

Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Anbar
College of Computer Science and Information Technology
Department of Computer Science



Intelligent Intrusion Detection System in Internal Communication Systems for Driverless Cars

**A Thesis Submitted to the Department of Computer Science,
College of Computer Science and Information Technology,
University of Anbar as a Partial Fulfillment of the Requirements
for Master Degree of Science in Computer Science**

By

Nuha Abdulla Hamad

Supervised By

Assist. Prof. Dr. Khattab M. Ali Alheeti

And

Assist. Prof. Dr. Salah Sleibi Al-Rawi

1442 A.H.

2020 A.D

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

﴿وَيَرَى الَّذِينَ أُوتُوا الْعِلْمَ الَّذِي أَنْزَلَ إِلَيْكَ

مِنْ رَبِّكَ هُوَ الْحَقُّ وَيَهْدِي إِلَى صِرَاطٍ الْعَزِيزِ الْحَمِيدِ﴾

صدق الله العظيم

سورة سبأ الآية (٦)

اسم الطالبة: نهى عبدالله حمد

الكلية : علوم الحاسوب وتكنولوجيا المعلومات - قسم علوم الحاسبات

عنوان الرسالة: نظام كشف التسلل الذكي في أنظمة الاتصالات الداخلية
للسيارات بدون سائق.

طبقا لقانون حماية المؤلف رقم 3 لسنة 1971 المعدل العراقي فإن للمؤلف حق منع أي حذف أو تغيير للرسالة أو الاطروحة بعد اقرارها و هي الحقوق الخاصة بالمؤلف وحده والتي لا يجوز الاعتداء عليها. فلا يحق لاحد ان يقرر نشر مصنف أحجم مؤلفه عن نشره او اعادة نشر مؤلف لم يقر مؤلفه بذلك، فإذا قام بذلك اعتبر عمله غير مشروع لأنه استعمل سلطة لا يملكها قانونا.

Supervisors' Certification

*We certify that we read this thesis entitled “**Intelligent Intrusion Detection System in Internal Communication Systems for Driverless Cars**” that is carried out under my supervision at the Department of Computer Science of the University of Anbar, by the student “**Nuha Abdulla Hamad**”, in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.*

Signature:

*Name : **Dr. Khattab M. Ali Alheeti***

*(**Supervisor I**)*

Date : / /2020

Signature:

*Name : **Dr. Salah Sleibi Al-Rawi***

*(**Supervisor II**)*

Date : / /2020

Certification of the Examination Committee

We the examination committee certify that we have read this thesis entitled " Intelligent Intrusion Detection System in Internal Communication Systems for Driverless Cars " and have examined the student " Nuha Abdulla Hamad ", in its contents and what is related to it, and that in our option it is adequate to fulfill the requirements for the degree of Master of Computer Science.

Signature:

Name: Assist.Prof.Dr. Mustafa Dhiaa Al-Hassani (Chairman)

Date: / /2020

Signature:

Name: Prof. Dr. Salah Awad Salman (Member)

Date: / /2020

Signature:

Name: Lecturer.Prof.Dr. Foad Salem Mubarek (Member)

Date: / /2020

Signature:

Name: Assist.Prof.Dr. Khattab M. Ali Alheeti (SupervisorI)

Date: / /2020

Signature:

Name: Assist.Prof.Dr. Salah Sleibi Al-Rawi (SupervisorII)

Date: / /2020

Approved by the Dean of the College of Computer Science and Information Technology, University of Anbar.

Signature:

Name: Prof. Dr. Salah Awad Salman (Dean of the College)

Date: / /2020

Student name: Nuha Abdulla Hamad

Thesis title: Intelligent Intrusion Detection System in Internal Communication Systems for Driverless Cars

Abstract

The modern car is a complicated system consisting of Electronic Control Units (ECUs) with engines, detectors, and wired and wireless communication protocols, that communicate through different types of intra-car networks. The cyber-physical design relies on this ECU network that have been evidenced to be susceptible to attacks, by security researchers through physical and remote access to the cars' internal network. The internal network contains several security vulnerabilities that make it possible to launch attacks via buses and propagation over the entire ECU network, for example preventing the engine to work or cutting the brakes by injection fabricated messages. therefore, anomaly detection technology, which represents the security protection, can efficiently reduce security threats.

This work proposes anomaly Intrusion Detection System (IDS) using the Artificial Neural Network (ANN) to monitor the state of the car by packets collected from internal buses and to achieve security of the internal network through training the CAN packet to devise the fundamental statistical feature of normal and attacking packets and in defense, extracted the related attribute to classify the attack. Generating new features and using the K-means clustering algorithm to differentiate similar samples and addressing them as a single class to improve the accuracy of the model. Features are evaluated to measure its discrimination ability between classes and to select the best existing features.

