informatics

MDPI

Article

Supervised Machine Learning for Real-Time Intrusion Attack Detection in Connected and Autonomous Vehicles: A Security Paradigm Shift

Ahmad Aloqaily [1], Emad E. Abdallah [1,*], Hiba AbuZaid [1,*], Alaa E. Abdallah [2] and Malak Al-hassan [3]

[1] Department of Information Technology, Faculty of Prince Al-Hussein Bin Abdullah II for Information Technology, The Hashemite University, P.O. Box 330127, Zarqa 13133, Jordan; aloqaily@hu.edu.jo
[2] Department of Computer Science, Faculty of Prince Al-Hussein Bin Abdullah II for Information Technology, The Hashemite University, P.O. Box 330127, Zarqa 13133, Jordan; aabdallah@hu.edu.jo
[3] Department of Information Technology, King Abdullah II School of Information Technology, The University of Jordan, Amman 11942, Jordan; m_alhassan@ju.edu.jo
* Correspondence: emad@hu.edu.jo (E.E.A.); 2070605@std.hu.edu.jo (H.A.)

Abstract: Recent improvements in self-driving and connected cars promise to enhance traffic safety by reducing risks and accidents. However, security concerns limit their acceptance. These vehicles, interconnected with infrastructure and other cars, are vulnerable to cyberattacks, which could lead to severe costs, including physical injury or death. In this article, we propose a framework for an intrusion detection system to protect internal vehicle communications from potential attacks and ensure secure sent/transferred data. In the proposed system, real auto-network datasets with Spoofing, DoS, and Fuzzy attacks are used. To accurately distinguish between benign and malicious messages, this study employed seven distinct supervised machine-learning algorithms for data classification. The selected algorithms encompassed Decision Trees, Random Forests, Naive Bayes, Logistic Regression, XG Boost, LightGBM, and Multi-layer Perceptrons. The proposed detection system performed well on large real-car hacking datasets. We achieved high accuracy in identifying diverse electronic intrusions across the complex internal networks of connected and autonomous vehicles. Random Forest and LightGBM outperformed the other algorithms examined. Random Forest outperformed the other algorithms in the merged dataset trial, with 99.9% accuracy and the lowest computing cost. The LightGBM algorithm, on the other hand, performed admirably in the domain of binary classification, obtaining the same remarkable 99.9% accuracy with no computing overhead.

Keywords: autonomous vehicles; vehicle security; intrusion detection; connected vehicles; cyber security; machine-learning; smart city

Academic Editors: Augusto Neto and Roger Immich

Received: 25 October 2024
Revised: 25 December 2024
Accepted: 27 December 2024
Published: 6 January 2025

Citation: Aloqaily, A.; Abdallah, E.E.; AbuZaid, H.; Abdallah, A.E.; Al-hassan, M. Supervised Machine Learning for Real-Time Intrusion Attack Detection in Connected and Autonomous Vehicles: A Security Paradigm Shift. *Informatics* **2025**, *12*, 4. https://doi.org/10.3390/informatics12010004

1. Introduction

Traditional mechanical vehicles helped people to move more quickly and efficiently in earlier decades, but as cities and their infrastructures continue to grow and develop, the concept of a smart city has emerged. This concept features interconnected smart infrastructure that collects and shares data among various services to enable intelligent and appropriate operational decisions [1]. New vehicles have appeared (connected and autonomous vehicles) that provide great comfort for passengers, as these vehicles can make the best driving decisions because the vehicle understands its surroundings by analyzing data extracted from sensors and in addition to data collected from the other vehicles connected to it wirelessly [2,3].

Recent advances in technology, such as autonomous and connected vehicles, promise to significantly improve road safety by reducing the probability of accidents and the hazards that go along with them. However, there are some challenges to adopting these cars because of the security of their communication networks. The connected and autonomous vehicles (CAVs) reliance on extensive interconnectivity with infrastructure and other vehicles renders them susceptible to malicious cyberattacks. Such attacks, if successful, could compromise critical control systems or manipulate sensor data, potentially resulting in significant physical harm, including fatalities. This highlights the imperative need for robust cyber defense mechanisms to mitigate these emerging threats.

To achieve their primary objectives of minimizing human error, reducing traffic accidents, and optimizing resource utilization both internally and externally, modern vehicles are equipped with comprehensive communication capabilities. To satisfy these requirements, vehicles must incorporate a variety of electronic control units (ECUs) in charge of managing various vehicle functions, a variety of sensors, and sophisticated internal and external communication systems [4,5].

As part of their internal communication, electronic control units in modern vehicles use several well-established standards and technologies, such as FlexRay, Media Oriented Stream Transfer (MOST), Local Interconnect Network (LIN), Controller Area Network (CAN), and Local Interconnect Network [6,7]. Due to its real-time capabilities and affordability, the CAN stands out among them and is a chosen option for reducing internal communication between electronic control units. It is important to know that the initial CAN design gave minimal thought to security issues and did not guarantee things like accessibility, secrecy, authenticity, non-repudiation, or integrity. Modern vehicles are now vulnerable to cyberattacks as a result of CAN-bus vulnerabilities.

CAVs automobiles, on the other hand, rely on Vehicle Ad Hoc Networks (VANETs) for external communication with other vehicles or infrastructure. According to [8], VANETs enable the sharing of data, such as control information, emergency messages linked to brakes, accidents, and emergency notifications among drivers. The threat to the VANTS external communication system has increased for a number of reasons, including the increasing number of connected vehicles on the road, the availability of an open radio communication medium, speed, and a dynamic topology [9], which has given rise to numerous attacks like floods, black holes, and wormholes.

Vehicle attacks are categorized as: 1. Target the CAN-bus or another internal vehicle control system. 2. Target the communication system of the vehicle, such as Vehicle-to-Vehicle or Vehicle-to-Infrastructure communication. 3. Target the back-end systems, such as self-driving technology or updating vehicle software [10].

Recent studies have emphasized the importance of the CAN protocol as part of vehicle security. The requirement to constantly monitor the CAN-bus message to distinguish between typical and abnormal behavior faults, as well as the rise in attack volume, all contributed to an increase in research into possible defenses for vehicles. Today, there are two approaches to safeguarding the security of modern and autonomous vehicles. First, cryptography is a passive technique that secures the vehicle using algorithms for encryption and decoding that use a lot of resources and computing time. Second, the intrusion detection system (IDS) prioritizes real-time threat detection through a lightweight and active architecture. IDS is driven by the recognition that attackers constantly evolve their strategies, exploiting novel vulnerabilities. Therefore, the IDS focuses on rapidly identifying anomalous activities rather than attempting to pre-emptively defend against every conceivable attack vector [11].

In this article, we propose a different framework for in-vehicle communication security through the deployment of an IDS specifically designed for the CAN-bus. The framework

focuses on detecting unusual and anomalous patterns in CAN-bus data, thus modifying the risk of malicious intrusions and safeguarding connected vehicle communication. In our proposed framework, we used a real-world dataset, advanced preprocessing techniques, and integrated multiple machine learning algorithms to achieve higher accuracy.

The remainder of this paper is organized as follows: Section 2 provides an overview of connected and autonomous vehicles, including their definitions, development, impact on transportation systems, communication systems, and architectural design. Section 3 presents related work. Section 4 outlines the proposed methodology. Section 5 presents the experimental results and analysis. Finally, Section 6 concludes the paper and suggests future directions for research.

2. Background

A CAV is a vehicle that has wireless communication capabilities and can have some or all of its operations automated. Unlike connected vehicles, which maintain the driver's ability to drive while allowing him to communicate to learn about his surroundings and recent traffic news to improve driving and increase safety, autonomous vehicles can carry out driving tasks independently without the driver's intervention based on data collected through sensors to move safely [12].

2.1. CAV's Development History

General Motors began developing the first self-driving car in the early 1950s, bringing the idea of automated cars into the modern day. A design for an autonomous vehicle was presented in 1986 by Ernst Dickens of the University of Munich; in 1987, this truck reached a speed of 60 km/h.

The first self-driving automobile competition was sponsored by the Defense Advanced Research Project Agency (DARPA) in 2004, and Carnegie Mellon University won with a vehicle that covered just seven miles. The challenge was repeated the following year, and Stanford University succeeded with a 140-mile-per-hour vehicle. In the third episode, the challenge was completed on jammed roads; in 2007, Google started looking into making its own self-driving cars; by 2010, it was able to travel 140,000 miles; and in 2014, Tesla vehicles with intelligent cruise control went on sale; it is anticipated that all modes of transportation will be driverless by 2030 [13]. The evolution of linked and autonomous vehicles is shown in Figure 1.

