Journal of
Sensor and Actuator Networks

Article

Towards a Lightweight Intrusion Detection Framework for In-Vehicle Networks

Dheeraj Basavaraj and Shahab Tayeb *

Department of Electrical and Computer Engineering, California State University, Fresno, CA 93740, USA; bdheeraj@mail.fresnostate.edu
* Correspondence: tayeb@csufresno.edu

Abstract: With the emergence of networked devices, from the Internet of Things (IoT) nodes and cellular phones to vehicles connected to the Internet, there has been an ever-growing expansion of attack surfaces in the Internet of Vehicles (IoV). In the past decade, there has been a rapid growth in the automotive industry as network-enabled and electronic devices are now integral parts of vehicular ecosystems. These include the development of automobile technologies, namely, Connected and Autonomous Vehicles (CAV) and electric vehicles. Attacks on IoV may lead to malfunctioning of Electronic Control Unit (ECU), brakes, control steering issues, and door lock issues that can be fatal in CAV. To mitigate these risks, there is need for a lightweight model to identify attacks on vehicular systems. In this article, an efficient model of an Intrusion Detection System (IDS) is developed to detect anomalies in the vehicular system. The dataset used in this study is an In-Vehicle Network (IVN) communication protocol, i.e., Control Area Network (CAN) dataset generated in a real-time environment. The model classifies different types of attacks on vehicles into reconnaissance, Denial of Service (DoS), and fuzzing attacks. Experimentation with performance metrics of accuracy, precision, recall, and F-1 score are compared across a variety of classification models. The results demonstrate that the proposed model outperforms other classification models.

Keywords: Controller Area Network; Internet of Vehicles; intrusion detection system; security

Citation: Basavaraj, D.; Tayeb, S. Towards a Lightweight Intrusion Detection Framework for In-Vehicle Networks. *J. Sens. Actuator Netw.* **2022**, *11*, 6. https://doi.org/10.3390/jsan11010006

Academic Editor: Torsten Braun

Received: 27 September 2021
Accepted: 1 December 2021
Published: 10 January 2022

1. Introduction

In the past decades, devices such as cellular phones, vehicles, and home appliances were connected to the Internet due to the advancements in the Internet of things (IoT) [1]. The phenomenal increase in the number of users utilizing IoT devices has made the technology even popular. Due to the popularity and expansion of technology, various types of networked devices led to larger attack surfaces and to an increase in critical attacks [2].

In recent decades, there has been an increase in the development of vehicular technologies such as Connected Vehicles (CVs), Autonomous Vehicles (AVs), and Electric Vehicles (EVs) [3]. The development of vehicular technologies has a significant impact on the reduction of socioeconomic costs. For instance, with the use of EVs, the effect of greenhouse gases emissions can be reduced and mobility services for children, mobile-impaired individuals, and seniors provided. Similarly, CVs and AVs, i.e., Connected Autonomous Vehicles (CAVs), are expected to reduce parking space, traffic congestion, road accidents, and cost of transportation to a large extent [3]. The combination of these three technologies will enhance the services and operations in a smart city [4].

Intelligent Transportation Systems (ITS) provide efficient and reliable transportation with respect to security, and they also enable vehicles to interact with roadside-based units [5].

Figure 1 illustrates the potential of automobiles when technologies are combined. The combination of heterogeneous technologies will further benefit society, i.e., combining AVs and CVs improves technological benefits (as in self-driving vehicles). Additionally, combining EVs and AVs improves the existing features of the smart city. Finally, social lifestyle will be improved (smart mobility assistance) by combining CVs and EVs for physically challenged and elderly citizens [2,6].

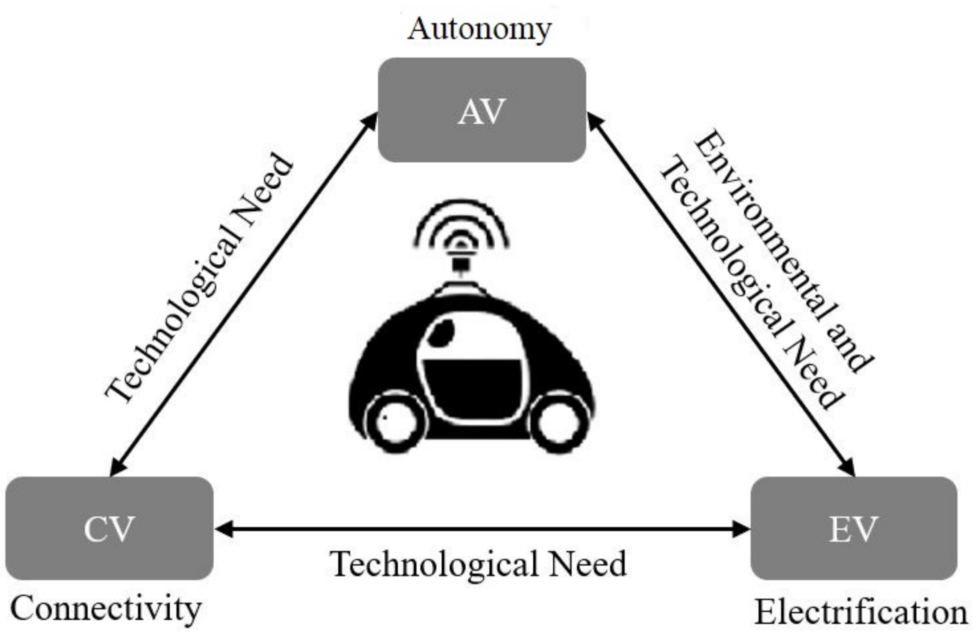


Figure 1. Technologies that demonstrate the potential of automobiles (adapted from [2]).

Although automobile technologies are popular due to their mobility characteristics and communication with infrastructures, these technologies are vulnerable to security issues, and in the case of CVs, they are also affected by communication with other vehicles [7]. The vehicles become smart due to the usage of IoT devices in their design. As per the survey by the Group of Intelligent Transport Systems, only one percent were electronic components in the 1980s, and this has increased to 50 percent currently, compared to previous use of electronic components in the transportation system. [5]. Grouping all electronic components together can be used as a smart module for in-vehicle communication, and this is referred to as Electronic Control Module (ECM) in the vehicular system. Communication between two ECMs is through serial communication protocols, i.e., vehicle bus. This vehicle bus is designed as an internal communication network that interconnects electronic components embedded in vehicles and is referred to as protocols, which are illustrated in Table 1.

Table 1. Classification of the in-vehicle network.

Protocol	Description	Application
CAN	Multi-master, Asynchronous-serialnetwork, Fast and reliable, Max speed 1Mbps	Real-time communication
LIN	Single-master, Multiple-slave serialnetwork, Cheap and slow	Body-control such as Door lock, seat belts
FlexRay	Fast:10Mbps, Costly	Multimedia

In this section, we review the CAN protocol as we are analyzing the attacks in this communication protocol.

1.1. Development of CAN Protocol and Communication Mechanism

Controller Area Network (CAN) was developed in 1985 for in-vehicle networks by Bosch. Traditionally, automobile industries connected electronic devices in the vehicle using a wiring system as illustrated in Figure 2. Using wires across communication units produces a bulky and expensive system. CAN provides a low-level network solution for high-speed in-vehicle communication and defines the datalink and physical layer of Open System Interconnection (OSI). It was designed to reduce wiring costs so that separate ECMs can communicate with single wire pairs. These electronic control units may be suspension, Anti-lock Braking Systems (ABS), air conditioner, instrument panel, seat position, transmission, engine control and batteries as shown in Figure 3.

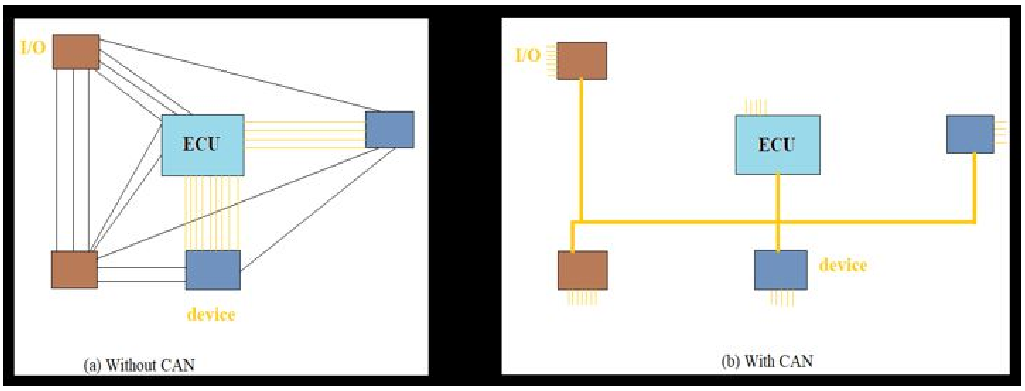


Figure 2. CAN networks significantly reduce wiring.