The experimental study has performed using two sets of data, Simulated data Open Car Test-Bed and Network Experiments (OCTANE) and Real dataset to evaluate our detection system. The experimental results on (OCTANE) demonstrate that the IDS has achieved good performance with a false-positive rate of 1.7%, a false-negative rate of 24.6%, and average accuracy of 92.1%. Experimental evaluation on a real dataset shows that the proposed system has a low false-negative rate of 0.1% and an error rate of 0.006 with an average accuracy of 87.63%.

Acknowledgments

First of all, thanks to Almighty God for guidance, protection, and skills for enabling me to complete my study.

I would like to express my thanks and gratitude to my supervisor Assist Prof. Dr. Khattab M. Ali Alheeti for giving time and valuable guidance, instructions, and continuous advice.

I would like to express my appreciation and gratitude to Assist Prof. Dr. Salah Sleibi Al-Rawi for understanding, valuable advice and suggestions, and encouragement.

I would like to thanks the Dean of the College of Computer Science and Information Technology Assist. Prof. Dr. Salah Awad Salman.

I wish to offer my gratitude to president of department of the college of computer in university of Anbar Assist. Prof. Dr. Wesam Mohammed alrawi.

I am grateful to Dr. Ruqayah Rabeea Al-Dahhan to her help.

I would like to thanks all staff of the college of computer in university of Anbar who offering their help.

Special thanks to my husband who encouraged me to fly toward my dreams.

My deepest thanks to members of my family for their patience, and encouragement.

Special thanks to all my friends for their love, support, who gave me help during my study.

*Nuha Abdulla
2020*

Dedication

To everyone who works on the renaissance of the nation, to every educator who knows the value of his mission and performs the trust of building a better tomorrow than ours. Building a generation capable of making the right decisions will restore the nation to its glory. It is a great honesty, to which the hearts of the sincerely fall.

To my parents whose have been a source of inspiration, and have raised me to be the person I am today.

To my caring, loving, and supportive husband, Abdul Hakeem. Your encouragement when the times got rough this means a lot to me.

To my brothers, sisters, (especially Sumaya) who shared their words of advice and encouragement to finish this study

To my children who allowed me time away from them to research and write.

*Nuha Abdulla
2020*

List of Abbreviations

<i>Abbreviations</i>	<i>Meaning</i>
ANN	Artificial Neural Network
AN	Artificial Neuron
CAN	Controller Area Network
CTI	Collaborative Trust Index
DoS	Denial of Service
DNN	Deep Neural Network
DBN	Deep Belief Networks
DIDS	Distributed Intrusion Detection System
DT	Decision Tree
DLC	Data Length Code
DCNN	Deep Convolution Neural Network
ECU	Electronic Control Unit
EOF	End of the Frame
ER	Error Rate
FFDNN	Feed Forward Deep Neural Networks
FEU	Feature Extraction Unit
FN	False-Negative
GCDC	Grand Cooperative Driving Challenge
HIDS	Host- Intrusion Detection Systems
HTM	Hierarchical Temporal Memory
IDS	Intrusion Detection System /Sensor
IDPS	Intrusion Detection and Prevention Systems
KNN	K-Nearest Neighbor
LIN	Local Interconnect Network
MCM	Markov Chain Model
MOST	Media Oriented Systems Transport
NB	Naive Bayes
NIDS	Network- Intrusion Detection Systems
OBD	On-Board Diagnostics
OCTANE	Open Car Test-Bed and Network Experiments
RF	Random Forest
RTR	Remote Transmission Request
RNN	Recurrent Neural Networks
SVM	Support Vector Machines
SAM	Survival Analysis Model
TP	True-Positive
V2V	Vehicle- to- Vehicle

Contents

<i>Chapter One General Introduction</i>	
1.1 Overview	1
1.2 Related Work	4
1.3 Problem Statement	8
1.4 Motivation	8
1.5 Research Objectives	9
1.6 Contribution	9
1.7 Thesis structure	10
<i>Chapter Two Theoretical Background</i>	
2.1 Introduction	11
2.2 Self-driving Car	12
2.3 Communication system for self-driving vehicles	13
2.3.1 External Communication system	13
2.3.2 Internal Communication system	14
2.4 CAN network	17
2.5 Attack Types	18
2.5.1 Denial of Service Attack (DoS)	19
2.5.2 Message Injection and Replay Attack	19
2.5.3 Message Manipulation	20
2.5.4 Masquerade Attack	20
2.5.5 Malware Attack	20
2.6 Internal Communication System	20
2.6.1 Local access	21
2.6.2 Remote access	22
2.7 Security Technique	24
2.8 Intrusion Detection System	25

