrupts the normal network traffic, resulting in malfunctioning of the system. The attack design for various cyber attacks are given in Equation (3). For example, an intelligently crafted attack design includes injection of all zero ($0 \times 000$) values in the data for DoS, masqueraded and faked data for specific CAN ID in a spoofing attack, and random values for arbitrary CAN ID for a fuzzy attack. DoS is primarily performed by the blockade of service to the legitimate ECU by sending the CAN messages of the highest priority. In the CAN-bus network, any message with the $0 \times 000$ identifier is interpreted as of the highest priority, which actually contains no meaningful information except making the bus idle.

$$c = \begin{matrix} [0, 0, \ldots, 0], & for\ DoS\ attack \\ [s_1, s_1, \ldots, s_n], & for\ spoofing\ attack \\ random([s_1, s_2, \ldots, s_n]), & for\ fuzzy\ attack \end{matrix} \tag{3}$$

The input dataset after the addition of attack vector becomes

$$\hat{X} = X + c \tag{4}$$

where $X$ is the normal data values, without addition of an attack vector.

### *3.4. Proposed PD-ML Intrusion Detection Classifier*

In order to classify the CAN-bus data as normal or attacked, this study presents a novel machine learning-based lightweight intrusion detection classifier. To learn the pattern and identify relationships among different features of the data for accurate predictions on new data, an ML-based scheme requires splitting of input data into training and testing sets in a suitable proportion. In the proposed scheme, the classifier is trained using 70% of the total samples while holding out the remaining 30% samples used for testing. The selection of 70/30 is a normal practice adopted by researchers as in [51,52] for the development of machine learning-based models. In addition, performing an extensive working on algorithmic splitting, the authors of [53] proposed 70/30 as the best ratio for model training and testing. N-fold cross-validation is also opted in ML models, where N-1 $\times$ folds are used for training and 1 $\times$ fold is used for testing in each iteration, and the process is repeated N times. Consequently, the N-fold cross-validation approach ends with the development of a heavyweight ML model, increasing the computational time by N times, rendering its impracticality in the case of smart cars where a fast decision-making power is required in real time. Therefore, the 70/30 training/testing approach is adopted for developing a lightweight ML classifier. Further, splitting the data by 70/30 overcomes the model over-fitting problem, which normally results in 80/20 and 90/10 data division.

The proposed scheme is simple, lightweight and efficient involving (i) attack injection by the threat actor followed by its (ii) detection using a PD-based ML classifier. During the injection, the adversary acting as an authentic ECU gains unauthorized access to the smart car through the OBD-II port connected to Raspberry pi3 coupled with a laptop through wireless connection. The cyber intrusion is then injected into the CAN-bus, manipulating the normal data transmitted on the bus by the legitimate ECU. The affected ECU can be the engine control unit, the body control module, the brake control module, the battery management system, or any other. Next, the CAN-bus traffic including data from both normal and compromised ECUs are fed to the network monitoring unit for analysis and monitoring. The proposed PD-based ML intrusion detection classifier, a part of the network monitoring unit, collects all the incoming traffic, evaluates the data samples on trained PD-ML calculation (calculation process is explained in Section 3.4.1), and makes the decision by comparing the samples with the threshold value. A holistic view of the proposed classifier, including attack injection and detection, is given in Figure 5, and the pseudocode outlining the key steps involved in the classifier development for intrusion detection is presented in Algorithm 1.

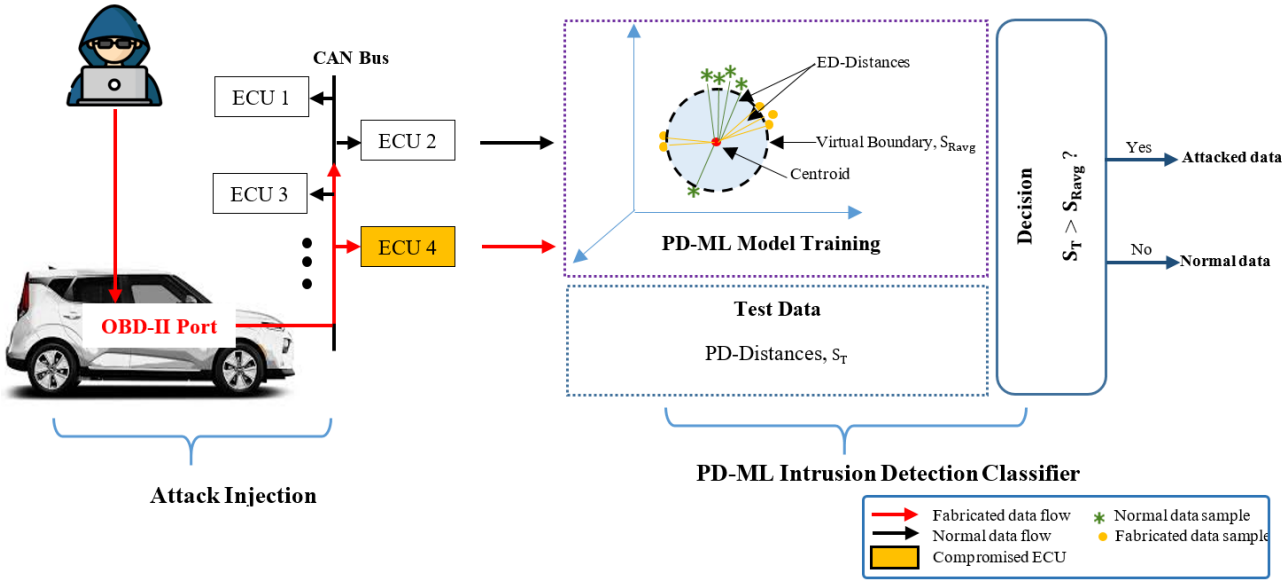



**Figure 5.** Representation of the proposed scheme.

**Algorithm 1** PD-based ML Classifier for Intrusion Detection

```
% Step-I Data Collection
    data = CAN_bus_data()
% Step-II Data Preprocessing
    X̂ = normalize(data)
% Step-III Data Splitting
    For input X̂()
            input.split f (train sample, test samples)
        For training samples
            Y = input.uplimit → 0.7 * X̂
        For test samples
            Z = input.uplimit+1:end → 0.3 * X̂
% Step-IV PD-ML Vectors Calculation
    Function = OutlierDetection()
        For input Y()
            calculate centroid, μ
            calculate PD vector, S_R,
            calculate average of S_R, S_Ravg
        For input Z()
            calculate PD vector, S_T,
% Step-V Intrusions Detection
        Check
            if S_T > S_Ravg
                Ŷ = It is compromised data
            else
                Ŷ = It is normal data
    end
% Step-VI Classifier Development
Classifier(inputs = X̂, outputs = Ŷ)
```

#### 3.4.1. Measurement of PD-ML Vectors

In line with the basic workflow of machine learning techniques, the input data $\hat{X} = [x_1, x_2, \ldots, x_m]$ where $m$ is the total number of samples are first splitted into a train sample $Y = [y_1, y_2, \ldots, y_a]$, where $a$ is the total number of samples, and a test sample $Z = [z_1, z_2, \ldots, z_b]$, where $b$ is the total number of samples. During the training phase, the

centroid $\mu$ of $m$ samples each withan $a$ number of features is calculated. The calculation of the centroid is the measure of central tendency which ensures that the impact of all the values in the data is considered, and variation in any single value is highlighted in the mean value. The pythagorean distance (PD) vector $S_R$ is then designed by measuring the distance of each point in the data pool from the centroid. Then, the model training ends with the calculation of average PD, $S_{R_{avg}}$. Next, the model testing begins with the measurement of the PD vector $S_T$ for test samples. The classifier model then decides for identification of normal and attacked values by checking the condition as $S_T > S_{R_{avg}}$. If the given condition is met, it is then an outlier; otherwise, it is a normal value. The set of equations governing the working of the proposed classifier includes Equations (5)–(8).

$$c_d = [\mu_1, \mu_2, \ldots, \mu_a], \tag{5}$$

where $\mu_i = \frac{1}{m}\sum_{i=1}^{a} x_i$

$$S_R = [S_{R1}, S_{R2}, \ldots, S_{Ra}], \tag{6}$$

where $S_{Rp} = \sqrt{\sum_{i=1}^{a}(\mu_i - y_i)^2}$, and $y_i$ is the training dataset, $p = [1, 2, \ldots, a]$.

$$S_{R_{avg}} = \frac{1}{m}\sum_{i=1}^{a} S_{Rp}, \tag{7}$$

$$S_T = [S_{T1}, S_{T2}, \ldots, S_{Tb}], \tag{8}$$

where $S_{Tq} = \sqrt{\sum_{i=1}^{b}(\mu_i - z_i)^2}$, and $z_i$ is the test dataset, $q = [1, 2, \ldots, b]$.