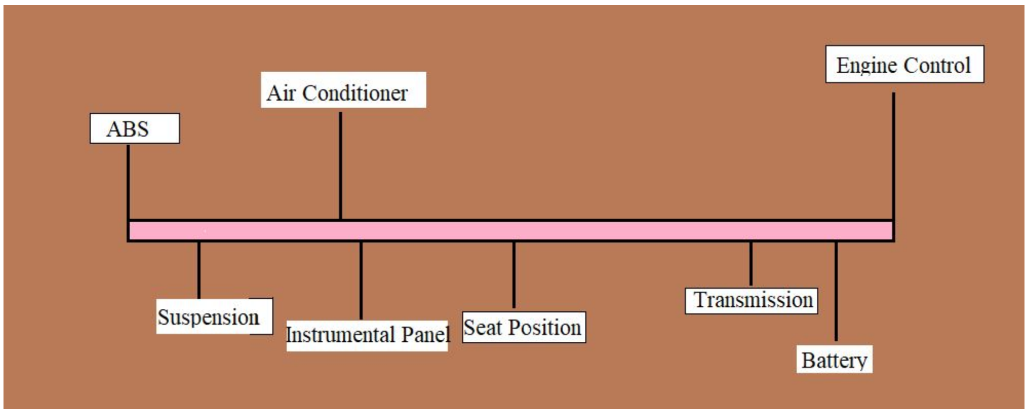
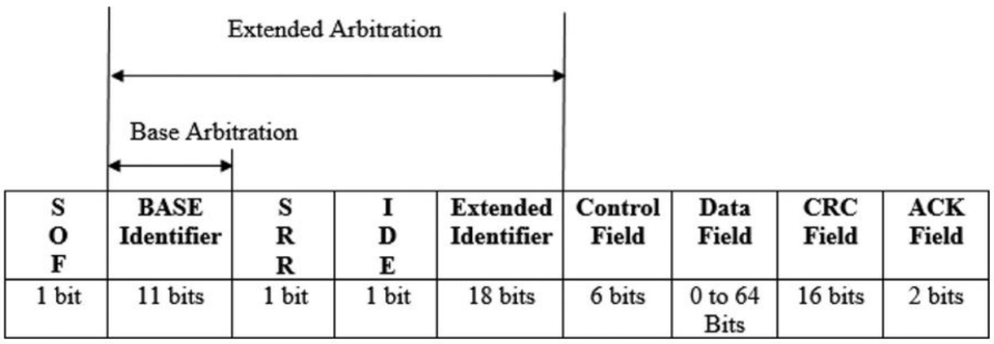


Figure 3. CAN used for ECM communication in cars.

In the application layer, device Net is used as a networking protocol in the automotive industry. It reduces the wiring between I/O devices and the control system.

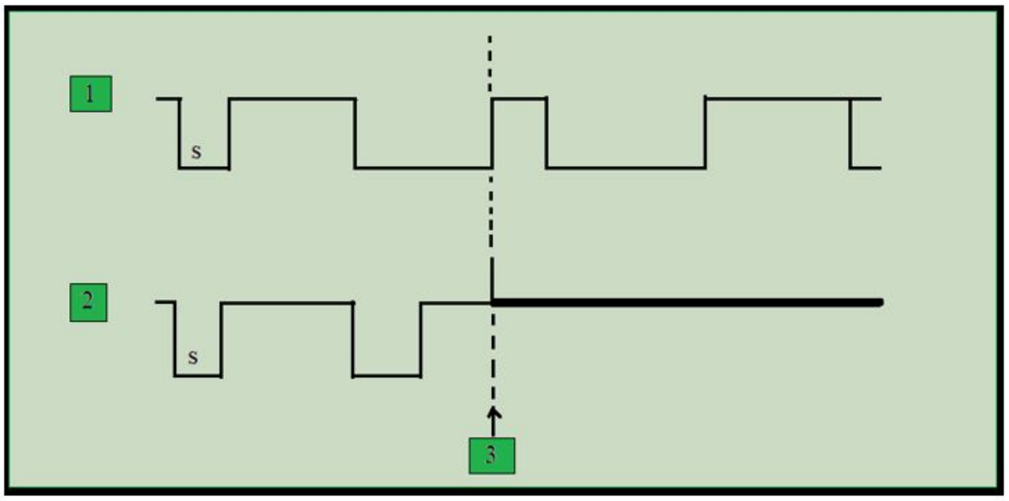
CAN devices transmit data across the networks in frames called CAN frames. CAN frames consist of the following fields. Figure 4 illustrates the CAN frame format, and its details are presented in [8].

CAN communicates as a peer-to-peer network, i.e., there is no master role to control the transmission. Devices (CAN node) will write on the CAN frame when it is ready and if the bus is not busy. The CAN frame does not contain the recipient address, but it has a message priority bit (Arbitration ID) which is unique throughout the frame. All the nodes receive the frame based on an arbitrary bit of the transmitted frame; each node decides to accept the frame received. When there is multiple nodes transmission, based on an arbitrary bit, the node with the highest priority is given access to the bus, and the lower priority node needs to wait until the highest priority node completes its execution. This illustration is demonstrated in Figure 5.



S O F	BASE Identifier	S R R	I D E	Extended Identifier	Control Field	Data Field	CRC Field	ACK Field
1 bit	11 bits	1 bit	1 bit	18 bits	6 bits	0 to 64 Bits	16 bits	2 bits

Figure 4. The standard CAN frame format [adopted from [8]].



1 Device A:ID: 11001000111
2 Device B:ID: 11011111111
3 Device B loses Arbitration, Device A wins Arbitration and Proceeds
S = Start Frame Bit

Figure 5. CAN contain built-in priority for messages to avoid conflicts.

1.2. Security Threats in CAN Protocol

CAN are vulnerable to attacks as they do not inherently possess any security features. Meanwhile, they can be protected by implementing some additional security features. With the lack of an authentication scheme, the CAN system allows unauthorized nodes to join the communication network. Due to the lack of encryption in the CAN system and the way transmission with which CAN messages are broadcast, all nodes in the CAN communication can receive and intercept the value. This leads to resource and data leakage during transmission.

A malicious node can have a higher priority with an arbitration bit set to access the CAN bus infinite times, and this may result in resource leakage and finally lead to corrupting software and physical damage of hardware in the system. CAN protocol can be analyzed with the security model based on the Confidentiality, Integrity and Availability (CIA) triad as shown in Table 2.

Table 3 summarizes security threats on electronic equipment on vehicles on various access points during in-vehicle communication. These threats will affect driver's safety.

1.3. Objective and Scope

Network-enabled devices are vulnerable to security attacks discussed previously, and these attacks may be life-threatening for passengers.

The objective of the paper is to define and implement the security mechanism to reduce the risk associated and reduce attacks on the vehicular environment.

The scope of the implemented security mechanism can also be applied in other fields such as medical, crypto-mining, etc.

Table 2. CIA security model analysis of CAN protocol.

Security Triad	Security Analysis
Confidentiality	• CAN protocol does not have any security measures. • Manufacturer uses cryptographic methods for local functionalities such as keyless entry.
Integrity	• CAN has checksum that validates the bits corrupted during communication. • Modified data from outsider cannot be detected, and therefore, CAN fails on w.r.t integrity.
Availability	• Not possible by CAN because of the arbitrary rule and its implementation. • If a note with higher priority transmits all the time, the bus cannot be accessible by the other nodes.