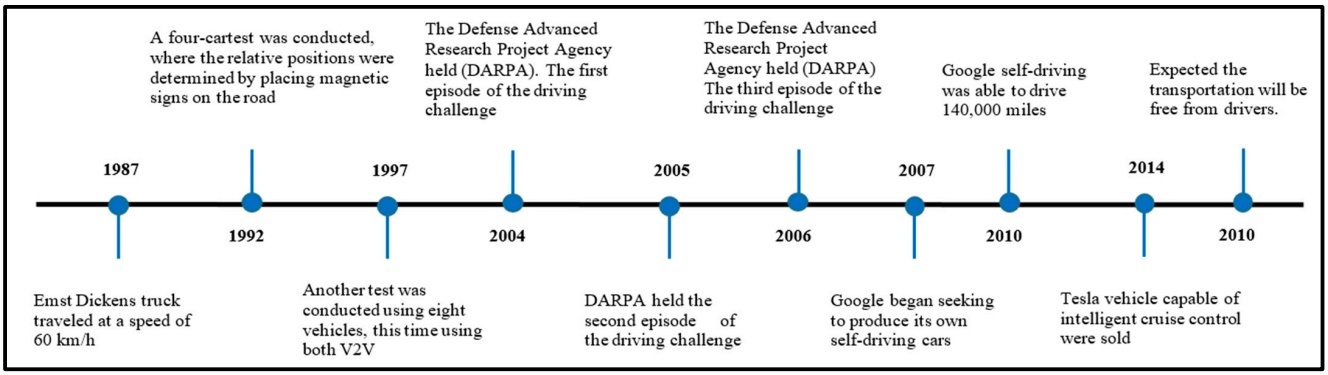


Figure 1. The evolutionary path of connected and autonomous vehicles.

2.2. Architecture Design of CAV

Mechanical parts are no longer the only kind of vehicle components. More than 100 programmed electronic control units have been added to CAV that use sensor data to carry out various tasks. The CAN-bus is used to connect the electronic control units [14,15].

Figure 2 shows that the CAV hardware system consists of sensors, actuators, and V2X (which means that the vehicle is wirelessly connected to everything, such as (Vehicle-to-Vehicle (V2V), Vehicle-to-Pedestrian (V2P), and Vehicle-to-Infrastructure (V2I)). Additionally, the software system in CAV consists of perception, planning, and control [16].

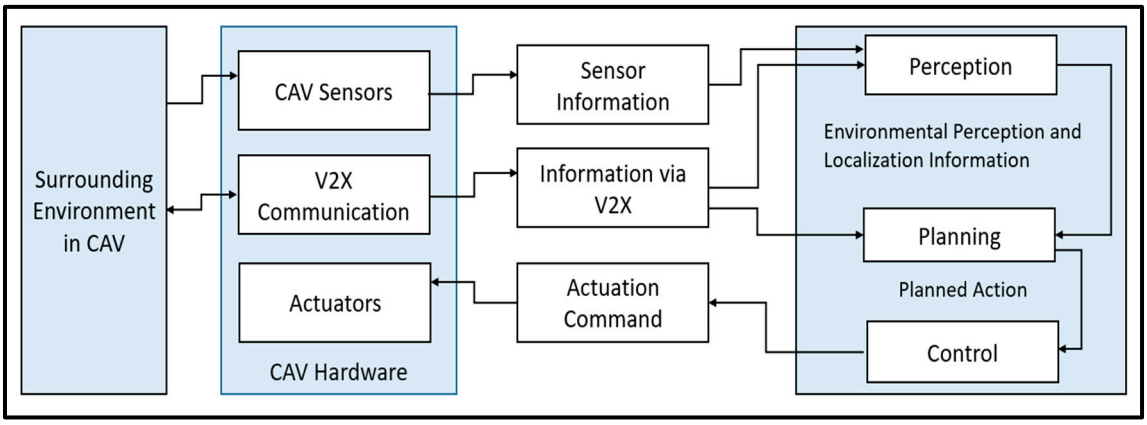


Figure 2. CAV system architecture.

Vehicular Ad Hoc Network enables vehicles to obtain additional information to achieve safety on the road by connecting the vehicle to the following:

- Vehicle-to-Vehicle (V2V) communication: vehicles communicate wirelessly with one another to exchange safety alerts such as lane change warnings, forward collision warnings, left turn warnings, do not pass warnings, and crash alerts [17].
- Vehicle-to-Infrastructure (V2I): means the vehicle's connection to the surrounding environment for the purpose of obtaining information, such as traffic lights, speed limit signs, and stop signs [18].

2.3. Controller Area Network Bus (CAN)

Before the CAN protocol in vehicles, electronic control units were wired to each other. This technology was not scalable because designers had to rewire it each time a new control was installed. The first successful CAN communication protocol inside the vehicle was in 1983, which connects electronic control units simply, at a lower cost, and scalable so that when a new electronic control component is added, no changes to the network infrastructure or any other node are made [19].

The essential features that distinguish the CAN protocol and make it the finest automotive standard include: 1. Broadcast channel: Because the message does not contain the destination address, CAN-bus messages are broadcast to the whole network. All nodes read and validate the message using the CAN ID to determine whether it should be processed [20]. 2. Priority access to the bus: Any node can send and receive messages over the CAN-bus; if there is no activity, each message contains an identifier to prioritize transmission and avoid collision during transmission; the message with the highest CAN ID is given the lowest priority for transmission. 3. Robustness: The CAN protocol has a means for checking mistakes at several levels, such as the message or bit level, as well as preventing repeated errors. If any node exceeds this limit, it is regarded as a failed node, and the connection is severed, as described by CAN-bus as a strong protocol.

The CAN protocol is considered the best for automobiles due to its qualities, such as speed, cost-savings, and real-time communication [21]. It was designed, however, for an isolated setting in which the vehicles were not connected to the outside world. The vehicle is now internally and externally connected, allowing for the inclusion of additional features and greater vehicle efficiency. The external link, on the other hand, permitted attackers to

acquire entire control of the car and compromise the lives of passengers; thus, the vehicle must be secured.

CAN-bus messages (frames), according to Purohit and Govindarasu [22], are sent between electronic control units to communicate with one another. These frames are divided into four categories: (1) Data frame: This is the most common and is used to transport data between electrical control units (ECUs). (2) Remote frame: It is used to request data from another ECU and carries no data other than the request. (3) Error frame: This frame is sent by the electronic control unit when an error in the sent frame is detected. (4) Overload frame: The frame transmitted by the ECU informs all other ECUs that the ECU is busy and cannot receive additional data.

When CAN was first utilized in vehicles, there were not many ECUs, so 11 bits for CAN ID (Standard format) were sufficient. However, as the number of ECUs increased, the expanded format was developed by adding extra bits. Figure 3 illustrates two types of data frame formats: standard and extended [23].

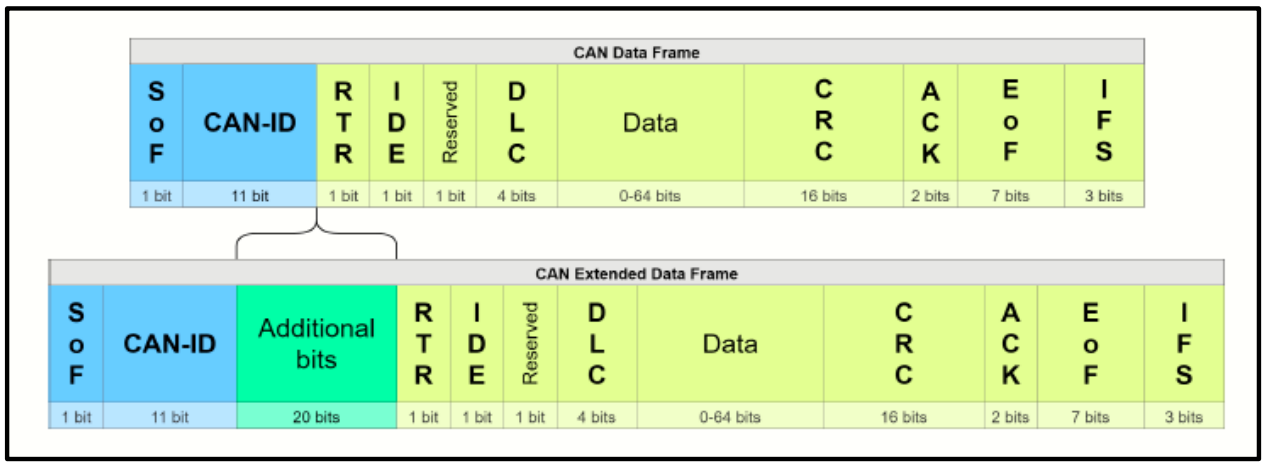


Figure 3. CAN data frame format.