### *3.5. Performance Evaluation Metrics*

This section describes the elementary evaluation metrics used for the performance analysis of an ML-based classifier. It includes accuracy, F1-score, precision, recall, and receiver operating characteristic (ROC) curves. For the better understanding of the given metrics, let us first see the associated parameters governing these terms.

- True positive value, Tp: The actual positive value is correctly predicted as positive.
- False positive value, Fp: The actual negative value is falsely predicted as positive.
- True negative value, Tn: The actual negative value is correctly predicted as negative.
- False negative value, Fn: The actual positive value is falsely predicted as negative.

Accuracy is a measure of total correctly predicted values including both true positives and true negatives. F1-score is a measure of model effectiveness by evaluating its class-wise prediction skills contrary to accuracy which considers the overall model performance. It is the harmonic mean of recall and precision. Recall presents the measure of rightly predicted positives out of all the positive values in the given data. Precision offers an understanding of rightly measured positive values of all the predicted positives. All these metrics are significant in determining the efficiency of an ML model; however, precision is most relevant when considering the prediction of a particular target, i.e., attacked data in this case. Mathematical description of these metrics is given in Equations (9)–(12).

$$Acc = \frac{Tp + Tn}{Tp + Fp + Tn + Fn} \tag{9}$$

$$F_1 = 2\left(\frac{P_r \times R_e}{P_r + R_e}\right) \tag{10}$$

where precision $P_r$ and recall $R_e$ are computed as follows:

$$P_r = \frac{Tp}{Tp + Fp} \tag{11}$$

$$R_e = \frac{Tp}{Tp + Fn} \tag{12}$$

#### 3.5.1. ROC Curve

The receiver operating characteristic curve contributes significantly to evaluating the performance of a binary classifier, which is a case this study deals with. It is a graphical representation of the model's sensitivity (sensitivity is analogous to recall) (true positive rate, Equation (13)) and specificity (false positive rate, Equation (14)) with area under the curve (AUC) summarizing the overall model's performance. The curve closer to the top-left corner where sensitivity and specificity are high represents the optimal model's performance. Moreover, an AUC value approaching one or zero presents the best or the worst model development.

$$Sensitivity = \frac{Tp}{Tp + Fn} \tag{13}$$

$$Specificity = \frac{Fp}{Fp + Tn} \tag{14}$$

#### 3.5.2. Error Metrics

Along other performance measuring metrics, error metrics has great significance in analyzing the performance of a machine learning classifier. Primarily, it is a quantitative measure of performance discrepancy and a function of measured output and maximum theoretical output as given in Equation (15). In this study, absolute error is calculated using Equation (16), which is a difference in the measured and actual values, and it offers a clear insight to assess the classifier's reliability and effectiveness.

$$Error = f(\hat{Y}, max\ \hat{Y}) \tag{15}$$

$$Absolute\ Error = \left|\hat{Y} - \max \hat{Y}\right| \tag{16}$$

## 4. Results

In order to evaluate the performance of the proposed classifier, various widely accepted CAN-bus datasets with multiple attack designs are employed. As described above, many simulation parameters are used for the model configuration, performance optimization, and overcoming model over-fitting and under-fitting problems.

The simulation results presented in Tables 4–6 show an extraordinary performance for basic requirements from an ML model, i.e., simultaneously obtaining high accuracy, good precision, commendable recall, and excellent F1-score. There are indicators of overall correctness, minimized false positive predictions, and model suitability for the imbalanced data.

### *4.1. Accuracy*

Accuracy being the measure of overall correct predictions by the classifier is calculated using Equation (9), and the obtained results are given in Table 4. These results show outstanding classifier performance, as with >99% efficiency it correctly classifies the attacked and normal data for a range of cyber attacks introduced in the CAN-bus system of different types of SCs. Specifically for the fuzzing attack in Dataset 5, the accuracy measure was 99.922% followed by 99.716%, 99.691%, 99.691%, and 99.568% for the spoofing attack in Dataset 2, and fuzzy, impersonation, and DoS attacks in Dataset 4, respectively. The range of accuracy measurements for different attacks is as follows: 97.833–99.568% (DoS), 98.167–99.716% (spoofing), 99.691–99.922% (fuzzy), and 95.0–99.412% (flooding). The malfunction and impersonation attacks in Datasets 3 and 4 also presented measurable accuracy results of 98.722% and 99.691%. However, the lowest achieved accuracy value was 95.0% for the flooding attack in Dataset 3, and the possible reason of this deviation by 5% from the maximum value could be the dataset size and or inappropriate classifier tuning

for classification. Summing up, the obtained results are sure proof of the dynamic nature of the proposed classifier and its ability to detect the attacks with high degree of accuracy.

**Table 4.** Accuracy measure of the proposed classifier for six datasets under different attack designs.

| Datasets | Attack Type | Accuracy (%) |
|---|---|---|
| Dataset 1 | DoS | 97.833 |
| | Spoofing (Gear) | 99.50 |
| | Spoofing (RPM) | 98.167 |
| Dataset 2 | Flooding | 99.390 |
| | Spoofing | 99.716 |
| Dataset 3 | Flooding | 95.00 |
| | Malfunction | 98.722 |
| Dataset 4 | DoS | 99.568 |
| | Fuzzy | 99.691 |
| | Impersonation | 99.691 |
| Dataset 5 | DoS | 98.585 |
| | Fuzzing | 99.922 |
| Dataset 6 | Flooding | 99.412 |

### *4.2. F1-Score*

F1-score, a measure of the classifier reliability and its effectiveness in making class-wise predictions, is measured using Equation (10), and the calculated results are presented in Table 5. Normally, a scale of 0–1 is used for the efficiency measurement of F1-score: the closer the value is to one, the better the performance, and vice versa. Satisfying the classifier performance, the F1-score for all the attack designs and datasets is almost one in the range of 0.95541–0.99961 with a maximum and minimum absolute errors of 0.04459 and 0.00039, respectively. Analyzing the datasets, Datasets 2, 4, 5, and 6 performed exceptionally well with the highest F1-scores of 0.99857, 0.99844, 0.99961, and 0.99488, respectively. However, Datasets 1 and 3 also showed remarkable performance with the lowest values of 0.96073 and 0.95541. Attack-wise analysis showed that the best results were obtained for fuzzy attacks (Absolute Error: 0.00156–0.00039), followed by spoofing attacks (Absolute Error: 0.02949–0.00143), DoS attacks (Absolute Error: 0.03927–0.00219), and flooding attacks (Absolute Error: 0.04459–0.00375). F1-score for the malfunction and impersonation attacks in Datasets 3 and 4 is also apprehendable, showing values of 0.98432 and 0.99844, respectively. Summarizing, the proposed classifier is highly reliable in attack classification and detection for the imbalanced datasets. Note that all the datasets considered are imbalanced as the sample size for attacked and normal data is different.

### *4.3. Precision*

Precision, a measure of correct positive predictions out of all the predictions, is calculated using Equation (11), and the measured results are shown in Table 6. The proposed classifier showed extraordinary performance for the fuzzing attack in Dataset 5, with a precision score of 0.999, reducing error to 0.001. In the same fashion, the classifier's outcome for other attack designs and datasets is also up to mark, showing precision of 0.997, 0.996, 0.993, 0.99, 0.985, and the list goes on with all values $> 0.9$ ($\approx$1). The maximum absolute error found in precision results is 0.085, and this could be due to imbalanced data. Wrapping up, the output of the proposed classifier proves its capability to identify attacks, anomalies and flaws in the controller area network in smart cars with minimal false positives and high precision.