Table 3. Security threats on vehicle electronic system

Attack Surface	Security Threats	Related to
Communication in vehicle (Wireless) (Bluetooth, RF)	(1) Traffic signal control. (2) Sending fake message	Integrity
Communication environment (Wireless) (V2V / V2I)	Injecting messages into the CAN bus	Integrity, Availability
Diagnostic interface (OBD II)	Injecting messages on CAN bus	Integrity, Availability
Infotainment system	Unauthorized vehicle control	Safety/Security
Physical tampering	(1) Odometer fraud. (2) Illegal tuning of engine	Integrity
CAN Traffic	Eavesdropping is a serious concern that necessitates CAN data encryption	Confidentiality

1.4. Intrusion Detection System

To counter and identify these attacks, there is a need for multiple systems to prevent or detect attacks. The first layer of the security intrusion prevention system comprises the firewall, and the intrusion detection system resides in the second layer. If the intrusion prevention system is unable to prevent the attack from happening, therefore, it is the responsibility of the intrusion detection system to detect malicious activity happening and keep records of activity for future analysis. These recorded data will be used to update the system in the prevention stage and help in the detection of specific attack categories in the future. Though there is a significant improvement in the intrusion detection system as part of the network security detection systems, these systems have a weakness as they provide a signature-based attack classification pattern which detects most known attacks but is unable to detect and classify unknown attack types.

IDS plays a vital role in network security as its objective is to prevent disruption in the communication network. Ideally, static probability, neural networks, and deep learning algorithms will be used to evaluate if data are malicious or not [9]. To overcome signature-based attack classification drawbacks, the research focuses on dynamic approaches based on machine learning. Here, actions of the network are learned by using the data recorded to identify new types of attacks. Work on machine learning has expanded the security of the systems overall[9], and the major research topic in network security is IDS due to the significant impact of attacks and violations in real-time user applications [9]. Denning et al. [10] specify an efficient model of real-time IDS systems that can detect penetration and break-in of attacks in computer attacks. The model should include profiling the attacks that are unclassified and learned later. Mukherjee et al. [11] mention that there is a new type of challenge in IDS due to the proliferation of different networks since they have an impact on connectivity issues, giving easy access to an outsider.

The major contributions of this paper are as follows:

- Most of the researchers had considered one evaluation metric for evaluating the performance of their IDS model, whereas our model is evaluated by taking all evaluation metrics, that is, accuracy, precision, and F-1 score.
- Based on the results obtained, our model has achieved high efficiency in terms of accuracy and overall loss while training.
- We have considered multi-class classifiers in contrast to the majority of the research paper as they have considered the binary classification models in their studies.

This paper studies the following key issues.

- Before implementing the method, the dataset is analyzed using data processing methods to learn the distribution of data.
- Based on the distribution of data in the dataset, various sampling methods were used (normalization and standardization) to distribute data in a specific scale range.
- After proper distribution of the dataset, a level of the neural network is implemented and learning weight is updated to predict the accurate attack class.
- The model is evaluated based on accuracy, precision, and recall.
- Additionally, the implemented model is experimented with by adding more neurons and dense layers until function loss (cross-entropy) shows minimum loss while training the model.
- Finally, results are compared with other methods used in the literature to showcase the accuracy of the methods used.

Here, we will be modeling an efficient Intrusion detection system that will detect anomalies in a real-time generated CAN dataset [12] using three cars. The remainder of the paper is organized as follows: Section 2 discusses brief literature reviews of various intrusion detection methods in vehicular systems and network communications. Few papers discuss how efficiently machine learning algorithms can be modeled as IDS with the use of different datasets, and few papers discuss testbed for the generation of data in vehicular systems. Section 3 discusses the proposed method, i.e., how the data are generated from the vehicle, and how the model is implemented to get a high detection accuracy. Section 4 shows the result and findings of the implemented model. Section 5 summarizes the paper and provides the path for future work that can be considered.

2. Literature Review

Works related to CAV have emerged recently with the potential to impact the daily lives of individuals. Many governmental policies were introduced to support and accelerate the development of CAVs. The features and roles played by CAVs in real-life scenarios have attracted researchers to examine this technology and enhance the features that may help to build better road safety infrastructure.

Qiyi He et al. [2] proposed a CAV cybersecurity framework for the generation of a new communication dataset, i.e, the CAV-KDD dataset which classifies the vulnerabilities in CAV systems considering the existing KDD dataset as a benchmark and addresses potential attacks by building a UML-based CAV framework to analyze attack threats to CAVs inspired by the UK CAV cybersecurity framework that supports and gives solutions for secure CAV systems and data transferred. Mehdi Moukhafi et al. [13] capture real-time data using simulated LAN US air force LAN that were captured during nine weeks with multiple attack types that are used as the highly trained dataset. This dataset was captured based on the KDD99 cup. The datasets captured in these two articles overcome the limitations of attack types, i.e., using these frameworks more attack types can be injected.

To generate accurate in-vehicle data, Vita Santa Barletta et al. [14] and Huaxin Li et al. [1] generated data by accessing the CAN network via OBD-II of the real vehicle. Messages were injected to perform a specific attack on the CAN bus, and data between two sensors were analyzed to estimate the category of the attack. Based on the data obtained, the estimated value and actual value obtained are evaluated and considered to detect anomalies. To obtain an actual value, a regression model is executed for estimating the relationship between variables, and this is used for training the model. Vehicular speeds are used as the feature for the model, and they are collected from homogeneous sensors and sent over multiple ECMs of CAN bus. A random forest regressor is used for the feature fetching stage, and the estimated value is given as output at each prediction. Detection is based on the parameters of the CAN messages. The difference is calculated between estimated value and actual value categories. Initially, the model was tested on one of the datasets based on the distance that was the input to the k-means algorithm, and the distances between the weighted input CAN vector and neurons in rx maps were computed to train the Kohonen SOM network.

The new model by Mohammed Hasan Ali et al. [15] is designed based on an Artificial neural network (ANN) to reduce the false positives and false negatives metrics as ANN can learn efficiently from the examples. The paper develops a model based on a fast-learning network (FLN) based on Particle Swarm Optimization (PSO) based on KDD99 datasets for performance evaluation. They use an ANN that approximates complex nonlinear mapping from input parameters that will be used in applications that have proven to be efficient in terms of performance metrics. Around 10 percent of the features in the datasets were discarded as they were not related to any attack. The paper emphasizes how to find the optimal values of hidden layers in a Single hidden layer Feed-forward Neural network (SLFN) and FLN to get high accuracy from the model. Similarly, Peiying Tao et al [16] propose an FWP-SVM-Genetic Algorithm (GA) to reduce the false positive and false negative. Here, they propose a new FWP-SVM-Genetic Algorithm (GA)., feature selection, and weight and parameter optimization of SVM based on GA. The algorithm first optimizes the crossover probability and mutation probability of GA according to population evolution algebra and fitness value. Then, it uses the feature selection method based on GA with an innovation in fitness function that decreases the SVM error rate and increases the true positive rate.

The FLN method proposed in Mohammed Hasan Ali et al. [15] is based on an Extreme Learning Machine (ELM) concerning distribution and assignment of optimizing weights. Using the approach of Particle Swam Optimization (PSO), optimization is based on designing a particle to represent the solution of FLN weights. Results of the proposed model show that an increase in neurons in the hidden layer increases the accuracy and that class R2L category accuracy is more efficient than other classes. Even though results were efficiently better than those of other approaches, its limit is inaccuracy in a particular class, i.e., it fails to perform efficiently in class R2L unlike in other classes. This limitation is improved by using the FWP-SVM-Genetic algorithm (GA) where optimal feature subset, the feature weight, and parameters of SVM are simultaneously optimized. Results show that there is an increase in intrusion detection rate, accuracy rate, true positive and decrease in false-positive, and reduced training time of SVM. The limitation in detection rate was addressed by James Brown et al. [17] who proposed a new Evolutionary General Regression Neural Network (E-GRNN) model that classifies classes for implemented Evolutionary General Regression Neural Network (E-GRNN), classifying classes for intrusion detection system on features of application layer protocols that were simulated from the network simulated environment. Another article by Sunil Kumar Gautam et al. [18] improves detection rate and ensures the security of the information data against cyberattacks.