2.4. CAN Weakness

The most significant shortcomings in the CAN protocol are:

1. The Absence of: Because CAN communications are quick and easy, messages are broadcast to all electronic control units without encryption. This allows hackers to breach the confidentiality of messages they send if they get access to the CAN-bus.
2. Insufficient Authentication: The CAN protocol fails authentication, making it impossible to identify the message sender. The non-repudiation principle is breached, making it easy for an attacker to pretend to be any controller and send criminal messages.
3. Poor Data Integrity Checks: Failure to conduct data integrity checks.
4. Absence of Availability: To avoid collisions, the CAN-bus prioritizes messages with shorter identifiers. By attacking this vulnerability, attackers can send a message with the lowest identifier, stopping the execution of other messages and launching a Denial of Service (DoS) attack [24].

The hacker can gain unauthorized access to the CAN-bus via wired or wireless access points [25–28]. Once hackers have reached the CAN's edge, they can launch a variety of attacks including Denial of Service (DoS), Replay attack, Spoofing attack, Fuzzy attack, Target identifier attack, and Malware.

2.5. CAN-Bus Countermeasure

Experts have created measures to protect against different risks [29]. Preventive methods aim to prevent an attack before it occurs by simulating an attack on the present system and then implementing defenses against the discovered attacks. Preventive efforts include encryption and network changes. Evaluation of the system is needed for preventative measures, which take more time and effort and, therefore, conflict with CAN (a real-time protocol).

On the other hand, the most common reactive defense is an intrusion detection system (IDS), the most effective approach to detect attacks and monitor the system for any unusual activity. When anomalies are detected, the IDS immediately tells the user, allowing them to take immediate action to defend the system from any risks [6,30]. Because the intrusion detection system is in charge of monitoring all CAN operations, it must be placed in strategically important areas. The anomaly-based detection technique is the most commonly used among these methods in the automotive setting [10].

2.6. Machin-Learning (ML) Techniques-Based IDS

Vehicles have hundreds of sensors that collect a great quantity of data, as well as data from other cars and infrastructure. Machine-learning-based intrusion detection is the most common because it can evaluate a huge amount of vehicle data to distinguish between normal and unusual behavior, as well as offer a model for attack detection and prediction. An intrusion detection system powered by machine learning can detect previously unknown threats [31,32]. Machine learning is classified into two types: supervised and unsupervised. Supervised learning differs from unsupervised learning in that labeled data must be trained for detection and prediction, whereas unsupervised learning does not. This article focuses on supervised machine learning algorithms for intrusion detection systems.

3. Related Work

This section summarizes the most important intrusion attack detection and car safety procedures proposed in previous studies. The solution categories include cryptography systems and machine-learning-based intrusion detection systems. The ML intrusion detection system has three detection methods: signature-based, anomaly-based, and specification-based. Among them, the anomaly-based technique was the most popular and widely used to ensure vehicle security.

Vehicles have hundreds of sensors that produce a large amount of data, as well as data from other vehicles and infrastructure. ML intrusion detection is the most popular method because it can analyze a large amount of vehicle data to detect normal vs. abnormal behavior and then provide a model for attack detection and prediction. A machine learning intrusion detection system can even identify unidentified attacks [31].

To illustrate the improvements in this area, Abdallah, Aloqaily, and Fayez [10] conducted a comprehensive survey of various machine learning algorithms related to intrusion detection systems. Their work highlights how machine learning can enhance the security of future connected and autonomous vehicles by offering a detailed taxonomy of ML intrusion detection systems. The authors demonstrate that the classification performance of supervised learning algorithms is robust and capable, based on their review of four well-known datasets in this field. Table 1 presents a taxonomy of IDS for connected and autonomous vehicles based on supervised machine-learning methods. We present the dataset used, the highest-performing algorithm, and the best reported accuracy.

Table 1. A taxonomy of IDS for CAV.

Paper	Attack Detection	Dataset	Algorithms	Best Learning Method	Accuracy	
Kalkan and Sahingoz [33]	DoS, Fuzzy, Rpm spoofing, Gear spoofing	Car Hacking Dataset v1	Random Forest, Bagging Tree, Adaptive Boosting, and Neural Network	Random Forest, Bagging Tree	97%	
Song, et al. [34]	D0S, Fuzzy, Rpm spoofing, Gear spoofing	Car Hacking Dataset v1	Convolutional Neural Network (CNN)	Convolutional Neural Network (CNN)	99.93	
Lin, et al. [35]	DoS, Fuzzy, Rpm spoofing, Gear spoofing	Car Hacking Dataset v1	VGG16, XBOOST	XBOOST	99.7%	
Alshammari, et al. [36]	DoS, Fuzzy	CAN Intrusion Dataset.	KNN, SVM	KNN	DoS	97%
					Fuzzy	99%
Ahmed, et al. [37]	DoS, Fuzzy	Can Intrusion (OTID) Dataset	KNN, SVM, RF, VGG16	VGG16	DoS	95.5%
					Fuzzy	95.5%
Moulahi, et al. [38]	DoS, Fuzzy, Impersonation	Can Intrusion (OTID) Dataset	SVM, DT, MLP, And RF	RF	98.5%	
Hossain, et al. [39]	Flooding, Fuzzy, Malfunction	Survival Analysis Dataset	LSTM	LSTM	Flooding	100%
					Fuzzy	99.9%
					Malfunction	100%
Basavaraj and Tayeb [40]	DoS, Fuzzy, Impersonation Replay, Suspension	Automotive Controller Area Network Intrusion Dataset v2	DNN	DNN	98.67%	
He, et al. [41]	DoS, R2L, U2R, PROPE	CAV KDD	Decision Tree, Naive Bayes	Decision Tree	99.80%	
Anthony, Cyn [42]	DoS, Fuzzy	NSL-KDD	K-NN, naïve Bayes, and Logistic egression	K-NN	98.8%	
Onur, Furkan [43]	Deauth Attack, DoS, DDoS	Network traffic	Random Forest	Random Forest	96.1	

Recently, there has been increasing discussion regarding using supervised ML techniques to automate the intrusion detection process. In [33], they utilized the automobile hacking dataset. The idea is separated into two models. The initial model integrated the sub-datasets (DoS, Fuzzy, Spoofing RPM, and Spoofing gear) into a single database with 16.5 million instances. The second model handles preparing data by training each database separately and then extracting results. The authors employed Logistic Regression, Naive Bayes, adaptive boosting, Random Forest, Bagging Tree, and artificial neural network techniques. The Decision Tree achieved the highest classification results in the combined datasets model, with an accuracy of 97%.

Alshammari in [36] employed two datasets in opposite directions. Because the two data sets have similar structures, the same processing was applied which included adding right headers and removing redundant columns. The authors utilized Support Vector Machine and K-Nearest Neighbor. K-Nearest Neighbor achieved the highest accuracy of 93.5%, while Support Vector Machine achieved 93.3% accuracy.

Convolutional Neural Networks (CNNs) [44] to identify Denial of Service and spoofing risks are proposed in [34]. The model was evaluated only offline. The idea depends on receiving CAN data with no additional preparation. However, the model is unsuitable with previous vehicles because it is difficult to utilize via the internet.

The CAR hacking dataset is used in [45] to detect many kinds of attacks including DoS attacks, Fuzzy attacks, and Spoofing attacks. Each group of the dataset was classified

using KNN, Random Forest, SVM, and Multi-layer Perceptron methods. Similarly, ref. [46] presented a classification algorithm based on Neural Networks and Multi-layer Perceptron on the same dataset. Their results showed that Multi-layer Perceptron achieved higher results with 93.3, especially when more than three hidden layers were included in the model.