**Table 5.** F1-score measure of the proposed classifier for six datasets under different attack designs.

| Datasets | Attack Type | F1-Score |
|---|---|---|
| Dataset 1 | DoS | 0.96024 |
| | Spoofing (Gear) | 0.9923 |
| | Spoofing (RPM) | 0.97051 |
| Dataset 2 | Flooding | 0.99625 |
| | Spoofing | 0.99857 |
| Dataset 3 | Flooding | 0.95541 |
| | Malfunction | 0.98432 |
| Dataset 4 | DoS | 0.99781 |
| | Fuzzy | 0.99844 |
| | Impersonation | 0.99844 |
| Dataset 5 | DoS | 0.99266 |
| | Fuzzing | 0.99961 |
| Dataset 6 | Flooding | 0.99488 |

**Table 6.** Precision measure of the proposed classifier for six datasets under different attack designs.

| Datasets | Attack Type | Precision |
|---|---|---|
| Dataset 1 | DoS | 0.929 |
| | Spoofing (Gear) | 0.985 |
| | Spoofing (RPM) | 0.943 |
| Dataset 2 | Flooding | 0.993 |
| | Spoofing | 0.997 |
| Dataset 3 | Flooding | 0.915 |
| | Malfunction | 0.969 |
| Dataset 4 | DoS | 0.996 |
| | Fuzzy | 0.997 |
| | Impersonation | 0.997 |
| Dataset 5 | DoS | 0.985 |
| | Fuzzing | 0.999 |
| Dataset 6 | Flooding | 0.99 |

In summary, the precision score for all the datasets and the attack designs is above 0.9 achieving the maximum value of 0.999 (Absolute Error: 0.001) This is a clear indication of the outstanding performance by the proposed classifier in accurately predicting the true positives with little or no false positive predictions. In addition, the detection accuracy and F1-score in all the cases are more than 99% and 0.99, respectively, except for the DoS and spoofing (RPM) attacks in Dataset 1, malfunction and flooding attacks in Dataset 3, where the accuracy values are 97.833%, 98.167%, 98.722%, and 95.0% with 0.96073, 0.97051, 0.98432, and 0.95541 F1-score, respectively. These values are even close to the perfect results (1.0) with a maximum error of 0.04459.

### *4.4. Recall*

Accuracy, F1-score, and precision posess inbuilt trade-offs like insensitivity to imbalanced data and no differentiation between false negative and false postivie errors by the accuracy measure. F1-score offers equal weight to precision and recall and does not count for true negatives, which is important in spam filtering for incorrect flag detection.

Precision focuses only on false positives and is inappropriate when false negatives are costly. In these scenarios, the proposed model needs to be evaluated for recall, ROC-AUC to check its authenticity while applying for imbalanced data or in applications where true negatives or false negatives are important. This section evaluates the recall rate, while the next section discusses ROC and AUC for the proposed classifier.

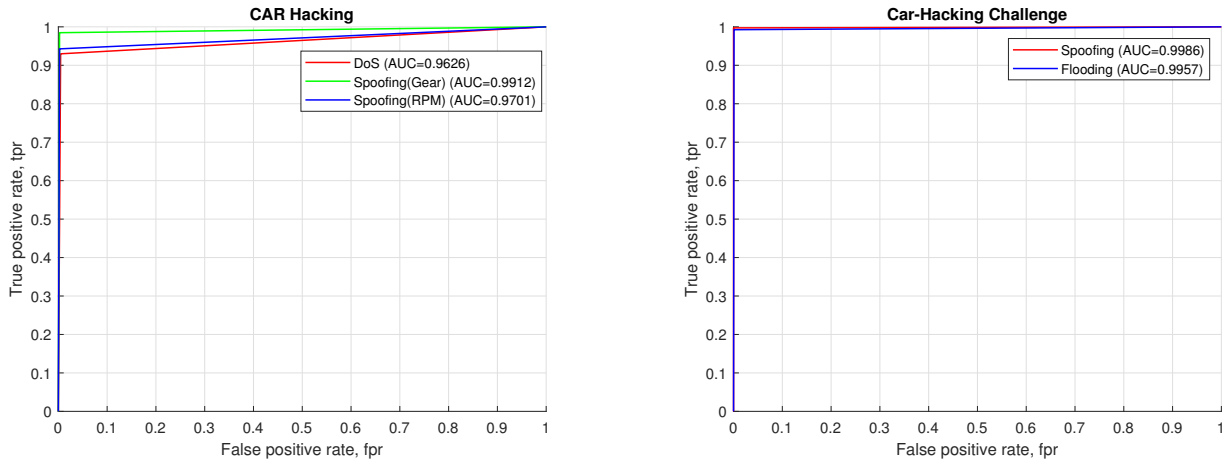
Recall, a measure of correctly predicted positives of all the positives in the given data, is calculated using Equation (12), and the calculated results are presented in Table 7. The results are remarkable, where the actual positive instances are 100% accurately predicted as positives for all the attack designs except for the DoS anomaly in the Car Hacking dataset. The possible reason for this discrepancy is more imbalanced data distribution in DoS attacks compared to other anomalies disturbing the normal functioning of a smart car. The presence of a noise element could also contribute towards this slight deviation from the maximum achievable value. Ensuring suitability, these results encourage the adoption of the proposed classifier in smart cars for intrusion detection.

**Table 7.** Recall measure of the proposed classifier for six different datasets under different attack designs.

| Datasets | Attack Type | Recall |
|---|---|---|
| Dataset 1 | DoS | 0.994 |
| | Spoofing (Gear) | 1.0 |
| | Spoofing (RPM) | 1.0 |
| Dataset 2 | Flooding | 1.0 |
| | Spoofing | 1.0 |
| Dataset 3 | Flooding | 1.0 |
| | Malfunction | 1.0 |
| Dataset 4 | DoS | 1.0 |
| | Fuzzy | 1.0 |
| | Impersonation | 1.0 |
| Dataset 5 | DoS | 1.0 |
| | Fuzzing | 1.0 |
| Dataset 6 | Flooding | 1.0 |