James Brown et al. [17] use network data generated from the UNB ISCX Intrusion Detection Evaluation Dataset. These data were extracted from the features in the application layer used in network activities. The model selects salient features from the feature set in the dataset to increase detection accuracy and decrease the computational complexity of the model. The General Regression neural network consists of Gaussian neutrons created with kernels functions using training instances. To classify the labeled weight, averages are used that are calculated using fire strength. The E-GRNN model was able to achieve a detection rate of 93.63 percentage, and there were fewer false positives and false negatives. This model is more suitable to solve problems when normal and abnormal behavior is not static. To overcome this limitation, two methods of the computational model are designed for host intrusion detection systems, i.e., General Regression Neural Network (GRNN) and Multilayer Perceptron Neural Network (MPNN) were modeled by Sunil Kumar Gautam et al. [18]. They used an offline dataset that was generated considering the evaluation of the system's performance. This dataset consists of 15 features and 2000 records for intrusion detection, and the accuracy of this model was found to be around 98.5%, which is critical for IDS. The article mentions that the proposed method can be used for different types of data mining for intrusion detection in the future.

Saman Masarat et al. [19] implemented a new multi-step framework to create an efficient classifier based on a machine learning algorithm and summarized a few artificial intelligence and data mining techniques used in IDSs by listing out the advantages and challenges faced by the model while implementing a solution for classifying the attacks effectively. However, there is mention of computational costs and memory constraints taken into consideration while modeling. This is carried out in Doohwan Oh et al. [20] by proposing a lightweight security system to reduce memory constraints and computational costs in terms of power and time which limits the efficient use of IDS. To mitigate the performance degradation due to memory constraints and computational power, the paper proposes two methods, namely, auxiliary shift and early decision. Firstly, the paper proposes a malicious pattern detecting system that has lower computational complexities and requires small memory to protect against security breaches in the system. This method was further modified by using auxiliary shift values to reduce the computation costs further. If no matching pattern is seen, by deciding early on the patterns, the memory and the computational cost can be further reduced.

Life threatening attacks in vehicular ad hoc networks (VANET) were addressed in Fuad A. Ghaleb et al. [21] by taking into consideration that these networks share their movement information and status information with surrounding infrastructure and so they are prone to cyber attacks. The author mentions that existing cooperative IDS (CIDS) are vulnerable to malicious attacks as these systems leak and manipulate personal data and disrupt the normal behavior of IDS. The paper proposes a new Misbehavior-aware on-demand Intrusion detection system (MA-CIDS) based on distrusted ensemble learning. This uses random forest for training the local IDS classifier, and after training, the model shares locally on-demand with the vicinity of the vehicle. Many studies and models were developed based attacks related to the CAN bus.

Yoshihiro HAMADA et al. [7] propose an intrusion detection system for monitoring control data and payload in CAN protocol with the target if detecting an anomaly in the system. The article briefly explains the frame in the CAN protocol, the significance of each bit, and the way malicious users manipulate these bits to gain unauthorized access to the intended system. The paper also explains and compares the performance of the different Intrusion detection systems in various applications such as network and vehicle applications. Mrugnayana S. Savekar et al. [22] produced a model for detecting impersonation attacks that originate from injecting malicious messages between the source and destination of the message in VANETs. Due to the mobility characteristics of VANETs, the message from one node is transmitted to others, and these systems are prone to attacks such as impersonation attacks. This paper briefly examines different models in its literature review and finally models using SVM and k-NN algorithm on the CAN intrusion dataset generated from the Hacking and Countermeasures Research lab (HCRL). In this paper, two attacks were targeted, namely, the DoS attack and the Fuzzy Attack. The paper also mentions that the model predicts different attack types as well. The paper also contributes to the practical comparison of kernel functions of the SVM algorithm. While they examine the CAN related datasets, there are not security measures taken in the picture to CAN message time interval. This is carried out in Hyun Min Song et al. [23]to enhance the level of vehicular security by proposing a lightweight detection method based on the analysis of time intervals between CAN messages.

Zadid Khan et al. [24] mainly focus on the limitations from the papers reviewed, i.e., some papers do not consider CAN features for attack detection, and few consider only CAN features for attack detection. It develops a software solution for an in-vehicle network for protecting against false information injected packets and illustrates a few countermeasures against an ECU attack first by detecting malicious activity by measuring the few attributes from the multiple sensors in the system. This paper presents SDN-based in a vehicle and analyzes the security features of the network, i.e., analyzes information and categorizes it into false and normal behavior. This paper considers two applications of SDN for detection of attack model. Moayad Aloqaily et al. [13] mention transportation will be significant and will play a prominent role in shaping the automobile industry beyond recognition. The emergence of these smart vehicles gave prominence to vehicular cloud availability that plays as service manager for the vehicles in the smart city environment. There is a need for high-performance cloud architecture free from malicious attacks on the system. This article introduces an automated continuous secure cloud service framework that enables an intrusion detection mechanism against cyber attack categories and meets user constraints w.r.t Quality of Service (QoS) and Quality of Experience (QoE). This service cloud was developed by binding smart vehicles to the cluster of specific services available in the cloud. A vehicular node's services are selected based on cluster cloud and on communication infrastructure through third party entities that function as mediators between service providers and requesters.

Niraj Thapa et al. [3] proposed a new deep learning model for a network intrusion detection system (NIDS) by comparing different machine and deep learning models for NIDS. The paper explains the need for a dynamic security system to identify unknown attacks on the network by pointing out the limitations of static signal-based network intrusion available presently. The paper firstly models different ML and DL models on Coburg intrusion detection datasets (CIDDSs) which were captured using an internal network by emulating business environments using open stack and using an external network server. CIDDS dataset comprises two versions and has 92 and 22 attacks in an open stack and external server environment, modeling k-NN, XGBoost, and CART. The algorithm of the model is evaluated based on metrics such as accuracy, precision, and finally performance and is noted for further reference. By using the results obtained previously, the paper proposes an efficient model , combining both ML and DL to achieve high performance. Finally, the paper benchmarked the model based on the CIC-IDS2017 dataset for comparison with different models. These models are used based on the required performance metric and cost functions in case of training time. These models, i.e., CNN + embedding, were compared with previous work. With LSTM + embedding, it is noted that improved accuracy is achieved. To improve the performance of the detection rate and reduce bias towards frequent attacks Longjie Li et al. [25] propose a hybrid method based on binary classification and k-NN algorithm. The paper uses the NSL-KDD dataset for performance evaluation. This method is modeled in two steps, first, employing several binary classifiers that have the role of detecting the classes of network connections. After step 1 has evaluated the classes which were uncertain and which it was not possible to classify, these were passed through the k-NN algorithm. The implemented model is compared with five other machine learning techniques and with other hybrid models. The results of the implemented model show that there is an increase in detection rate and accuracy of the hybrid model over another single model approaches.

The paper by Moayad Aloqaily et al. [13] proposes a three-phase intrusion detection mechanism that is incorporated within entities that are participating in the communication. Vehicle cloud data will be the first to go with an intrusion detection algorithm. This involves analyzing data and deleting redundant information in the communication dataset. Finally, hybrid solutions are based on the Deep Belief Network (DBN), which is based on un-supervised learning that uses D2H IDS for dimension reduction and ID3 for attack classification and which is based on Decision Tree. This DBN initiates DNN parameters and creates their dataset using a packer generator, and anomaly behaviors are injected by inserting noise and manipulating packets. As for the DNN algorithm, LASTM is executed to detect the attacks, and this works on the raw data. The proposed solutions achieved an accuracy of 99.43 with a detection rate of 99.92. The paper also mentions that the proposed model can be utilized to collect big data from vehicles. These data can be analyzed using an intelligence model that is used to improve the security of vehicles and improve road safety infrastructure. Yusuf Sani et al. [26] presented an overview of a Neural Network in the use of an anomaly intrusion detection system. The paper gives an overall review of how a neural network works and the way it improves the efficiency of detecting anomalies compared to other systems. The paper briefly explains the IDS system, the classifiers in the system, and different methods of implementing the IDS system based on the features in the data. Many researchers have used this paper for their reference in developing an efficient Neural network-based IDS system.