An Intelligent IDS for IoV systems that uses a modified CNN with hyperparameter optimization methodologies to improve intrusion detection and classification of malicious AVs is proposed in [47]. The framework operates in a 5G Vehicle-to-Everything. The testing findings show that the suggested IIDS is 98% accurate in identifying attacks.

In summary, the evaluated papers show an important activity toward using supervised ML techniques to improve intrusion detection systems in automotive networks. Researchers' use of several kinds of approaches and datasets indicates an increasing awareness of the important need for strong cybersecurity protections in connected and autonomous vehicles. We note that the techniques achieving higher accuracy rates were primarily evaluated using binary classification rather than multiclass classification. Binary classification is inherently simpler as it involves distinguishing between only two categories, whereas multiclass classification, used in our study, is more complex and challenging due to the need to differentiate between multiple attack types.

The positive results obtained, especially with algorithms like Decision Trees, K-Nearest Neighbor, and Multilayer Perceptron, demonstrate the ability of these approaches to successfully categorize attack. However, difficulties remain, especially with real-time integration and model adaption for existing vehicle systems. As the subject grows, additional studies will be required to develop these strategies and ensure their usefulness in protecting automotive networks from potential threats.

4. Methodology

As previously stated, the fundamental goal of this article is to develop a thorough behavioral model for the CAN-bus. This model will use the ability of supervised machine learning methods to distinguish between normal data frames and those with anomalies or irregular behavior. This study attempts to improve our understanding of CAN-bus operations while also increasing its security by enabling the automatic identification of anomalous data frames. Figure 4 shows the proposed methodology that is used to detect internal attacks on connected and self-driving vehicles. The Data Collection phase focuses on gathering CAN traffic data, which is necessary for accurate model training. The chosen dataset obtained from real vehicles includes four sub-datasets representing various attack scenarios: DoS, Fuzzy, Gear spoofing, and RPM spoofing. These sub-datasets provide illustrative samples of both normal and malicious behavior to ensure the model can effectively generalize and detect anomalies. In the Data Preprocessing stage, the raw dataset went through several changes, including missing value removal, hexadecimal-to-decimal conversion, feature selection, and normalization, to make it suitable for machine learning. This logical preparation ensures consistency across all subsets and improves model efficiency. Finally, the Performance Evaluation phase involves testing seven machine learning algorithms to classify CAN messages as normal or attacked. The evaluation is carried out in three steps: individual analysis of each subset, binary classification of merged datasets, and multi-classification with labeled attacks to determine the most effective method for anomaly detection.

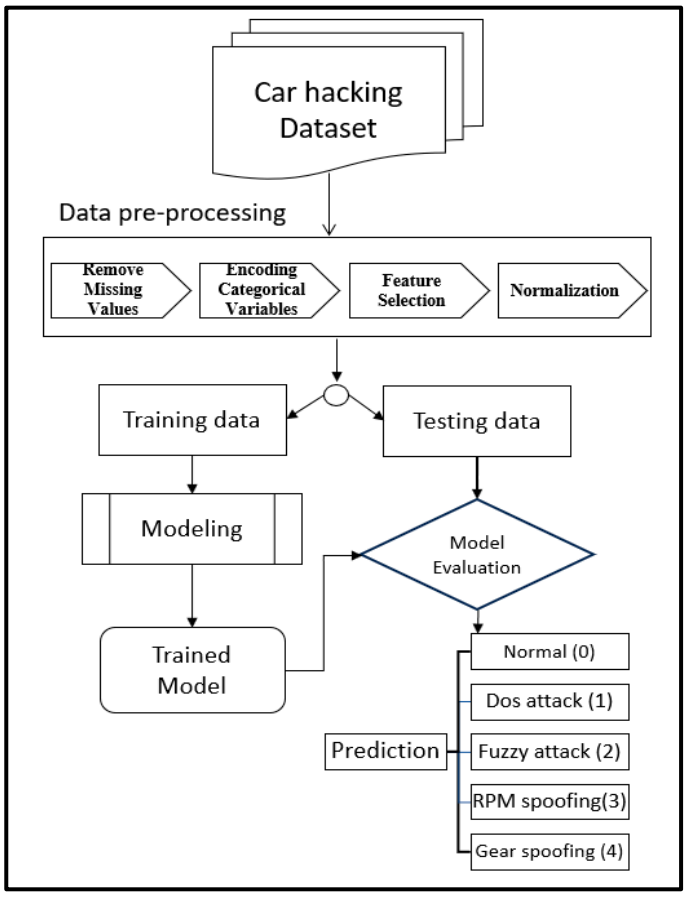


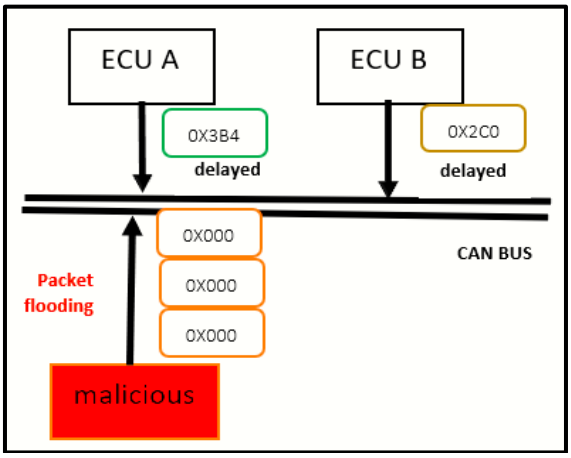
Figure 4. The proposed methodology for detecting internal attacks on connected and self-driving vehicles.

4.1. Data Collection

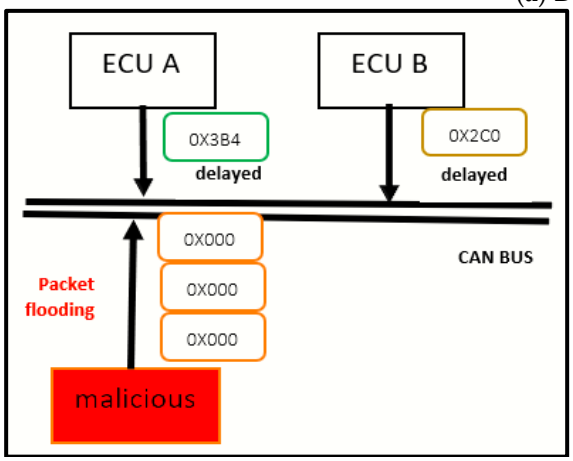
Because the proposed model's performance is based on the quality of the data on which it is trained, the dataset selection stage is crucial. CAN Traffic datasets are composed of collected packets and a collection of data organized around the packets. The dataset was chosen using criteria established by Ring, et al. [48], which are as follows:

1. Make use of a dataset more representative of current attacks, as attack types are always changing.
2. Choosing a dataset with a wide range of assaults so that the model can train more effectively and enhance its capacity to detect new, undiscovered attacks.
3. Selecting a dataset with real-world typical behavior (rather than simulated behavior) to reduce false alarms.
4. To improve the training model's efficiency, the dataset size should be large.

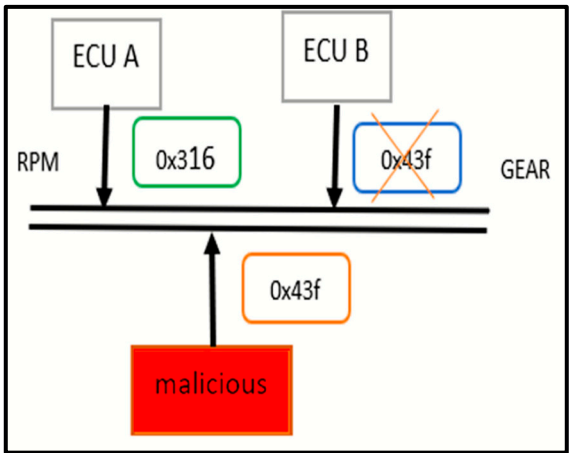
The car hacking dataset by the Hacking and Countermeasures Research Laboratory (HCRL) is used in our experiments [49], which covers four types of genuine attacks on the YF Sonata and is broken into four independent sub-datasets: the DoS, Fuzzy, Gear, and RPM datasets. Figure 5 depicts three attack scenarios assumed in the car hacking dataset. Data were collected from an actual automobile via the OBD-II connector. The experiments carried out during the investigation were divided into three main sections. The first section concentrated on applying classification methods to each of the different subset datasets. Following that, binary classification methods were used in the second experiment to examine the merged sub-dataset, which included the DoS, Fuzzy, RPM, and Gear datasets. Finally, the third experiment examined the merged dataset with the addition of five distinct labels.