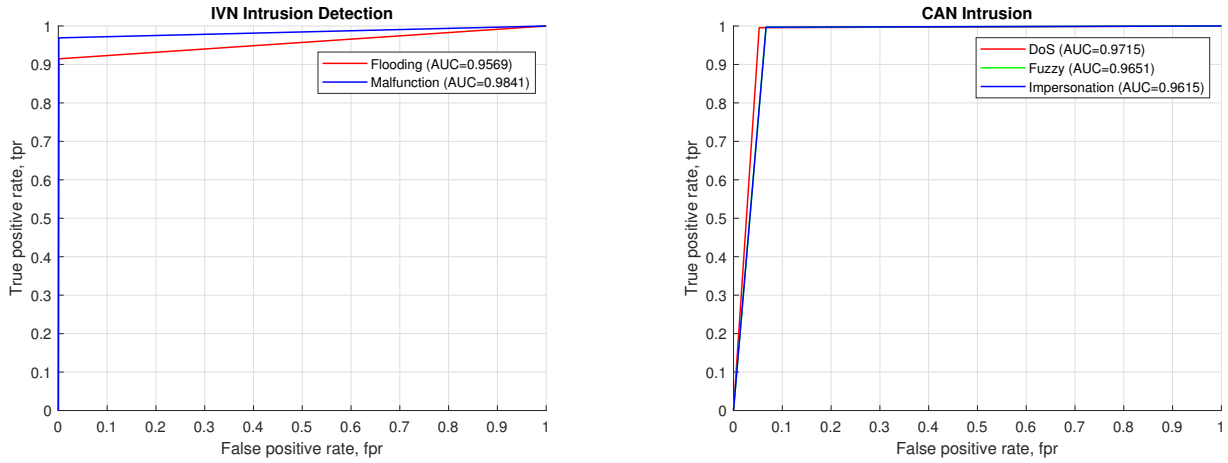
### *4.5. ROC Curves*

Receiver operating characteristic curves offer a visual insight into the overall model performance and robustness along with the area under the curve demonstrating the effectiveness of the model. Describing the discriminative power of a machine learning classifier, the area under the curve (AUC) value is a good mearue to distinguish a normal sample from a malicious one at a probability scale of zero and one. The AUC value $0 < AUC < 0.5$ indicates an issue in the classifier design, $AUC = 5$ reflects randomness in anomaly detection, whereas $0.5 < AUC < 0.7$ shows a suboptimal classifier, $0.7 < AUC < 0.8$ represents a decent classifier, $0.8 < AUC < 9$ indicates a good classifier, and $AUC > 0.9$ reflects an excellent classifier with high discriminating capability. Following the norms of the literature, a 95% confidence Interval is selected for calculating AUC values. Figures 6–8 show ROC curves obtained from the output of the proposed classifier for multiple cyber attacks introduced in CAN-bus data of different smart cars. It is clearly visible that all the curves (bends) are closely aligned with the top left corner where sensitivity and specificity values are high. Further, the AUC value is above 0.95, almost approaching 1 for all ROC curves with a minimum value being 0.9569, representing the high reliability and applicability of the proposed classifier for intrusion detection in the controller area network in smart cars.



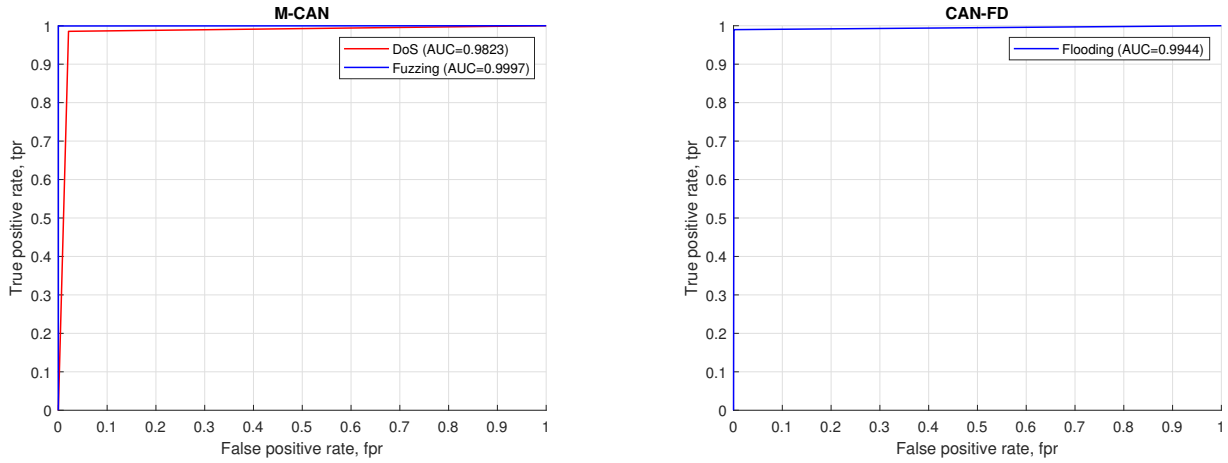

(**a**) (**b**)

**Figure 6.** ROC curves for the proposed ML classifier for different cyber attacks in (**a**) car hacking and (**b**) car-hacking challenge datasets.




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

**Figure 7.** ROC curves generated using the outcome of the proposed ML classifier for different cyber attacks in (**a**) IVN Intrusion and (**b**) CAN Intrusion datasets.




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

**Figure 8.** ROC curves generated using the outcome of the proposed ML classifier for different cyber attacks in (**a**) M-CAN and (**b**) CAN-FD datasets.

### *4.6. Confusion Matrices*

Lastly, the measure of any machine learning performance parameter is directly linked with the calculation of the confusion matrix. It helps visualize the potency of the ML classifier by evaluating the actual and predicted values. Figure 9 shows the general structure of a confusion matrix. Given the same nature of the actual and predicted values, the outcome is a true function and vice versa. Analysis of the confusion matrix involves the values in its diagonal. The primary diagonal (true) values should count to the maximum while the secondary diagonal (false) values should approach zero, ideally be zero instead. Given the various datasets and cyber attacks in this study, the confusion matrices obtained from the proposed classifier are given in Figures 10–13. Having a close look at these matrices, all the true positives and true negatives show a reasonably good count of correct predictions. The false positives are all zero except for the first case where it is 'one', indicating exceptionally high recall and precision rates. The possible reason of low false positives and negatives is the classifier's sensitivity towards predicting normal instances when certain anomaly encounters smart cars. While developing the model, the number of samples varies as attack type varies; this is because of overall variation in the data size. We focus on the CAN-Intrusion and M-CAN datasets, where the true positives are low, which is conventionally accepted to be high in the case of high accuracy, F1-score, and precision. The low value of true positives primarily reflects the low number of positive samples in the datasets, indicating the presence of imbalanced data linked with diverse functionalites of smart cars.