Valentin Zieglmeier et al. [27] testbed is implemented on the use case based on a client-server architecture that is used popularly in the automobile industry. Here, the server will be a reference, and vehicles are abstracted from the outer perspective that sends a data-request to the server. Based on the communication with the server, the normal and compromised vehicles are differentiated. Intrusion may be because of the data generated by a software bug or modification of data generated by components that differ from the normal actions of the system. This model is implemented using the Robot Operating System (ROS) and is programmed in python. This test best solves the existing problems in existing datasets such as few features, class imbalance, and redundancy. The testbed is evaluated with a comprehensive solution for IDS. The reproductivity of the data in this test is high and can be tested multiple times to ensure performance. However, Dupont, Guillaume et al. [12] generated real-time in-vehicle communication protocol data, i.e., CAN data in the urban environment for two vehicles speeding around 40 miles/h. Attacks were injected by manipulating the generated data. The data are captured in real-time with a laptop connected to the CAN interface port (OBD-II port). Based on this dataset, our implemented model performance is evaluated in the paper.

Table 4 shows the comparative performance review of the proposed models discussed in the literature review.

Research Gap

By reviewing the work of various research fraternities, it is observed that the following points are not addressed correctly in many articles. The summary of the research gaps from the above research papers is as follows:

1. Only a few have used datasets related to vehicles for their analysis.
2. Most of them use traditional classifiers that limit performance and creativity.
3. The majority of IDS models are not evaluated using real-time datasets.
4. The major challenge is the availability of relevant datasets related to vehicular systems.
5. Most of the work in the papers used only one performance metric in evaluating the proposed model.

Table 4. Comparative performance of models discussed in the literature.

Paper Cited	Model	Accuracy(A) or/and Precision(P)or/and Recall(R)(%)	Dataset	Dataset Type and Related to	Classifier	Performance Metric
He, Q et al. [2]	Naive Bayes	99.42(A)	CAV-KDD	Combining real-time and traditional network traffic	Binary	Accuracy
He, Q et al. [2]	J48	99.8(A)	CAV-KDD	Combining real-time and traditional network traffic	Binary	Accuracy
Moayad Aloqaily et al. [13]	Hybrid (Deep brief network, Decision tree)	99.43(A)	NSL-KDD	Available network-traffic	Multi-class	Accuracy
Niraj Thapa et al. [3]	k-NN	94.22(A)	CIDDS	Available network-traffic	Multi-class	Accuracy
Niraj Thapa et al. [3]	LSTM	98.91(A)	CIDDS	Available network-traffic	Multi-class	Accuracy
Niraj Thapa et al. [3]	CNN+ Embedding	99.11(A)	CIDDS	Available network-traffic	Multi-class	Accuracy
Niraj Thapa et al. [3]	LSTM+ Embedding	99.14(A), 94.54(P)	CIDDS	Available network-traffic	Multi-class	Accuracy, Precision
khan long et al. [24]	PSO-FLN.	98.78(A)	KDD99	Traditional network-traffic	Binary	Accuracy
Ghaleb F et al. [21]	SVM-PSO	91.66(A), 89.87(R)	NSL-KDD	Available network-traffic	Binary	Accuracy, Recall
M. Moukhafi et al. [28]	RNN and GRU	99.42(A), 99.31(A)	KDD99, NSL-KDD	Available network-traffic	Multi-class	Accuracy
M. H. Ali et al. [15]	Binary-k-NN classifier	94.92(A)	NSL-KDD	Available network-traffic	Multi-class	Accuracy
L. Li et al. [25]	FWP-SVM-genetic algorithm	96.61(A), 83.89(P)	KDD cup99	Traditional Network	Binary	Accuracy, Precision
P. Tao et al. [16]	E-GRNN	96.38(A)	KDD99	Traditional Network	Binary	Accuracy

3. Proposed Method

Attacks can be injected into the CAN bus. In real-time eavesdropping, false data injection, and DoS attacks are carried out because of lack of encryption. Data can be manipulated by injecting an unauthorized CAN frame into the network as the CAN frame lacks authorization. By using this approach, Guillaume Dupont et al. [12] have generated CAN datasets which are used in this research.

3.1. Generation of Dataset

In the paper, Guillaume Dupont et al. [12] have generated data in real-time with two cars, namely, a Renault Clio and an Opel Astra, and with another obtained by prototyping (it consists of a VW instrument cluster, 2 Arduino boards with CAN bus shields, and a joystick); detailed information is provided by Reference [12].

In the case of cars, normal data are captured by driving the car in the urban environment, with a laptop and CAN to usb-interface connected to the OBD-II port, and then dumping the data captured from the CAN utility tool.

Data manipulations are carried out on CAN bus data to hack the automobile units such as the body control, engine control, and brake control modules of the car. These modules display false readings of fuel level and speedometer reading and also prevent the activation of critical car modules such as brake and airbag warning signals. These types of attacks will hide the theft event and risk the life of passengers resulting in the unreliability of security measures provided by the car manufacturer. Moreover, cars can be unlocked via the CAN network as modern cars provide various types of wireless interfaces such as keyless entry and infotainment systems. This wireless interface communicates with the CAN network via a gateway. The attack modifies the ECM of the car, and access to the car can be obtained without a key.

The motivation for generating this dataset is to generate accurate data in terms of the vehicular environment in real-time. Several IDS were implemented on generic datasets such as KDD99 and CIDDS, and few works involve modifying this generic dataset by adding relevant data related to vehicle action.

Data captured have been modified to get a specific CAN bus attack. This dataset is available for research purposes in the field of IDS for CAN bus. In this work, some attacks are injected, and the role and impact of the attacks are explained in Table 5.

Table 5. Attack types used in the paper.

Attack Type	Role	Impact
Dos Attack	Incoming traffic floods victim; these originate from one source.	Machine or network will be unavailable to provide its services for its users.
Reconnaissance Attack	Configure, monitor the attacks.	Hidden software backdoor is highly vulnerable, and entire system in network can be affected by knowing backdoor.
Fuzzing Attack	Bug detection by providing invalid, unexpected, or random data as input	Causes unexpected behavior, resource leakage and crashes.

Different datasets are captured by injecting attacks and these are injected as follows:

1. **Normal data**: No modifications are made to the captured files. There are 3,735,198 normal data.
2. **Attack data**: These datasets have been created by modifying copies of the testing dataset. Each copy presents a specific CAN bus attack. In case of injection attacks, the packets are added, and timestamps are adjusted accordingly.
3. **Reconnaissance**: These attacks were injected into all platforms (car and prototype) where CAN ID between values ’700’ and ’7FF’ were added. This attack type consist of 1677 normal data.
4. **Dos**: Attacks with CAN ID ’000’ (with high priority) are injected for 10 seconds, which creates a 500 kbps flood on the CAN bus. Here Dos packets are replaced in a block of 10 s with a high priority CAN message at a rate of 4 packets per millisecond. This attack type consists of 120,053 normal data.
5. **Fuzzing canid**: These attacks were injected and are not a part of legitimate values (i.e., CAN ID values that are not seen in normal data).
6. **Fuzzing payload**: Attacks consist in modifying the payload from CAN ID ’0C9’ (Opel Astra), ’5A0’ (prototype), and ’18A’ (Renault Clio) to payload value that is not used in legitimate traffic.

Features of the dataset consist of five columns:

1. **Timestamp**: Displays the time of data captured. Responsible for detecting a few types of attack as discussed in attack datasets.
2. **Protocol**: In this case, it will be the CAN protocol as all the data were captured on the CAN bus.
3. **CAN ID**: Determines source of data and type of CAN service and differentiates normal and abnormal packets.

4. **Payload**: Defines the actual information that needs to be addressed. It may consist of false data injected by the attacker.
5. **Attack label**: Determines whether actual row is normal or a specific attack type based on certain criteria.

All these datasets were captured in .log files, and for this paper, before being subjected to the pre-processing stage, all these data files were converted to the .csv format by programming in java script. Datasets captured from all three platforms are merged, and the distribution of the output label is shown in Table 6.

Table 6. Value repartition for attack label.

Attack Label	Distribution
Normal	3,735,198
Dos	120,053
Reconnaissance	1677
Fuzzing CanID And Fuzzing Payload	4993

3.2. Methodology

A key limitation of existing anomaly detection IDSs is that they cannot easily categorize and define normal and abnormal actions efficiently. In this paper, an artificial neural network is modeled that can classify behavior by learning and work efficiently in pattern recognition.