(**a**) DoS attack



(**b**) Fuzzy attack



(**c**) Spoofing attack

Figure 5. Attack scenarios.

The car hacking dataset assumes the following attack scenarios:

1. Denial of Service attack: The CAN protocol employs the CAN ID as a priority for sending to avoid transmission collisions. Attackers exploit this CAN weakness by delivering continuous messages with the lowest ID, which is (0), and thus the highest priority for sending win the bus, preventing the execution of the other ECU message. Consider what would happen if you needed to utilize the brake system and there was a Denial of Service assault; the results would be terrible, especially since the lives of the passengers were at stake.
2. Fuzzy Attack: The Fuzzy attack involves sending messages to the CAN-bus that contain random identifiers and data. This action causes the payload within these messages to be executed, which can have unintended consequences. These unexpected results may cause the vehicle to carry out directives such as door opening or engine shutdown, possibly risking passenger safety.
3. Spoofing attack: A knowledgeable hacker is required to inject CAN with messages containing specific IDs and fake data, impersonate other trustworthy ECUs, and transfer data to CAN.

Table 2 presents the count of normal message records, injection message records, and the total number of message records for each sub-dataset.

Table 2. Summary of sub-datasets in car hacking dataset.

Sub-Datasets	#Normal Message Record	#Injection Message Record	#Total Message Record
DoS dataset	3,078,250	587,521	3,655,771
Fuzzy dataset	3,347,013	491,847	3,838,860
Gear spoofing dataset	3,845,890	594,252	4,443,142
RPM spoofing dataset	3,966,805	654,897	4,621,702

4.2. Data Preprocessing

Prior to using the dataset, it must go through a data preprocessing phase to make it appropriate and compliant for building a machine learning model. The raw data frequently contains missing values and is disorganized. The pre-processing process, which consists of multiple stages, transforms raw data into a comprehensible and appropriate format. We used a dataset with four subsets, each with a homogeneous structure, necessitating the use of identical preprocessing approaches across all subsets. The steps below detail the data preparation methods used:

- Remove Missing Values: In order for machine learning algorithms to forecast effectively, missing data in datasets must be deleted or restored. We identified missing data in the following columns of the used dataset: 'DATA3', 'DATA4', 'DATA5', 'DATA6', 'DATA7', and 'attack type', resulting in a dataset size of (16,368,806) records in our experiment.
- Convert from Hexadecimal to Decimal: We identified hexadecimal format in all columns that were converted to decimal in the used dataset.
- Encoding Categorical Variables: The Categorical Variables in the dataset were converted to numeric format.
- Feature Selection: Other features that used to be considered low-information data and required extra effort and time have been removed. We limited the number of features to 10 by removing [DLC, timestamp] from the dataset in use.
- Data Splitting: After completing the pre-processing stage and making the data suitable and understandable for machine learning models, we partitioned the data set into two sections. In all experiments, the database was split 70% for the training group and 30% for the test group.
- Normalization: This step is performed to improve the model's efficiency without changing the characteristics of the original data. In all experiments, minimum–maximum normalization is used to re-measure the values of the selected features within the range [0.0, 1.0].

4.3. Performance Evaluation

Following the completion of dataset preprocessing, the processed data is used in an extensive assessment of supervised machine learning methods. The primary goal of this assessment is to evaluate the classification performance of these algorithms and their ability to distinguish between two essential message categories: normal and attack communications. To conduct this analysis, we used a diverse range of seven classification methods in our tests. These algorithms encompass a variety of approaches, each with its own set of strengths and qualities. Among the algorithms chosen for evaluation are: Decision Tree, Random Forest, Naive Bayes, Logistic Regression, XG Boost, LightGBM, and Multi-layer Perceptron classifier. We intend to evaluate the performance of these classification methods in the context of CAN message classification.

The novelty of our framework compared to existing works lies in using: 1. Lightweight machine learning models like Random Forest and LightGBM, which are optimized for real-time intrusion detection. These models achieve an exceptional balance of high accuracy

and low computational cost, making them particularly suitable for resource-constrained environments like autonomous vehicles. 2. We have highlighted the significance of time complexity in real-time systems, where quick detection is critical. 3. We ensure data quality and consistency through severe pre-processing steps, feature selection, and normalization. These measures are crucial for achieving reliable and effective intrusion detection results.

5. Experimental Results

In this section, we look at the most successful machine learning algorithms for identifying intrusions, focusing on minimizing test time. We evaluated the suggested methodology on a well-known car-hacking dataset [49] in the domain. Our evaluation used important assessment criteria such as accuracy, precision, recall, F-measure, and false positive rate (FPR) to extensively examine model performance and choose the most successful technique.

Python was used to simulate and build the suggested framework, taking advantage of its many libraries and features for machine learning and data preprocessing.

5.1. Experiment 1—Sub-Datasets Experiments

In the first experiment, we employed a series of sub-datasets, including DoS, Fuzzy, RPM, and Gear attacks, to evaluate the performance of various machine learning models. Each sub-dataset was utilized to train the model using seven supervised learning algorithms, allowing for a wide-ranging comparison of their effectiveness.

A comprehensive analysis of these algorithms in terms of several performance indicators. In order to examine the effectiveness of each algorithm in classification, the accuracy, precision, recall, and F-measure of each sub-dataset were compared. Further, we recorded the testing time for each of the models to give us an understanding of the computational complexity of these models. This analysis helped to determine the advantages and disadvantages of each approach to supervised learning in relation to the given sub-datasets and make the right choice in favor of the best model for intrusion detection.

The performance evaluation across different attack datasets (DoS, RPM, and Gear) demonstrates the effectiveness of the proposed algorithms in intrusion detection. All algorithms achieved perfect scores in key metrics, including F1-measure, recall, precision, and accuracy, while maintaining a false positive rate (FPR) of 0%. Among the tested models, the Decision Tree algorithm consistently exhibited the fastest testing times, making it the most time-efficient across all datasets (62.5 µs for DoS, 78.1 µs for RPM, and 78.1 µs for Gear).

Other models, such as Random Forest, Naive Bayes, and Logistic Regression, also performed very well in terms of accuracy but required longer testing times compared to Decision Tree. Advanced algorithms like XG Boost and Multi-layer Perceptron (MLP) provided similar accuracy levels but had significantly higher testing times, with MLP showing the highest computational demand (up to 9690 µs for the Gear dataset).

These results highlight the Decision Tree's balance of accuracy and computational efficiency, making it highly suitable for real-time intrusion detection in resource-constrained environments. Table 3 provides the Performance Evaluation for the Fuzzy dataset experiment, which is the only dataset among the experiments that did not achieve perfect accuracy.

The proposed approach achieves high performance with 100% accuracy for the majority of the algorithms. This indicates that the model can easily learn from a dataset using a single attack, but given the importance of time in vehicle message classification, the test time may be the cut-off for selecting the best model. The high accuracy and recall values can be attributed to the nature of the experiment, which involves a simple binary classification task. This task is less complex than multi-class classification, as it only distinguishes between normal and attack messages.

Table 3. Performance evaluation for the first experiment—Fuzzy dataset.

Fuzzy Dataset	Classification Algorithm	Accuracy	Precision	Recall	F1-Measure	FPR	Testing Time (µS)
	Decision Tree	**99.9**	**99.9**	**99.9**	**99.9**	**0**	**78.1**
	Random Forest	99.9	99.9	99.9	99.9	0	844
	Naive Bayes	94.8	96	94.8	95.1	0.051	375
	Logistic Regression	98.6	98.6	98.6	98.6	0.09	62.5
	XG Boost	99.8	99.8	99.8	99.8	0.008	1110
	Multi-layer Perceptron	99.9	99.9	99.9	99.9	0	6190

Although the accuracy of the DoS, RPM, and Gear dataset classification models is comparable, the Decision Tree algorithm is thought to be the best because it has the quickest testing time. With the fuzzy dataset, classification was more challenging because the dataset differs from others in that the attack uses random packets rather than specified values, making detection more confusing. Additionally, we observe that while each of the three algorithms, Decision Tree, MLP, and Random Forest accuracy is 99.9%, Decision Tree is thought to be the best because it requires the least amount of testing time (78.1 s).