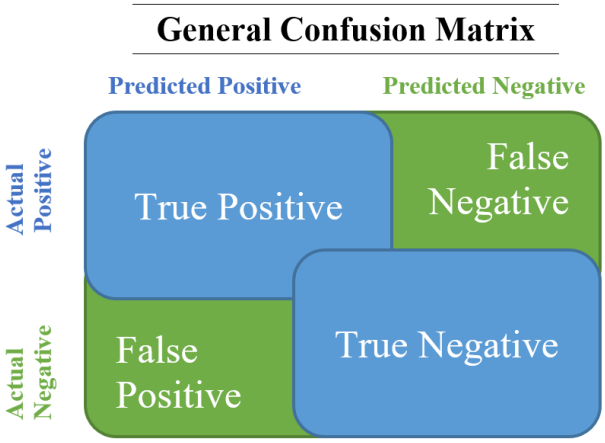



**Figure 9.** General representation of a confusion matrix of an ML classifier.

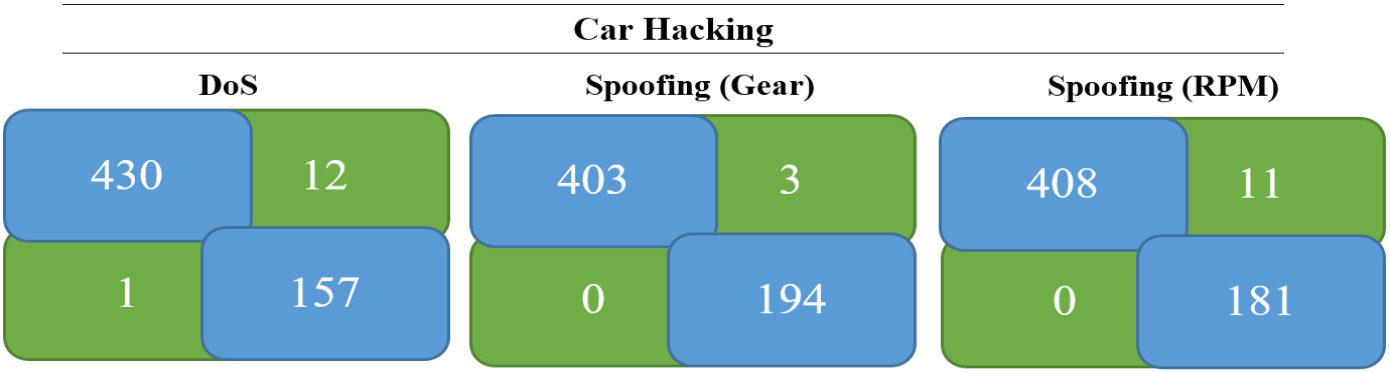



**Figure 10.** Confusion matrix of the proposed ML classifier for different cyber attacks in the car hacking dataset.

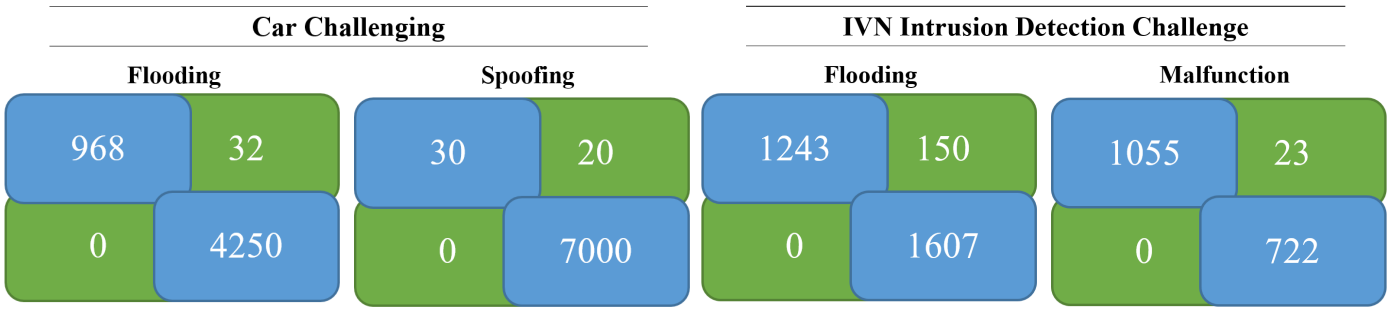



**Figure 11.** Confusion matrices of the proposed ML classifier for different cyber attacks in (**a**) challenging and (**b**) IVN Intrusion detection challenge datasets.

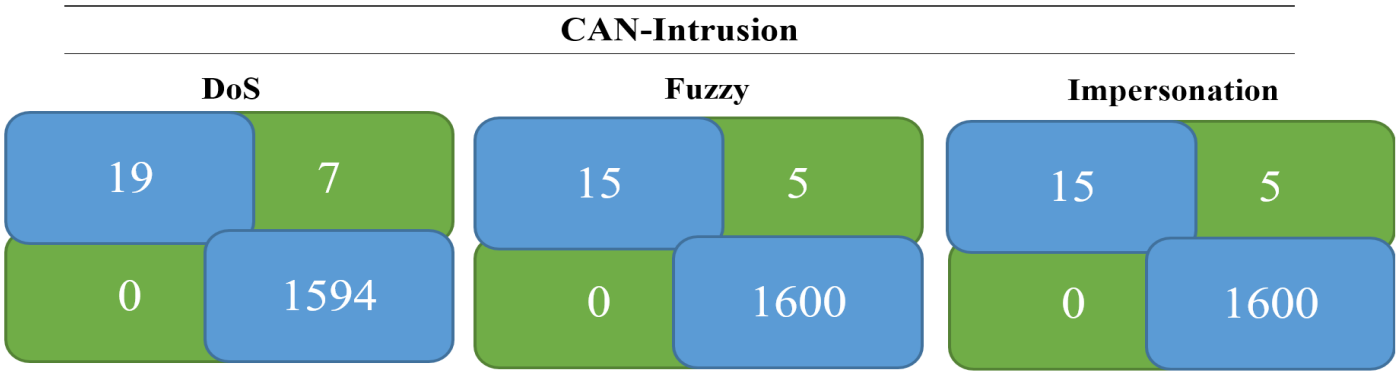



**Figure 12.** Confusion matrix of the proposed ML classifier for different cyber attacks in the CAN-Intrusion dataset.

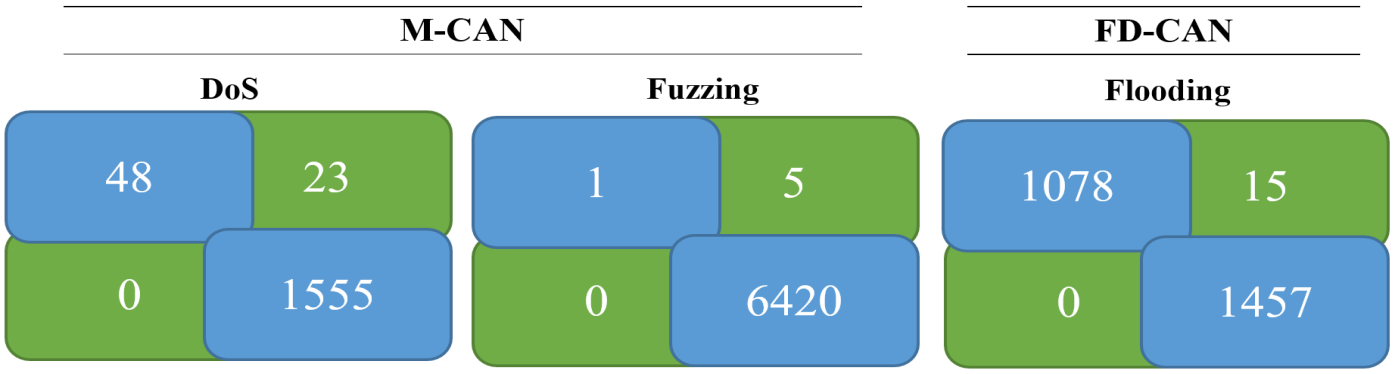



**Figure 13.** Confusion matrices of the proposed ML classifier for different cyber attacks in M-CAN and FD-CAN datasets.

### *4.7. Performance Comparison*

#### 4.7.1. Computational Time

Computational time plays a key role in real-time cyber attack detection. It is a measure of the total time required for building the model followed by model testing. The simulation environment for the computation of the proposed classifier uses MATLAB 2023a installed on the Microsoft Windows 11 Pro operating system in the computational machine with an Intel(R) Core(TM) i7-1165G7 @ 2.80GHz processor with 12 GB RAM.

While evaluating the time complexity of various traditional machine learning schemes for cyber attack detection, it is found that the proposed classifier detects the intrusion in a CAN-bus network in smart cars at greatly reduced time, evident from Table 8. Utilization of the quadratic programming to find the hyperplane for data segregation contributes to the increased computational complexity of SVM taking more time to tune the model limiting

the use in fast detection applications. Compared to SVM, the computational cost for NN is low, but is 99.99% higher compared to the proposed scheme. The probable cause could be multiple hidden layers, the batch processing and backpropagation phenomenon, which is an integral part of the neural network learning process. With the inherent nature of the idle during training phase, the kNN is considered a lazy approach. Extending all efforts to calculate the threshold value and make predictions during the testing phase increases the computational cost considerably. The recursive nature of DT, which involves repetition of the training process for optimal selection of split to classify data as compromised or normal, increases computational time by many times. Compared to all these baseline methods, the proposed scheme is free from recursive nature, backpropagation, and the quadratic programming issue. Further, it calculates the threshold value during the training phase, where during testing, simple comparison is made between the incoming value and the threshold value. Similarly, comparison with some studies which have proposed different cyber intrusion detection mechanisms given in Table 9 reflected an extraordinary performance by the proposed classifier in reducing computational complexity.

**Table 8.** Computational time-based performance comparison of the proposed study with existing ML schemes.

| ML Schemes | Computational Time (s) |
|---|---|
| Support Vector Machine | 25.71253 |
| Neural Network | 1.735529 |
| k-Nearest Neighbor | 14.20163 |
| Decision Tree | 10.97992 |
| Proposed study | 0.058697 |

**Table 9.** Computational time-based performance comparison of the proposed study with existing studies.

| References | Computational Time (s) | |
|---|---|---|
| | Build Time | Execution Time |
| He et al. [24] | 2.42 | 0.94 |
| Khan et al. [42] | 403 | 3 |
| Ullah et al. [54] | - | 51 |
| Kang et al. [44] | 4.15 | 0.0020584 |
| Proposed study | 0.058427 | 0.00027 |

#### 4.7.2. Distance Metrics-Based Comparison

Distance metrics can be used for classification and clustering of points with similar characteristics. As the proposed scheme relies on Pythagorean distance, its comparison with other distances is necessary to justify its choice for the classifier design. It is given in Table 10. It shows that the proposed distance performed extraordinarily good with a 38% higher accuracy than the Mahalanobis distance followed by 24%, 5.7%, and 3% better performance than the Manhattan, Minkowski, and Chebyshev distances, respectively.