Neural Network

The neural network is built in a similar way to the one used by the human brain to process and take decisions based on critical information. Based on the idea that they are built to take critical decisions, these networks are popularly known as Artificial Neural Networks (ANN). ANN were initially developed to work like a human brain on problems such as pattern, image, and speech recognition, getting the best possible results. The provision of a trainable model in ANN provides the luxury of improving the results to a larger extent.

Neural networks are composed of basic processing elements called neurons just as the human brain that sends information from the brain to the nervous system. Neurons are interconnected with many layers and are responsible for processing the information based on the weight on each layer while processing.

The layer known as the Dense layer that interconnects Deep Neural Networks (DNN) consists of neurons and operates on input and produces an output. The idea is that the input is taken at the input layer; the dense layers perform the processing and update the weights, and the output layer is given as input to another input layer, and finally, the output layer produces an output. The results obtained are the characteristics input elements and weights inclined with the layers that can be controlled and updated. The weight is updated based on the learning of the neurons associated with each interconnected layer. Figure 6 shows a Neuron model.

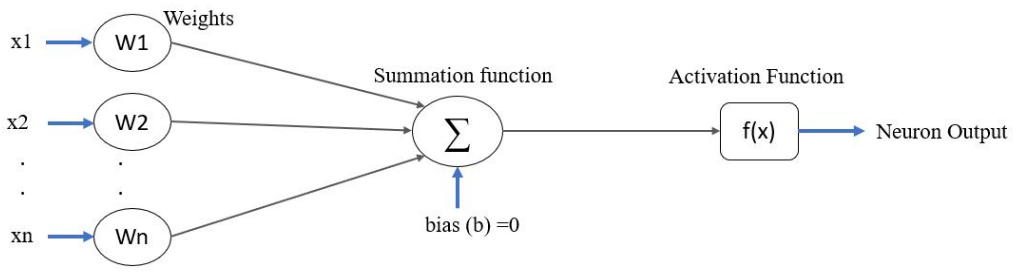


Figure 6. Neuron model

As Figure 6 illustrates:

1. x1 and x2 are the inputs (features in dataset) to the neural network.
2. W1 and W2 are the weights associated and are updated when the neurons are leaned on on each layer.
3. Neuron logical block performs summation operation based on the input and weights.
4. Y is the Activation function used based on the arrangement of data distribution.
5. After processing the data in activation function, neurons produce output (type of classifier).

Based on the arrangement of neurons, the architecture is defined.

1. **Feedforward network (FNN)**: Neurons are arranged in such a way that the first layer will be input to neurons, and the output of these neurons are the input of second layer neurons. This layer continues until the final layer is the output of the network.
2. **Recurrent Network (RNN)**:Neurons are chained together end to end.

The detailed information of the various types of neural network models is explained and described in [26].

3.3. Architectural Model

Reason behind choosing the Neutral Network for our implementation:

Deep Neural Network is chosen for modeling IDS because of its potential for training. It continuously adapts and learns the outcomes of normal and abnormal behavior of the system [15,16].

The advantages of modeling a Neural Network in IDS are the following:

1. A Neural Network provides accurate statistical distribution with fewer assumptions and is efficiently modified by learning.
2. A Neural Network has a low cost during development and it can be scalable compared to others.
3. They perform better in reducing false positives and false negatives that are important in IDS.

Some of the limitations of the existing machine learning methods that justify the use of neural networks for our application.

SVM

1. It is slow and requires a large amount of time to process a large dataset.
2. Its performance degrades when dealing with a dataset which has overlapped classes.

Logistic Regression

1. Poor performance when the data are non-linear.
2. Not a powerful algorithm, and it can be replaced by others easily.

Random Forest

1. Acts as a black box to analyze the modeling.
2. Power to predict accurately should be a feature.

Decision Tree

1. Prone to over-fitting
2. Requires enormous time to train the data.

k-NN

1. Slow for a large dataset.
2. Fails to predict when there is a missing value in the dataset.

The proposed method will have a monitoring module that monitors incoming CAN packets based on trained features of known attacks. If the packet received is identified as a new attack, then the profiling model records the attack type and updates the system for new packets. This is illustrated in Figure 7.

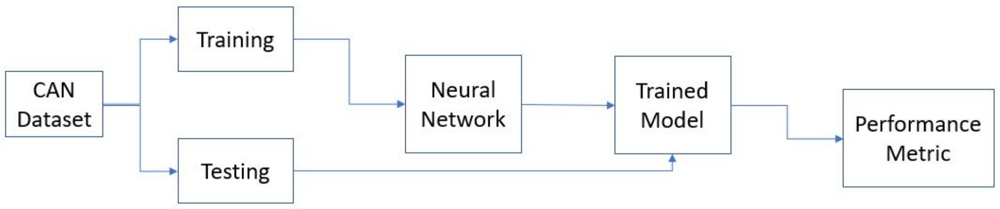


Figure 7. Modules of proposed IDS

The flow process of the proposed solution for anomaly detection is shown in Figure 8.

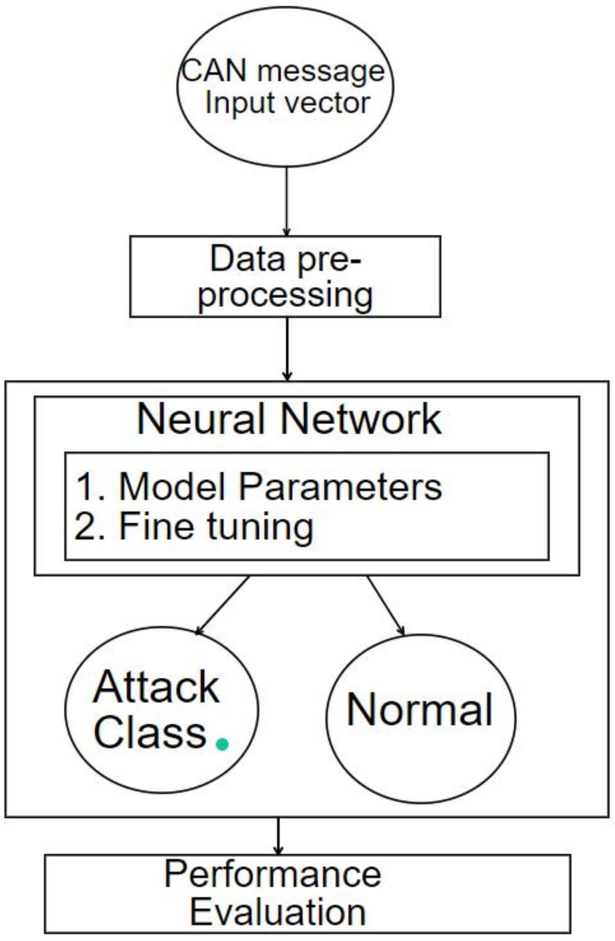


Figure 8. Flow chart of implemented model.

The following processes are undergone before training the data.

Here the CAN message input vector is a feature present in the dataset.

Data pre-processing:

1. Here, the data captured were composed of categorical data of object (string) type. This type of data should be encoded using one of the encoder mechanisms.
2. Attack label and protocol types are encoded using the dummies encoding method that will transform object type to float type. We took this encoding method because very few unique values were there in specific column categories.
3. For the data containing multiple unique values, in the case of CAN ID and payload, we have implemented the embedding column encoding method.
4. We assigned 75 percent of the packets to training data and 25 percent of the packets to validating it to avoid overfitting problems that arise while training. This is carried out by using the scikit learn API.

Implementing model:

1. After data are pre-processed, the next step is to create a neural network using the TensorFlow API.
2. Here we have used a sequential method to create a neural network, and we are adding two dense layers with five neurons each and the input features size.
3. The 'Relu' activation function is used more than others based on input data distribution (linear distribution), ensuring there are no negative outputs from any neuron.
4. At the output layer, we are specifying the size of the output label, using the 'SoftMax' activation function that specifies multi-class classification.
5. We are using the 'Adam' optimizer with a learning rate of 0.01 and compiling our model with a categorical cross-entropy loss function model.
6. Finally, using xtrain (featuring values used for training and prediction) and ytrain (output label values), the model is trained, and the performance metric is determined.