5.2. Experiment 2—Executing Multiple Types of Attacks

In the second experiment, we increased the complexity by combining different types of attacks into a single dataset. The previously separate sub-datasets were merged to create a more comprehensive dataset, which was then split into a training set for model development and a test set for evaluation. This approach allowed us to assess the model's performance in a more challenging and realistic scenario. In total, there were 16,569,471 records in the merged dataset. The labels "Normal" and "Attack" were used for categorization. The actual distribution of these labels varies based on the dataset used, but in general, most of the data is normal traffic, with only a small percentage representing attack situations. Figure 6 compares the ratio of normal and attack communications. To address the data imbalance issue observed in The performance measures for binary classification are shown in Table 4, and testing times are shown in Table 5. Table 6, future work will consider incorporating and evaluating data augmentation techniques, such as oversampling the minority classes using SMOTE and undersampling the majority class, to achieve a balanced dataset for multi-class classification.

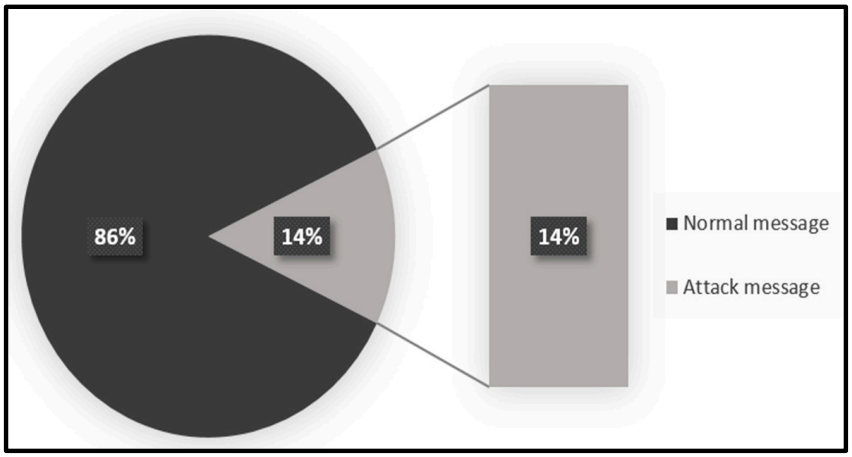


Figure 6. Ratio of normal and attack communications.

Table 4. Binary classification models performance.

Models	Accuracy Score	F1_Score	Recall Score	Precision Score	FPR %	TPR %	FNR %
Decision Tree	99.6	99.6	99.6	99.6	0.0002	97.5	2.4
Random Forest	99.9	99.9	99.9	99.9	0	99.9	0.02
Naive Bayes	92.9	91.9	92.9	93.5	0	50.8	49
XG Boost	99.8	99.8	99.8	99.7	0.33	57.1	42.8
Logistic Regression	93.6	93.8	93.6	92.9	0.0003	99.2	0.78
Multi-layer Perceptron	99.9	99.9	99.9	99.9	0.0004	99.9	0.01
LightGBM	99.9	99.9	99.9	99.9	0	99.9	0.002

Table 5. Testing time for the classification models (second, µs: millisecond).

Models	Decision Tree	Random Forest	Naive Bayes	Logistic Regression	XG Boost	MLP Classifier	LightGBM
Testing time	531 µs	5.64 s	1.3 s	344 µs	5.72 s	6.1 s	11 s
Training time	1 min 41 s	1 min 16 s	5.24 s	4 min 24 s	8 min 12 s	12 min 42 s	1 min 6 s

Table 6. Records distribution of various attack categories.

Message Type	Percentage (%)
Normal message	85.8%
Fuzzy attack	3%
Rpm spoofing attack	4%
Gear spoofing attack	3.6%
DoS attack	3.6%

As shown in Table 4. the proposed algorithms are capable of classifying the attack with high accuracy, with DT, RF, XG Boost, LightGBM, and MLP achieving an accuracy of 99.9%. However, when considering additional factors such as false alarms, training time, and testing time, the LightGBM algorithm is the best, with the lowest FNR, FPR rate, and testing time of 11 s. The Random Forest algorithm closely follows overall performance.

5.3. Experiment 3—Detect Malicious Attacks Using Five Labels

In this experiment, we used the merged datasets to identify malicious attacks, classifying data into five labels: Normal (0), DoS attack (1), Fuzzy attack (2), RPM spoof (3), and Gear spoof (4). Table 6 illustrates the distribution of normal and attack messages within these categories.

The performance metrics for the multi-class classification, derived from the confusion matrix for each class Normal, DoS, Fuzzy, RPM, and Gear are presented in Table 7 Additionally, Table 8 provides an overview of the testing times and accuracy for each classification model.

Table 7. Multi-class classification performance metrics.

Models	Accuracy Score	F1_Score	Recall Score	Precision Score	FPR %	TPR %	FNR %
Decision Tree	99.6	99.6	99.6	99.6	2.5	99.9	0
Random Forest	99.9	99.9	99.9	99.9	0.02	99.9	0.0004
Naive Bayes	85.8	88.8	85.8	95.2	16.3	83.6	0.55
XG Boost	99.9	99.9	99.9	99.9	0.4	99.9	0.002
Logistic Regression	93.7	90.8	93.7	90.9	2.3	99.9	0.04
Multi-layer Perceptron	99.9	99.9	99.9	99.9	0.01	99.9	0.0005
LightGBM	99.9	99.9	99.9	99.9	0	100	0.0002

Table 8. Testing time of classification models (second, µs: millisecond).

Models	Decision Tree	Random Forest	Naive Bayes	Logistic Regression	XG Boost	MLP Classifier	LightGBM
Testing time	594 µs	1.27 s	1.42 s	766 µs	2.3 s	38 s	1 min 32 s
Training time	1 min 37 s	1 min 11 s	8.51 s	7 min 14 s	3 min 55 s	26 min 7 s	5 min 21 s

The results presented in Tables 7 and 8 show that multi-class classification increased testing time. The models faced particular challenges in accurately recognizing Fuzzy attacks. Among the algorithms tested, the LightGBM classification algorithm performed very well, with an accuracy of 99.9% and a false positive rate of 0%, as illustrated in Table 7. While the Decision Tree algorithm had the lowest testing time, it also had a greater false positive rate than LightGBM, showing a trade-off between speed and accuracy. However, the Random Forest algorithm was shown to be the best balanced and successful solution for identifying malicious assaults in multi-class classification overall. It obtained the lowest false negative rate (FNR) and false positive rate (FPR) while keeping the testing time to 1.27 s. Our methodology leveraged lightweight machine learning models such as Random Forest and LightGBM, which inherently balance accuracy and efficiency. Table 9 compares binary and multi-class classification based on accuracy and testing time. Table 9 compares binary and multi-class classification in terms of accuracy and testing time.

Table 9. Compare binary vs. multi-class classification based on accuracy and testing time.

	Models	Accuracy	Testing Time	FPR %	TPR %	FNR %
Decision Tree	Binary classification	99.6	531 µs	0.0002	97.5	2.4
	multiclassification	99.6	594 µs	0	99.9	2.5
Random Forest	Binary classification	99.9	5.64 s	0	99.9	0.048
	multiclassification	99.9	1.27 s	0.02	99.9	0.0004
Naive Bayes	Binary classification	92.9	1.3 s	5.7	50.8	49
	multiclassification	85.8	1.42 s	16.3	83.6	0.55
Logistic Regression	Binary classification	93.6	344 µs	0.33	57.1	42.8
	multiclassification	92.9	766 µs	0.39	99.9	0.002
XG Boost	Binary classification	99.8	6.9 s	0.0003	99.2	0.78
	multiclassification	99.9	11 s	2.3	99.9	0.03
Multi-layer Perceptron	Binary classification	99.9	6.1 s	0.0004	99.9	0.01
	multiclassification	99.9	38 s	0.01	99.9	0.0005
LightGBM	Binary classification	99.9	11.3 s	0	99.9	0.002
	multiclassification	99.9	1 min 32 s	0	100	0.0002

Table 9 compares binary and multi-class classification based on accuracy and testing time. The results clearly indicate that the proposed methodology effectively identifies different types of attacks with performance accuracy and testing time that are very close to those achieved in binary classification. This demonstrates that the methodology maintains high efficiency and precision even when handling more complex, multi-class scenarios.