**Table 10.** Distance metrics-based performance comparison.

| Distances | Accuracy (%) | F1-Score |
|---|---|---|
| Manhattan | 75.746 | 0.867875 |
| Minkowski | 94.154 | 0.89764 |
| Mahalanobis | 61.94 | 0.57382 |
| Chebyshev | 96.891 | 0.94279 |
| Proposed study | 99.922 | 0.99961 |

#### 4.7.3. Comparison with Existing ML Schemes

Machine learning schemes are well-established techniques for classification of normal and compromised data. Comparison of the proposed scheme with existing ML schemes is crucial in order to validate its performance. Table 11 presents performance comparison with some ML schemes. It clearly shows the outstanding performance of the proposed scheme in cyber attack detection for the chosen datasets. The proposed scheme showed 2.4%, 2.1%, 2.1%, and 1.7% better performance in accuracy compared to ML schemes, where this difference in F1-score is 4.5%, 3.9%, 3.9%, and 3.4%. As the smart car data are highly non-linear, linked with diverse functionalities, the SVM showed less efficiency as one class dominates over the other class in the dataset. Neural networks showed good performance but were comparatively less efficient, probably because the number of neurons are in the saturation region, which does not contribute to the learning process, leading to reduced efficiency. Other possible reasons could be the convergence issue, overfitting, and or underfitting problems. Comparatively low efficiency of NN highlights the sensitivity of NN to variations in the data. Small changes in the smart car data result in significant change in the DT structure, reducing the learning efficiency which decreases the overall efficiency. Conversely, the proposed scheme is sound as the data sensitivity is overcome by calculating the centroid, and there is no concept of the saturation region.

**Table 11.** Performance comparison of the proposed study with existing ML schemes.

| ML Schemes | Accuracy (%) | F1-Score |
|---|---|---|
| Support Vector Machine | 97.5 | 0.95441 |
| Neural Network | 98.2 | 0.96552 |
| k-Nearest Neighbor | 97.833 | 0.96024 |
| Decision Tree | 97.833 | 0.96024 |
| Proposed study | 99.922 | 0.99961 |

#### 4.7.4. Comparison with Recent Studies

A comprehensive comparison of the proposed system and the various relevant studies is presented in Table 12. Researchers have investigated the various ML-based anomaly detection classifiers both for single and multiple attack types. A comprehendible detection accuracy of 99.8% and 99.43% is achieved for single and multiple type of cyber assaults, respectively; however, these studies offer no information about F1-score and precision results, which are important parameters while analyzing the efficiency of any ML classifier. The other relevant studies which include all the evaluation metrics presented good performance results, but the proposed study outperformed these with percentage gains of 1.163, 1.862, and 1.301 in accuracy, F1-score, and precision, respectively. Moreover, most of the studies remained silent when it comes to the area under the curve. AUC offers better understanding of the classifier's performance, measuring how well the classifier distinguishes between attacked and normal data. The proposed classifier AUC could achieve a 0.9997 score, showing a difference of 0.0003 from the ideal result, i.e., AUC = 1.0.

Generalizability of an IDS lies in its effectiveness to detect anomalies on unseen data. Utilizing the CAN-bus data, which represents the overall behavior of SCs, the proposed classifier was tested on new unseen real-time data collected from 6 different smart cars, i.e., Hyundai YF Sonata, Hyundai Avante CN7, CHEVROLET Spark, KIA Soul, Hyundai Sonata, and Genesis g80. Additionally, employing the 13 unique attack scenarios, the model presented a comprehendible performance as discussed above.

**Table 12.** Performance comparison of various machine learning-based studies for anomaly detection in smart cars. Accuracy is measured in percentages, whereas for the rest of the parameters, a scale of 0–1 is used, where 0 indicates the minimum value and 1 shows the maximum achievable number.

| References | Attack Types | Accuracy | Precision | F1-Score | AUC |
|---|---|---|---|---|---|
| Kishore et al. [22] | Multiple | 99.09 | 0.9901 | 0.9910 | 0.9990 |
| Alalwany et al. [29] | Multiple | 98.5 | 0.987 | 0.985 | 1.0 |
| Altalbe [32] | Multiple | 99.2207 | 96.3853 | 97.3098 | - |
| Pascale et al. [36] | Multiple | - | 0.986 | 0.981 | - |
| Salek et al. [40] | Single | 98.3 | - | - | - |
| Khan et al. [42] | Multiple | 86 | 0.9493 | 0.9497 | - |
| Ahmed et al. [55] | Single | 99.8 | - | - | - |
| Aloqaily et al. [56] | Multiple | 99.43 | - | - | - |
| Zhu et al. [57] | Multiple | 90.1 | 0.856 | 0.913 | - |
| Seo et al. [58] | Multiple | 98 | 0.983 | - | 0.999 |
| Loukas et al. [59] | Multiple | 86.9 | - | - | - |
| Kang & Kang [44] | Single | 97.8 | - | - | - |
| Yang et al. [60] | Single | 98.76 | - | - | - |
| Aldhyani & Alkahtani [23] | Multiple | 97.3 | 0.97 | 0.96 | - |
| Mehedi et al. [61] | Multiple | 98.1 | 0.9814 | 0.9783 | 0.9542 |
| Ma et al. [62] | Multiple | - | 0.9995 | 0.9992 | - |
| This study | Multiple | 99.922 | 0.999 | 0.99961 | 0.9997 |

## 5. Conclusions

The controller area network is an integral element to realize the concept of smart cars. It establishes communication among different electronic control units for various intelligent functionalities such as engine management, the braking system, remote monitoring and control, etc. However, owing to the inherited broadcasting and non-delineating nature, CAN-bus is highly vulnerable to cyber assaults and intrusions presenting serious security risks to the passengers. With the purpose of enhancing SC security, a PD-ML-based intrusion detection classifier is proposed to detect the cyber attack and separate it from the normal network traffic while employing real CAN-bus datasets with different types of attacks launched in the system. The simulation results showed a detection accuracy of 99.922% with a 0.99961 F1-score, 0.999 precision, and 1.0 recall rate in identifying and detecting the intrusions in the CAN-bus data with computational time reduced by 99.77% 99.58%, 99.47%, 96.61%, and 86.67% compared to existing schemes. The performance statistics indicate that the proposed model is highly sensitive to positive predictions, making it a good choice for smart cars, directly connected with the life of the commuters. The proposed classifier is a decent choice for the imbalanced datasets and can be utilized in critical applications like spam identification, fraud detection, healthcare monitoring, smart grids, autonomous drones, financial analytics, and many others. Additionally, this study can be extended to address the impact of cyber attacks not discussed here. Further, the development of firmware compatible with the central gateway and the target ECU in SCs is a key adaption challenge and could be an extension of this study.

**Author Contributions:** Conceptualization, Z.A.K., S.A. and A.A.; methodology, S.A., Z.A.K. and A.K.; software, A.K.; validation, S.A. and A.K.; formal analysis, Z.A.K., S.A. and A.K.; investigation, A.K.; resources, Z.A.K. and S.A.; data curation, A.K.; writing—original draft preparation, A.K., S.A., A.A. and Z.A.K.; writing—review and editing, A.K., S.A., A.A. and Z.A.K.; visualization, A.K.; supervision, Z.A.K. and S.A.; funding, Z.A.K. and A.A. All authors have read and agreed to the published version of the manuscript.

**Funding:** The author extends the appreciation to the Deanship of Postgraduate Studies and Scientific Research at Majmaah University for funding this research work through the project number (R-2024-1300).