Detection:

1. In this step, we predict the class of testing CAN packet with a trained Deep Neural Network (Figure 9). Here, the output is calculated based on trained weight parameters and extracted features from the CAN testing packet.
2. When input features are given to the network.
   - In the case of a binary classifier, output is determined as either 0 or 1, specifying whether the packet is normal or abnormal.
   - In the case of the multiclass classifier, the output is determined based on the classes' normal or specific attack category.

The performance of the model is validated using a real-time generated CAN dataset by determining accuracy, precision, recall, and F-1 score.

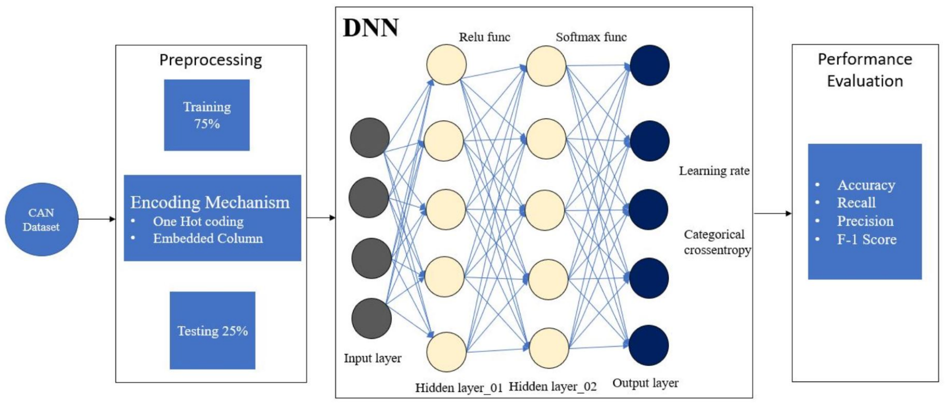


Figure 9. Structure of the proposed DNN model.

4. Results and Findings

The dataset used is an emerging dataset generated in November 2019 and revised in August 2020. The purpose of this dataset is to evaluate CAN bus Network Intrusion Detection Systems. Guillaume Dupont et al. [12] mention that implementing IDS based on this dataset will be an ideal choice for their future work.

Experimental Setup

The model is implemented in python using machine learning libraries such as tensor flow and scikit-learn in Google Collab.

4.1. Results

The overall performance of the model is evaluated on the basis of performance metrics, i.e., accuracy, precision, recall, and f-1 score. Comparison is carried out on different datasets with different models which are published by the research team. The metric used for comparison and what it specifies is given below.

- **Accuracy**: It shows the percentage of correctly predicted values compared to overall predictions.
- **Precision**: It refers to the ratio of correctly predicted positive observations to overall positive predicted observations.
- **Recall**:It refers to sensitivity. It is the ratio of correctly predicted positive observations to overall observations in the output label class.
- **F-1 score**: It refers to a weighted average of precision and recall. It is used when there is an uneven distribution of classes.

Table 7 shows the result of the performance metric of the proposed model multiclass classifier. The time and overall loss obtained while training the model are shown.

Table 7. Performance metric of the proposed model

Accuracy (%)	Precision(%)	Recall (%)	F-1 Score	Training Time	Overall Loss (%)
98.67	96.72	96.5	96.30	184s	15.24

The accuracy obtained in the DNN model is 98.67, which is phenomenally critical in the case of the IDS system. Other metrics such as precision and recall show that our model correctly predicted values.

We have implemented other IDS models (Random forest, Decision tree, and k-NN algorithm) using the same dataset to compare the performance metrics obtained in our model. Table 8 shows a comparison between our proposed method and other models for the same dataset used in this paper.

Table 8. Performance comparison with other standard classifiers

Model	Accuracy (%)	Precision (%)	Recall (%)	F-1 Score	Training Time	Overall Loss (%)
DNN	98.67	96.72	96.72	0.9671	184s	15.24
Random Forest	97.92	95.12	94.96	0.9505	519s	N/A
k-NN Algorithm	94.89	92.8	91.78	0.9212	93s	N/A
Decision Tree	96.38	94.18	93.54	0.9407	59s	N/A

As shown, our model improves the IDS system in all performance metrics compared to another standard classification models which were used on our dataset only for comparison. The random forest also shows a significant improvement in metric compared to the other two. Similarly, comparison is carried out on different datasets with different models, which are published by the research team as shown in Table 9.

4.2. Experimentation and Findings

Below are few observations made while fine-tuning the parameters and experimenting with the implemented model:

1. The model can be implemented in numerous ways using tensor flow libraries, e.g., here we implemented it using a sequential API and also using DNN classifier libraries. Metric results obtained are the same for both approaches.
2. Accuracy increases by minutes when dense layers are added until the model is learnt. Our model is experimented with by adding few layers, and the accuracy remains constant after two hidden layers.

3. Another parameter for which the detection accuracy significantly increases is the learning rate. The model performs efficiently with the slowest learning rate being 0.001 compared to the other learning rates of 0.01 and 0.1.
4. Once the model's learning process is completed involving the required number of neurons, it shows an spike in accuracy with addition of neurons to the network, i.e., we used 4 neurons for the input layer, and 5 neurons are sufficient for the hidden layers to train the model completely.
5. The epoch determines the number of times the learning algorithm will work through the entire training dataset. We monitored only two epochs, which were sufficient for our model to learn completely. In short, the number of epochs depends on the variation of data in the dataset.

Table 9. Overall comparison of classification models

Reference	Model	Accuracy (%)	Recall (%)	Precision (%)	Dataset
	Our DNN Model	98.68	96.72	96.78	CAN dataset
Yoshihiro HAMADA et al. [7]	Control Data Estimation anomaly detection Correlation data (CDEC)	93	N/A	N/A	CAN dataset
Niraj Thapa et al. [3]	Unsupervised Kohonen SOM Approach	82.55	27.91	61.76	CAN Hacking dataset
Barletta VS et al. [14]	long short-term memory (LSTM)	95	87	95	CAN dataset

Table 10 shows the value of the performance metric of the model with the various learning rates. As the learning rate decreases, the performance of the model progressively improves. Table 11 shows the effect of performance on the number of epochs.

Table 10. Performance of the model with different learning rates.

Learning Rate	Accuracy (%)	Precision (%)	Recall (%)	F-1 Score
0.1	96.59	94.25	93.82	0.9403
0.01	97.99	95.21	94.73	0.9497
0.001	98.68	96.72	96.72	0.9671

Table 11. Performance of the model with variation in the number of epochs.

Epoch	Accuracy (%)	Precision (%)	Recall (%)	F-1 Score
1	98.63	96.700	96.480	0.9659
2	98.68	96.718	96.718	0.9671
2	98.68	96.718	96.718	0.9671

5. Conclusions

The Internet of Vehicles (IoV) is gaining popularity as it enables vehicles to communicate with traffic externally and communicate with emergency modules internally. As this vehicular communication becomes popular, the CAN communication protocols become vulnerable to cyberattacks. As a result, there has been a rapid increase in research on automotive security.

The use of embedded and portal devices in vehicles provides communication outside and inside the vehicle's environment. As this vehicular communication gains popularity, the vulnerabilities of the communication protocol become critical. As a result, there has been a rapid increase in research on automotive cybersecurity.

The previous methods used different models for the benchmarked datasets, namely, KDD99, CIDDS, and CAN hacking data sets. In our paper, we used a dataset generated from the real-time environment of two cars (Opel Astra and Renault Clio) and of another one obtained by prototyping (consisting of a VW instrument cluster, two Arduino boards with CAN bus shields, and a joystick); these are addressed in [12]. We have encoded the data using an embedded column that improves with training, and a better performance can be achieved due to the dynamic nature of the embedding column encoding technique.

In this work, we propose an efficient solution for mitigating the attacks on the CAN bus protocol by modeling the Instruction Detection System using a Deep Neural Network (DNN) because neural networks detect actual correctness in the dataset.

Further observations are made based on the results of the proposed solutions, and it is examined whether the research gaps discussed in the literature review are filled.

The newly generated CAN dataset was analyzed using a classification model, namely, Decision Tree, k-NN algorithm, and Random Forest to test the accuracy of the dataset. Then, our model is proposed to improve the performance obtained in the classification model.

Our model performs efficiently well with an accuracy of 0.986 (98.68 percentage), a recall of 0.967, a precision of 0.967, and an F-1 of 0.967, which are critical for IDS. This efficiency is achieved by fine-tuning certain parameters while experimenting with the behavior of the model by varying the number of hidden layers, the encoding techniques, and the learning rate of the model.