6. Discussion and Comparative Analysis of Results

The experimental results demonstrate the robustness and efficiency of the proposed machine learning models in identifying vehicular cyberattacks. Across all experiments, models like Decision Tree, Random Forest, XG Boost, Multi-layer Perceptron, and LightGBM performed exceptionally well, achieving high accuracy, precision, recall, and F1 scores across various attack types in the car-hacking dataset.

In the first experiment, where individual attack types (DoS, Fuzzy, RPM, and Gear spoofing) were considered, all algorithms achieved 100% accuracy for most datasets. However, time complexity has become a differentiating factor, particularly for real-time systems where quick detection is crucial. Decision Tree emerged as the most time-efficient algorithm, with the fastest testing times across DoS, RPM, and Gear spoofing datasets. For the Fuzzy attack dataset, classification became more challenging as the random packet nature of the attack increased complexity. While algorithms like Decision Tree, Random Forest, and MLP all achieved 99.9% accuracy for this dataset, Decision Tree was still preferred due to its lower testing time (78.1 μS).

When these results are compared to other studies in the field, they hold up very well. For instance, Kalkan and Sahingoz [33] reported 97% accuracy using Random Forest and Bagging Tree, which is lower than the 99.9% accuracy achieved in this study. While their focus was on a similar car hacking dataset, their models did not perform as efficiently as the LightGBM and Random Forest models used here, particularly in terms of computational complexity.

Song, Woo and Kim [34] achieved 99.93% accuracy using Convolutional Neural Networks (CNN) on the same Car Hacking Dataset v1. Although the CNN model slightly outperformed our 99.9% accuracy, the higher computational costs and longer training times of CNN make our model a more practical choice for real-time vehicular systems.

In the second experiment, which involved a more complex dataset combining multiple types of attacks, the proposed models continued to perform well. LightGBM, in particular, stood out with the lowest false negative rate (FNR) and false positive rate (FPR), as well as a testing time of just 11 s. Although Random Forest was closely followed in terms of accuracy (99.9%) and testing time (1.27 s), LightGBM's superior performance in terms of both speed and precision makes it an ideal candidate for real-time intrusion detection systems. Compared to Lin, Wang, Chao, Lin and Chen [35], who used XG Boost and achieved 99.7% accuracy, our model's 99.9% accuracy with LightGBM is an improvement, demonstrating its capability in handling complex, combined attack datasets.

Other studies, such as Alshammari, Zohdy, Debnath and Corser [36] and Ahmed, Jeon and Ahmad [37], achieved high accuracies (99% for Fuzzy attacks in Alshammari et al. and 95.5% in Ahmed et al.), but these results still fall short of the 99.9% accuracy achieved by our model across multiple attack types. Additionally, our use of LightGBM and Random Forest offers a better balance between accuracy and computational efficiency, outperforming models like KNN and SVM used in these studies.

The third experiment, which involved multi-class classification across five attack types (Normal, DoS, Fuzzy, RPM spoof, and Gear spoof), further emphasized the strengths of the proposed models. Although multi-class classification increased testing time compared to binary classification, the models continued to demonstrate high accuracy, with LightGBM and Random Forest leading the pack. LightGBM achieved 99.9% accuracy with a false positive rate of 0%, outperforming Decision Tree, which, despite its faster testing time, had a slightly higher false positive rate. These findings align with those of Moulahi, Zidi, Alabdulatif and Atiquzzaman [38], who achieved 98.5% accuracy using Random Forest for multi-class classification but were still surpassed by the performance metrics obtained in this study.

Finally, comparisons with studies like Hossain, Inoue, Ochiai, Fall and Kadobayashi [39], who achieved 99.9% accuracy for Fuzzy attacks using LSTM, suggest that while LSTM models are effective, their higher computational complexity makes LightGBM a more practical alternative for real-time systems. Similarly, Basavaraj and Tayeb [40] achieved 98.67% accuracy using Deep Neural Networks (DNNs) for detecting multiple attack types, but this still falls short of the 99.9% accuracy and lower FPR achieved by LightGBM in our study.

7. Conclusions

This study demonstrates the efficacy of an effective intrusion detection algorithm designed to protect internal vehicle communications in connected and autonomous vehicles. Using real-world automotive network datasets with spoofing, DoS, and Fuzzy attacks, we applied seven different supervised machine-learning methods to classify data and distinguish between benign and malicious communications.

Our results show that, although multi-class classification required more testing time, the proposed methodology achieved excellent levels of accuracy and efficiency in complex scenarios. In particular, the Random Forest algorithm emerged as the best balanced, with 99.9% accuracy at minimal computational cost. LightGBM also performed exceptionally well, particularly in binary classification, matching Random Forest's accuracy.

In conclusion, the experimental results from this study consistently align with or exceed the performance metrics reported in the literature. The use of LightGBM and Random Forest proved particularly effective, achieving near-perfect accuracy across multiple attack types while maintaining low false positive and false negative rates. These results suggest that our methodology provides a robust and efficient solution for vehicular intrusion detection, outperforming many of the state-of-the-art models in terms of both accuracy and computational efficiency, especially in real-time environments.

These findings demonstrate the potential of the proposed intrusion detection system to improve the security of self-driving and connected vehicles, addressing the essential need for strong cyberattack defense. As the automotive industry continues to grow, the implementation of advanced detection systems will be crucial to ensuring the safe and secure operation of future vehicles.

Future work will explore unsupervised and hybrid machine-learning methods to detect new and unknown attack types without depending only on labeled datasets. Moreover, increasing the dataset to include additional attack situations and various vehicle models will also be important to improve the model's generalizability. Lastly, we plan to address class imbalance issues by incorporating data augmentation techniques such as oversampling minority classes, particularly for multi-class classification.

Author Contributions: E.E.A. and A.A. contributed to the study's Conception and Design and Project administration. E.E.A., A.A. and H.A. performed Data curation, Investigation, Methodology. H.A., A.E.A. and M.A.-h. Coding, Validation, Resources, Writing—original draft. A.E.A. and M.A.-h. organized the article writing process. All authors participated in writing, reviewing, editing the article and analyzing the results. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