**Data Availability Statement:** Data available in a publicly accessible repository.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

| | |
|---|---|
| BN | Bayseian Network |
| CAN | Controller Area Network |
| CNN | Convolutional Neural Network |
| DNN | Deep Neural Network |
| DT | Decision Tree |
| ECUs | Electronic Control Units |
| EM | Ensemable Methods |
| IVN | In-Vehicle Network |
| KNN | k-Nearset Neighbour |
| LW | LightWeight-based Machine Learning |
| ML | Machine Learning |
| MLP | Multi-Layer Perceptron |
| NB | Naive Bayes |
| NN | Neural Networks |
| OBD | On-Board Diagnostic |
| RF | Random Forest |
| RT | Random Tree |
| SGD | Stochastic Gradient Descent |
| SVM | Support Vector Machine |
| TCAN | Temoral Convolutional Attention Network-based Machine Learning |

## References


1. Alsaade, F.W.; Al-Adhaileh, M.H. Cyber attack detection for self-driving vehicle networks using deep autoencoder algorithms. *Sensors* **2023**, *23*, 4086. [CrossRef] [PubMed]
2. Andy, G. Hackers Remotely Kill a Jeep on the Highway—With Me in It. 2015. Available online: https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/ (accessed on 6 June 2024).
3. Olivia, S. Team of Hackers Take Remote Control of Tesla Model S from 12 Miles Away. 2016. Available online: https://www.theguardian.com/technology/2016/sep/20/tesla-model-s-chinese-hack-remote-control-brakes (accessed on 6 June 2024).
4. Eduard, K. API Flaw Exposes Nissan LEAF Cars to Remote Attacks. 2016. Available online: https://www.securityweek.com/api-flaw-exposes-nissan-leaf-cars-remote-attacks/#:~:text=An%20API%20used%20by%20Nissan,best%20selling%20all%2Delectric%20car (accessed on 6 June 2024).
5. Arghire, I.. 16 Car Makers and Their Vehicles Hacked via Telematics, APIs, Infrastructure. 2023. Available online: https://www.securityweek.com/16-car-makers-and-their-vehicles-hacked-telematics-apis-infrastructure/ (accessed on 6 June 2024).
6. Dikmen, M.; Burns, C.M. Autonomous driving in the real world: Experiences with tesla autopilot and summon. In Proceedings of the 8th International Conference on Automotive User Interfaces and Interactive Vehicular Applications, Ann Arbor, MI, USA, 24–26 October 2016; pp. 225–228.
7. Fagnant, D.J.; Kockelman, K. Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations. *Transp. Res. Part A Policy Pract.* **2015**, *77*, 167–181. [CrossRef]
8. Eustice, R. *University of Michigan's Work Toward Autonomous Cars*; University of Michigan: Ann Arbor, MI, USA, 2015.
9. Connected and Autonomous Vehicle Research and Development Projects. 2018. Available online: https://assets.publishing.service.gov.uk/media/5b8d327840f0b67daf8069fd/ccav-research-and-development-projects.pdf (accessed on 3 March 2024).
10. Shafin, S.S.; Karmakar, G.; Mareels, I. Obfuscated memory malware detection in resource-constrained IoT devices for smart city applications. *Sensors* **2023**, *23*, 5348. [CrossRef] [PubMed]
11. Shafin, S.S.; Karmakar, G.; Mareels, I.; Balasubramanian, V.; Kolluri, R.R. Sensor Self-Declaration of Numeric Data Reliability in Internet of Things. *IEEE Trans. Reliab.* **2024**. [CrossRef]
12. Bukhari, S.M.S.; Zafar, M.H.; Abou Houran, M.; Moosavi, S.K.R.; Mansoor, M.; Muaaz, M.; Sanfilippo, F. Secure and privacy-preserving intrusion detection in wireless sensor networks: Federated learning with SCNN-Bi-LSTM for enhanced reliability. *Ad Hoc Netw.* **2024**, *155*, 103407. [CrossRef]
13. Bukhari, S.M.S.; Zafar, M.H.; Abou Houran, M.; Qadir, Z.; Moosavi, S.K.R.; Sanfilippo, F. Enhancing cybersecurity in Edge IIoT networks: An asynchronous federated learning approach with a deep hybrid detection model. *Internet Things* **2024**, *27*, 101252. [CrossRef]
14. Bari, B.S.; Yelamarthi, K.; Ghafoor, S. Intrusion detection in vehicle controller area network (can) bus using machine learning: A comparative performance study. *Sensors* **2023**, *23*, 3610. [CrossRef]

15. Moulahi, T.; Zidi, S.; Alabdulatif, A.; Atiquzzaman, M. Comparative performance evaluation of intrusion detection based on machine learning in in-vehicle controller area network bus. *IEEE Access* **2021**, *9*, 99595–99605. [CrossRef]
16. Kavousi-Fard, A.; Dabbaghjamanesh, M.; Jin, T.; Su, W.; Roustaei, M. An evolutionary deep learning-based anomaly detection model for securing vehicles. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 4478–4486. [CrossRef]
17. Shahriar, M.H.; Xiao, Y.; Moriano, P.; Lou, W.; Hou, Y.T. CANShield: Deep Learning-Based Intrusion Detection Framework for Controller Area Networks at the Signal-Level. *IEEE Internet Things J.* **2023**, *10*, 22111–22127. [CrossRef]
18. Tanksale, V. Intrusion detection system for controller area network. *Cybersecurity* **2024**, *7*, 4. [CrossRef]
19. Alfardus, A.; Rawat, D.B. Machine Learning-Based Anomaly Detection for Securing In-Vehicle Networks. *Electronics* **2024**, *13*, 1962. [CrossRef]
20. Kim, T.; Kim, J.; You, I. An Anomaly Detection Method Based on Multiple LSTM-Autoencoder Models for In-Vehicle Network. *Electronics* **2023**, *12*, 3543. [CrossRef]
21. Wang, X.; Xu, Y.; Xu, Y.; Wang, Z.; Wu, Y. Intrusion Detection System for In-Vehicle CAN-FD Bus ID Based on GAN Model. *IEEE Access* **2024**, *12*, 82402–82412. [CrossRef]
22. Kishore, C.R.; Rao, D.C.; Nayak, J.; Behera, H. Intelligent Intrusion Detection Framework for Anomaly-Based CAN Bus Network Using Bidirectional Long Short-Term Memory. *J. Inst. Eng. (India) Ser. B* **2024**, 105, 541–564. [CrossRef]
23. Aldhyani, T.H.; Alkahtani, H. Attacks to automatous vehicles: A deep learning algorithm for cybersecurity. *Sensors* **2022**, *22*, 360. [CrossRef] [PubMed]
24. He, Q.; Meng, X.; Qu, R.; Xi, R. Machine learning-based detection for cyber security attacks on connected and autonomous vehicles. *Mathematics* **2020**, *8*, 1311. [CrossRef]
25. Pawar, Y.S.; Honnavalli, P.; Eswaran, S. Cyber Attack Detection On Self-Driving Cars Using Machine Learning Techniques. In Proceedings of the 2022 IEEE 3rd Global Conference for Advancement in Technology (GCAT), Bangalore, India, 7–9 October 2022; pp. 1–5.
26. Gupta, B.B.; Gaurav, A.; Marín, E.C.; Alhalabi, W. Novel graph-based machine learning technique to secure smart vehicles in intelligent transportation systems. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 8483–8491. [CrossRef]
27. Han, M.L.; Kwak, B.I.; Kim, H.K. Event-triggered interval-based anomaly detection and attack identification methods for an in-vehicle network. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 2941–2956. [CrossRef]
28. Onur, F.; Gönen, S.; Barışkan, M.A.; Kubat, C.; Tunay, M.; Yılmaz, E.N. Machine learning-based identification of cybersecurity threats affecting autonomous vehicle systems. *Comput. Ind. Eng.* **2024**, *190*, 110088. [CrossRef]
29. Alalwany, E.; Mahgoub, I. An Effective Ensemble Learning-Based Real-Time Intrusion Detection Scheme for an In-Vehicle Network. *Electronics* **2024**, *13*, 919. [CrossRef]
30. Anand, M.; Kumar, S.P.; Selvi, M.; SVN, S.K.; Ram, G.D.; Kannan, A. Deep learning model based IDS for detecting cyber attacks in IoT based smart vehicle network. In Proceedings of the 2023 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS), Erode, India, 23–25 March 2023; pp. 281–286.
31. Sharmin, S.; Mansor, H.; Kadir, A.F.A.; Aziz, N.A. Benchmarking Frameworks and Comparative Studies of Controller Area Network (CAN) Intrusion Detection Systems: A Review. *arXiv* **2024**, arXiv:2402.06904 . [CrossRef]
32. Altalbe, A. Enhanced Intrusion Detection in In-Vehicle Networks using Advanced Feature Fusion and Stacking-Enriched Learning. *IEEE Access* **2023**, *12*, 2045–2056. [CrossRef]
33. Ding, D.; Wei, Y.; Cheng, C.; Long, J. Intrusion detection for in-vehicle CAN bus based on lightweight neural network. *J. Circuits Syst. Comput.* **2023**, *32*, 2350110. [CrossRef]
34. Amato, F.; Coppolino, L.; Mercaldo, F.; Moscato, F.; Nardone, R.; Santone, A. Can-bus attack detection with deep learning. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 5081–5090. [CrossRef]
35. Kalkan, S.C.; Sahingoz, O.K. In-vehicle intrusion detection system on controller area network with machine learning models. In Proceedings of the 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kharagpur, India, 1–3 July 2020; pp. 1–6.
36. Pascale, F.; Adinolfi, E.A.; Coppola, S.; Santonicola, E. Cybersecurity in automotive: An intrusion detection system in connected vehicles. *Electronics* **2021**, *10*, 1765. [CrossRef]
37. Xiao, J.; Wu, H.; Li, X. Internet of things meets vehicles: Sheltering in-vehicle network through lightweight machine learning. *Symmetry* **2019**, *11*, 1388. [CrossRef]
38. Cheng, P.; Xu, K.; Li, S.; Han, M. TCAN-IDS: Intrusion detection system for internet of vehicle using temporal convolutional attention network. *Symmetry* **2022**, *14*, 310. [CrossRef]
39. Song, H.M.; Woo, J.; Kim, H.K. In-vehicle network intrusion detection using deep convolutional neural network. *Veh. Commun.* **2020**, *21*, 100198. [CrossRef]
40. Salek, M.S.; Biswas, P.K.; Pollard, J.; Hales, J.; Shen, Z.; Dixit, V.; Chowdhury, M.; Khan, S.M.; Wang, Y. A Hybrid Approach for Intrusion Detection in an In-vehicle Controller Area Network using Classical Convolutional Neural Network and Quantum Restricted Boltzmann Machine. *Authorea Prepr.* **2023**. [CrossRef]
41. Metwaly, A.A.; Elhenawy, I. Sustainable intrusion detection in vehicular controller area networks using machine intelligence paradigm. *Sustain. Mach. Intell. J.* **2023**, *4*, 44104. [CrossRef]
42. Khan, M.H.; Javed, A.R.; Iqbal, Z.; Asim, M.; Awad, A.I. DivaCAN: Detecting in-vehicle intrusion attacks on a controller area network using ensemble learning. *Comput. Secur.* **2024**, *139*, 103712. [CrossRef]