For future studies, more types of relevant attacks can be injected into the CAN bus, and different machine learning approaches can be implemented on the generated dataset. The improvement of detection machine learning algorithms and the generation of new sets of data related to vehicular communication increases the research topic for future work related to automotive security.

Author Contributions: Conceptualization, S.T.; methodology, D.B. and S.T.; software, D.B.; validation, D.B.; formal analysis, S.T.; investigation, D.B. and S.T.; resources, S.T.; data curation, D.B.; writing—original draft preparation, D.B.; writing—review and editing, S.T.; visualization, D.B.; supervision, S.T.; project administration, S.T.; funding acquisition, S.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by a research grant from Fresno State Transportation Institute and funded in part by an ASEE CISE-MSI Mini Grant under NSF award CNS-2039244.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

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

References

1. Li, H.; Zhao, L.; Juliato, M.; Ahmed, S.; Sastry, M.R.; Yang, L.L. POSTER: Intrusion Detection System for In-vehicle Networks using Sensor Correlation and Integration. In *ACM SIGSAC Conference on Computer and Communications Security (CCS '17)*; Association for Computing Machinery: New York, NY, USA, 2017; pp. 2531–2533.
2. 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]
3. Thapa, N., Liu, Z., Kc, D.B., Gokaraju, B.; Roy, K. Comparison of Machine Learning and Deep Learning Models for Network Intrusion Detection Systems. *Future Internet* **2020**, *12*, 167.
4. Tayeb, S.; Pirouz, M.; Latifi, S. A Raspberry-Pi Prototype of Smart Transportation. In Proceedings of the 2017 25th International Conference on Systems Engineering (ICSEng), Las Vegas, NV, USA, 22 August 2017; pp. 176–182. [CrossRef]
5. Trueblood, F.; Gill, S.; Wong, R.; Tayeb, S.; Pirouz, M. A Data-Centric Approach to Taming the Message Dissemination on the Internet of Vehicles. In Proceedings of the 10th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 6 January 2020; pp. 207–214. [CrossRef]

6. Gill, S.; Wong, R.; Tayeb, S.; Trueblood, F.; Pirouz, M. Optimizing Connectivity for the Internet of Vehicles. In *Proceedings of the Future Technologies Conference (FTC) 2020*; Arai, K., Kapoor, S., Bhatia, R., Eds.; Advances in Intelligent Systems and Computing; Springer: Cham, Switzerland, 2021; Volume 3. [CrossRef]
7. Hamada, Y.; Inoue, M.; Adachi, N.; Ueda, H.; Miyashita, Y.; Hata, Y. Intrusion Detection System for In-Vehicle Networks. *SEI Tech. Rev.* **2019**, *88*, 76–81
8. Pan, L.; Zheng, X.; Chen, H.X.; Luan, T.; Bootwala, H.; Batten, L. Cyber security attacks to modern vehicular systems. *J. Inf. Secur. Appl.* **2017**, *36*, 90–100. [CrossRef]
9. Davis, A.; Gill, S.; Wong, R.; Tayeb, S. Feature Selection for Deep Neural Networks in Cyber Security Applications. In Proceedings of the 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), Vancouver, BC, Canada, 9–12 September 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–7. [CrossRef]
10. Denning, D.E. An Intrusion-Detection Model. *IEEE Trans. Softw. Eng.* **1987**, *13*, 222–232. [CrossRef]
11. Mukherjee, B.; Heberlein, L.T.; Levitt, K.N. Network intrusion detection. *IEEE Netw.* **1994**, *8*, 26–41. [CrossRef]
12. Dupont, G.; Lekidis, A.; den Hartog, J.; Etalle, S. Automotive Controller Area Network (CAN) Bus Intrusion Dataset v2. 4TU.ResearchData. *4TU.ResearchData*2019. Available online: https://data.4tu.nl/articles/dataset/Automotive_Controller_Area_Network_CAN_Bus_Intrusion_Dataset/12696950/2 (accessed on 30 December 2021).
13. Aloqaily, M.; Otoum, S.; Ridhawi, I.A.; Jararweh, Y. An intrusion detection system for connected vehicles in smart cities. *Ad Hocnetworks* **2019**, *90*, 101842. [CrossRef]
14. Barletta, V.S.; Caivano, D.; Nannavecchia, A.; Scalera, M. Intrusion Detection for in-Vehicle Communication Networks: An Unsupervised Kohonen SOM Approach. *Future Internet* **2020**, *I2*, 119. [CrossRef]
15. Ali, M.H.; Al Mohammed, B.A.D.; Ismail, A.; Zolkipli, M.F. A new intrusion detection system based on fast learning network and particle swarm optimization. *IEEE Access* **2018**, *6*, 20255–20261. [CrossRef]
16. Tao, P.; Sun, Z.; Sun, Z. An improved intrusion detection algorithm based on GA and SVM. *IEEE Access* **2018**, *6*, 13624–13631. [CrossRef]
17. Brown, J.; Anwar, M.; Dozier, G. An Evolutionary General Regression Neural Network Classifier for Intrusion Detection. In Proceedings of the 25th International Conference on Computer Communication and Networks (ICCCN), Waikoloa, HI, USA, 1 August 2016; Volume 6, pp. 1–5.
18. Gautam, S.K.; Om, H. Computational neural network regression model for host-based intrusion detection system. *Perspect. Sci.* **2016**, *8*, 93–95. [CrossRef]
19. Masarat, S.; Taheri, H.; Sharifian, S. A novel framework, based on fuzzy ensemble of classifiers for intrusion detection systems. In Proceedings of the 4th International Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, Iran, 29–30 October 29; Volume 8, pp. 165–170. [CrossRef]
20. Oh, D.; Kim, D.; Ro, W.W. A malicious pattern detection engine for embedded security systems on the Internet of Things. *Sensors* **2014**, *14*, 24188–24211 [CrossRef] [PubMed]
21. A Ghaleb, F.; Saeed, F.; Al-Sarem, M.; Ali Saleh Al-rimy, B.; Boulila, W.; Eljialy, A.E.M.; Aloufi, K.; Alazab, M. Misbehavior-Aware On-Demand Collaborative Intrusion Detection System Using Distributed Ensemble Learning for VANET. *Electronics* **2020**, *9*, 1411. [CrossRef]
22. Mrugnayana, S.; Savekar; Sandeep, A. Identifying Impersonation Attack in VANET using k-NN and SVM Approach. *Int. J. Future Gener. Commun. Netw.* **2020**, *13*. 1266–1274.
23. Song, H.M.; Kim, H.R.; Kim, H.K. Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network. In Proceedings of the International Conference on Information Networking (ICOIN), Kota Kinabalu, Malaysia, 13 January 2016; pp. 63–68. [CrossRef]
24. Khan, Z.; Chowdhury, M.; Islam, M.; Huang, C.-Y.; Rahman, M. Long Short-Term Memory Neural Networks for False Information Attack Detection in Software-Defined In-Vehicle Network. *arXiv* **2019**, arXiv:1906.10203.
25. Li, L.; Yu, Y.; Bai, S.; Hou, Y.; Chen, X. An effective two-step intrusion detection approach based on binary classification and -NN. *IEEE Access* **2018**, *6*, 12060–12073. [CrossRef]
26. Sani, A.Y.; Mohamedou, K.; Ali, A.; Farjamfar, M. Azman and S. Shamsuddin, An overview of neural networks use in anomaly Intrusion Detection Systems. In Proceedings of the IEEE Student Conference on Research and Development (SCOReD), Serdang, Malaysia, 16 November 2009; pp. 89–92. [CrossRef]
27. Zieglmeier, V.; Kacianka, S.; Hutzelmann, T.; Pretschner, A. A Real-Time Remote IDS Testbed for Connected Vehicles. In Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing (SAC '19), Limassol, Cyprus, 8 April 2019; ACM: New York, NY, USA, 2019.
28. Moukhafi, M.; El Yassini, K.; Bri, S. A novel hybrid GA and SVM with PSO feature selection for intrusion detection system. *Int. J. Adv. Sci. Res. Eng.* **2018**, *4*, 129–134. [CrossRef]