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

References

1. Song, H.; Srinivasan, R.; Sookoor, T.; Jeschke, S. *Smart Cities: Foundations, Principles, and Applications*; John Wiley & Sons: Hoboken, NJ, USA, 2017.
2. Jabbarpour, M.R.; Nabaei, A.; Zarrabi, H. Intelligent guardrails: An iot application for vehicle traffic congestion reduction in smart city. In Proceedings of the 2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Chengdu, China, 15–18 December 2016; pp. 7–13.
3. Yang, Q.; Fu, S.; Wang, H.; Fang, H. Machine-learning-enabled cooperative perception for connected autonomous vehicles: Challenges and opportunities. *IEEE Netw.* **2021**, *35*, 96–101. [CrossRef]
4. Bedretchuk, J.P.; Arribas García, S.; Nogiri Igarashi, T.; Canal, R.; Wedderhoff Spengler, A.; Gracioli, G. Low-cost data acquisition system for automotive electronic control units. *Sensors* **2023**, *23*, 2319. [CrossRef] [PubMed]
5. Cherif, M.O. Optimization of v2v and v2i Communications in an Operated Vehicular Network. Doctoral Dissertation, Université de Technologie de Compiègne, Compiègne, France, 2010.
6. Jo, H.J.; Choi, W. A survey of attacks on controller area networks and corresponding countermeasures. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 6123–6141. [CrossRef]
7. Zurawski, R. *Embedded Systems Handbook 2-Volume Set*; CRC Press: Boca Raton, FL, USA, 2018.
8. Petrov, T.; Pocta, P.; Roman, J.; Buzna, Ľ.; Dado, M. A feasibility study of privacy ensuring emergency vehicle approaching warning system. *Appl. Sci.* **2019**, *10*, 298. [CrossRef]
9. Shah, S.A.A.; Fernando, X.N.; Kashef, R. A Survey on Artificial Intelligence based Internet of Vehicles utilizing Unmanned Aerial Vehicles. *Drones* **2024**, *8*, 353. [CrossRef]
10. Abdallah, E.E.; Aloqaily, A.; Fayez, H. Identifying intrusion attempts on connected and autonomous vehicles: A survey. *Procedia Comput. Sci.* **2023**, *220*, 307–314. [CrossRef]
11. Kleberger, P.; Olovsson, T.; Jonsson, E. Security aspects of the in-vehicle network in the connected car. In Proceedings of the 2011 IEEE intelligent vehicles symposium (IV), Baden-Baden, Germany, 5–9 June 2011; pp. 528–533.
12. Tennant, C.; Stilgoe, J.; Vucevic, S.; Stares, S. Public anticipations of self-driving vehicles in the UK and US. *Mobilities* **2024**, 1–18. [CrossRef]
13. Ghandriz, T.; Jacobson, B.; Laine, L.; Hellgren, J. Impact of automated driving systems on road freight transport and electrified propulsion of heavy vehicles. *Transp. Res. Part C Emerg. Technol.* **2020**, *115*, 102610. [CrossRef]
14. Sarieddine, K.; Sayed, M.A.; Torabi, S.; Atallah, R.; Assi, C. Investigating the security of ev charging mobile applications as an attack surface. *ACM Trans. Cyber-Phys. Syst.* **2023**, *7*, 26. [CrossRef]
15. Jo, K.; Kim, J.; Kim, D.; Jang, C.; Sunwoo, M. Development of autonomous car—Part II: A case study on the implementation of an autonomous driving system based on distributed architecture. *IEEE Trans. Ind. Electron.* **2015**, *62*, 5119–5132. [CrossRef]
16. Rahman, M.; Islam, M.R.; Chowdhury, M.; Khan, T. Development of a connected and automated vehicle longitudinal control model. *arXiv* **2020**, arXiv:2001.00135.
17. Ahangar, M.N.; Ahmed, Q.Z.; Khan, F.A.; Hafeez, M. A survey of autonomous vehicles: Enabling communication technologies and challenges. *Sensors* **2021**, *21*, 706. [CrossRef] [PubMed]
18. Sheikh, M.S.; Liang, J. A comprehensive survey on VANET security services in traffic management system. *Wirel. Commun. Mob. Comput.* **2019**, *2019*, 2423915. [CrossRef]
19. Lokman, S.-F.; Othman, A.T.; Abu-Bakar, M.-H. Intrusion detection system for automotive Controller Area Network (CAN) bus system: A review. *EURASIP J. Wirel. Commun. Netw.* **2019**, *2019*, 184. [CrossRef]
20. Bozdal, M.; Samie, M.; Jennions, I. A survey on can bus protocol: Attacks, challenges, and potential solutions. In Proceedings of the 2018 International Conference on Computing, Electronics & Communications Engineering (iCCECE), Southend, UK, 16–17 August 2018; pp. 201–205.
21. Sato, R.; Fukumoto, S. Response-time analysis for controller area networks with randomly occurring messages. *IEEE Trans. Veh. Technol.* **2020**, *69*, 3893–3902. [CrossRef]
22. Purohit, S.; Govindarasu, M. ML-based anomaly detection for intra-vehicular CAN-bus networks. In Proceedings of the 2022 IEEE International Conference on Cyber Security and Resilience (CSR), Rhodes, Greece, 27–29 July 2022; pp. 233–238.
23. Douss, A.B.C.; Abassi, R.; Sauveron, D. State-of-the-art survey of in-vehicle protocols and automotive Ethernet security and vulnerabilities. *Math. Biosci. Eng.* **2023**, *20*, 17057–17095. [CrossRef]
24. Tariq, S.; Lee, S.; Kim, H.K.; Woo, S.S. CAN-ADF: The controller area network attack detection framework. *Comput. Secur.* **2020**, *94*, 101857. [CrossRef]
25. Oruganti, P.S.; Appel, M.; Ahmed, Q. Hardware-in-loop based automotive embedded systems cybersecurity evaluation testbed. In Proceedings of the ACM Workshop on Automotive Cybersecurity, Richardson, TX, USA, 27 March 2019; pp. 41–44.

26. Cheah, M.; Bryans, J.; Fowler, D.S.; Shaikh, S.A. Threat intelligence for bluetooth-enabled systems with automotive applications: An empirical study. In Proceedings of the 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W), Denver, CO, USA, 26–29 June 2017; pp. 36–43.
27. Hoang, T.-N.; Kim, D. Detecting In-vehicle Intrusion via Semi-supervised Learning-based Convolutional Adversarial Autoencoders. *arXiv* **2022**, arXiv:2204.01193. [CrossRef]
28. Alzghoul, J.R.; Abdallah, E.E.; Al-khawaldeh, A.S. Fraud in Online Classified Ads: Strategies, Risks, and Detection Methods: A Survey. *J. Appl. Secur. Res.* **2024**, *19*, 45–69. [CrossRef]
29. Bnbusayyis, A.; Vaiyapuri, T. Identifying and benchmarking key features for cyber intrusion detection: An ensemble approach. *IEEE Access* **2019**, *7*, 106495–106513. [CrossRef]
30. Studnia, I.; Alata, E.; Nicomette, V.; Kaâniche, M.; Laarouchi, Y. A language-based intrusion detection approach for automotive embedded networks. *Int. J. Embed. Syst.* **2018**, *10*, 1–12. [CrossRef]
31. Limbasiya, T.; Teng, K.Z.; Chattopadhyay, S.; Zhou, J. A systematic survey of attack detection and prevention in connected and autonomous vehicles. *Veh. Commun.* **2022**, *37*, 100515. [CrossRef]
32. Khalil, A.; Farman, H.; Nasralla, M.M.; Jan, B.; Ahmad, J. Artificial Intelligence-based intrusion detection system for V2V communication in vehicular adhoc networks. *Ain Shams Eng. J.* **2024**, *15*, 102616. [CrossRef]
33. 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.
34. Song, H.M.; Woo, J.; Kim, H.K. In-vehicle network intrusion detection using deep convolutional neural network. *Veh. Commun.* **2020**, *21*, 100198. [CrossRef]
35. Lin, H.-C.; Wang, P.; Chao, K.-M.; Lin, W.-H.; Chen, J.-H. Using deep learning networks to identify cyber attacks on intrusion detection for in-vehicle networks. *Electronics* **2022**, *11*, 2180. [CrossRef]
36. Alshammari, A.; Zohdy, M.A.; Debnath, D.; Corser, G. Classification approach for intrusion detection in vehicle systems. *Wirel. Eng. Technol.* **2018**, *9*, 79–94. [CrossRef]
37. Ahmed, I.; Jeon, G.; Ahmad, A. Deep learning-based intrusion detection system for internet of vehicles. *IEEE Consum. Electron. Mag.* **2021**, *12*, 117–123. [CrossRef]
38. 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]
39. Hossain, M.D.; Inoue, H.; Ochiai, H.; Fall, D.; Kadobayashi, Y. LSTM-based intrusion detection system for in-vehicle can bus communications. *IEEE Access* **2020**, *8*, 185489–185502. [CrossRef]
40. Basavaraj, D.; Tayeb, S. Towards a lightweight intrusion detection framework for in-vehicle networks. *J. Sens. Actuator Netw.* **2022**, *11*, 6. [CrossRef]
41. 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]
42. Anthony, C.; Elgenaidi, W.; Rao, M. Intrusion detection system for autonomous vehicles using non-tree based machine learning algorithms. *Electronics* **2024**, *13*, 809. [CrossRef]
43. 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]
44. Otoom, A.F.; Abdallah, E.E. Deep learning for accurate detection of brute force attacks on IOT Networks. *Procedia Comput. Sci.* **2023**, *220*, 291–298. [CrossRef]
45. Alfardus, A.; Rawat, D.B. Intrusion detection system for can bus in-vehicle network based on machine learning algorithms. In Proceedings of the 2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York, NY, USA, 1–4 December 2021; pp. 0944–0949.
46. 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]
47. Anbalagan, S.; Raja, G.; Gurumoorthy, S.; Suresh, R.D.; Dev, K. IIDS: Intelligent intrusion detection system for sustainable development in autonomous vehicles. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 15866–15875. [CrossRef]
48. Ring, M.; Wunderlich, S.; Scheuring, D.; Landes, D.; Hotho, A. A survey of network-based intrusion detection data sets. *Comput. Secur.* **2019**, *86*, 147–167. [CrossRef]
49. 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.