43. Minawi, O.; Whelan, J.; Almehmadi, A.; El-Khatib, K. Machine learning-based intrusion detection system for controller area networks. In Proceedings of the 10th ACM Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications, Alicante, Spain, 16–20 November 2020; pp. 41–47.
44. Kang, M.J.; Kang, J.W. Intrusion detection system using deep neural network for in-vehicle network security. *PLoS ONE* **2016**, *11*, e0155781. [CrossRef] [PubMed]
45. ISO 11898-1:2024. Road Vehicles—Controller Area Network (CAN) Part 1: Data Link Layer and Physical Coding Sublayer. Available online: https://www.iso.org/standard/86384.html (accessed on 6 June 2024).
46. De Rosa, M. CAN Bus Security Analysis: A Fuzzing Approach. Ph.D. Thesis, Politecnico di Torino, Turin, Italy, 2024.
47. Smith, G.M. What Is CAN Bus (Controller Area Network) and How It Compares to Other Vehicle Bus Networks. 2024. Available online: https://dewesoft.com/blog/what-is-can-bus (accessed on 3 March 2024).
48. Controller Area Network (CAN). 2008. Available online: https://www.eecs.umich.edu/courses/eecs461/doc/CAN_notes.pdf (accessed on 4 March 2024).
49. Panda, S.; Panaousis, E.; Loukas, G.; Kentrotis, K. Privacy impact assessment of cyber attacks on connected and autonomous vehicles. In Proceedings of the 18th International Conference on Availability, Reliability and Security, Benevento, Italy, 29 August–1 September 2023; pp. 1–9.
50. Hacking and Countermeasure Research Lab. Available online: https://ocslab.hksecurity.net/Datasets (accessed on 1 March 2024).
51. Vrigazova, B. The proportion for splitting data into training and test set for the bootstrap in classification problems. *Bus. Syst. Res. Int. J. Soc. Adv. Innov. Res. Econ.* **2021**, *12*, 228–242. [CrossRef]
52. El-Sayed, N.; Zhu, H.; Schroeder, B. Learning from failure across multiple clusters: A trace-driven approach to understanding, predicting, and mitigating job terminations. In Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 5–8 June 2017; pp. 1333–1344.
53. Kahloot, K.M.; Ekler, P. Algorithmic splitting: A method for dataset preparation. *IEEE Access* **2021**, *9*, 125229–125237. [CrossRef]
54. Ullah, S.; Khan, M.A.; Ahmad, J.; Jamal, S.S.; e Huma, Z.; Hassan, M.T.; Pitropakis, N.; Arshad; Buchanan, W.J. HDL-IDS: A hybrid deep learning architecture for intrusion detection in the Internet of Vehicles. *Sensors* **2022**, *22*, 1340. [CrossRef] [PubMed]
55. Ahmad, U.; Song, H.; Bilal, A.; Alazab, M.; Jolfaei, A. Securing smart vehicles from relay attacks using machine learning. *J. Supercomput.* **2020**, *76*, 2665–2682. [CrossRef]
56. Aloqaily, M.; Otoum, S.; Al Ridhawi, I.; Jararweh, Y. An intrusion detection system for connected vehicles in smart cities. *Ad Hoc Netw.* **2019**, *90*, 101842. [CrossRef]
57. Zhu, K.; Chen, Z.; Peng, Y.; Zhang, L. Mobile edge assisted literal multi-dimensional anomaly detection of in-vehicle network using LSTM. *IEEE Trans. Veh. Technol.* **2019**, *68*, 4275–4284. [CrossRef]
58. Seo, E.; Song, H.M.; Kim, H.K. GIDS: GAN based intrusion detection system for in-vehicle network. In Proceedings of the 2018 16th Annual Conference on Privacy, Security and Trust (PST), Belfast, Ireland, 28–30 August 2018; pp. 1–6.
59. Loukas, G.; Vuong, T.; Heartfield, R.; Sakellari, G.; Yoon, Y.; Gan, D. Cloud-based cyber-physical intrusion detection for vehicles using deep learning. *IEEE Access* **2017**, *6*, 3491–3508. [CrossRef]
60. Yang, Y.; Duan, Z.; Tehranipoor, M. Identify a spoofing attack on an in-vehicle CAN bus based on the deep features of an ECU fingerprint signal. *Smart Cities* **2020**, *3*, 17–30. [CrossRef]
61. Mehedi, S.T.; Anwar, A.; Rahman, Z.; Ahmed, K. Deep transfer learning based intrusion detection system for electric vehicular networks. *Sensors* **2021**, *21*, 4736. [CrossRef] [PubMed]
62. Ma, H.; Cao, J.; Mi, B.; Huang, D.; Liu, Y.; Li, S. A GRU-based lightweight system for CAN intrusion detection in real time. *Secur. Commun. Netw.* **2022**, *2022*, 5827056. [CrossRef]