2.9 Type of Intrusion Detection Systems	26
2.9.1 Host-Based Intrusion Detection System (HIDS)	27
2.9.2 Network-Based Intrusion Detection (NIDS)	27
2.9.3 Hybrid-Based Intrusion Detection	28
2.10 Classification of Intrusion Detection System	29
2.10.1 Anomaly-based IDS	29
2.10.2 The signature-based IDS	29
2.10.3 Specification-based IDS	30
2.10.4 Hybrid-based approach	30
2.11 Feature Clustering	31
2.12 Feature Selection	32
2.12.1 Feature Selection methods	33
2.13 Limitation	34
<i>Chapter Three-The Proposed System Design and Implementation</i>	
3.1 Introduction	35
3.2 Intrusion Detection system architecture	36
3.2.1 Data collection	37
3.2.2 Feature sets	38
3.2.3 Data Preprocessing	38
3.2.4 Feature generation	39
3.2.5 Feature Clustering for real data	39
3.2.6 Normalization	40
3.2.7 Feature Evaluation for real data	40
3.2.8 Feature Selection for real data	41
3.3 The ANN Architecture	43
3.3.1 Real dataset	43
3.3.2 Simulation data	44
3.4 ANN Performance Steps	46

3.4.1 For Simulation	46
3.4.2 For Real Data	47
<i>Chapter Four – Results and Discussion</i>	
4.1 Introduction	53
4.2 Datasets for real data	54
4.3 Software –OCTANE (Open Car Testbed and Network Experiments)	54
4.4 Performance Evaluation Criteria for IDS	56
4.5 Evaluation performance of proposed system	57
4.6 Comparison with previous works	60
<i>Conclusion and Future Works</i>	
5.1 Conclusion	64
5.2 Future Works	65
References	66
Appendix A	A-1
Appendix B	B-1

List of Tables

<i>Table No.</i>	<i>Description</i>	<i>Page No.</i>
3.1	Sample of simulation data	37
3.2	Feature type of CAN	38
3.3	Artificial Neural Networks (ANN) Parameters	46
4.1	Overview of datasets	54
4.2	Feature evaluation result of real data	58
4.3	Feature selection result for real data	58
4.4	Classification results for real data of the proposed IDS	59
4.5	Classification accuracy and number of records of features used in the simulation.	59
4.6	Recognition Rate for simulation	60
4.7	Comparison with other algorithms	60

List of Figures

Figure No.	Description	Page No.
1.1	Communication of self-driving vehicles	2
2.1	Vehicle-to-Infrastructure (V2I) communication	13
2.2	Internal Communication Protocols in self driving cars	14
2.3	Car Network	16
2.4	CAN frame format	17
2.5	Example of attack in the internal car network	19
2.6	Typical scenarios for a message injection attack	21
2.7	Remote Communication	22
2.8	Classification of IDS based on used Methodology	26
2.9	Host-based IDS	27
2.10	Network-based IDS	28
2.11	The flow chart of the K_Means algorithm is applied with K=2	32
3.1	Block diagram of the Proposed System	36
3.2	Real Dataset Preparing	42
3.3	Structure of the Artificial Neural Network for real data	44
3.4	Structure of the Artificial Neural Network of simulation data	45
3.5	ANN Implementation for the proposed system	47
4.1	OCTANE (Bus Monitor)	55
4.2	OCTANE (Transmit Packet)	55
4.3	Performance Evaluation Comparison between proposed system and previous study	62
4.4	Performance Comparison between simulation data	62

Chapter one

Introduction

Chapter One

Introduction

1.1 Overview

Recently, development in the automotive system has become evident through the integration and use a set of computing devices Electronic Control Unit (ECU) that has replaced the traditional mechanical control part. These devices control and monitor subsystems to improve energy efficiency and reduce noise and shaking[1]. There are more than 70 ECU in today's car, each of which interacts with others through a bus – topology in the car network. Newly communication devices are used in vehicle networking services, essential for vehicle-to-vehicle communication and perform communication within vehicles also used to control important functions to the safety and efficiency of automobiles, such as engine control and braking [2][3]. These communication tools serve as an interface between the in-vehicle network and the out-vehicle network [4].

Effective fuel consumption is considered when calculating the speeds or distances of the connected cars. The cooperative strategy can control speed of car and it is equipped to transmit signals of velocity changes with memory from several cars by means of wireless vehicle-to-vehicle communication to improve safety and fuel efficiency [5][6]. Tang et al., suggested using communication to gain a better understanding of driving behaviors such as velocity and fuel consumption per vehicle[7][8]. Another researcher, showed strong vehicle-to-vehicle (V2V) connections based on traffic flow[9]. Moreover, Kesting et al. were able to progress a new plan of their search to push messages through communication [10].

In addition, wireless communications that enabled a collaborative platoon can also be used to obtain better traffic flow [11]. The best performing results in the Grand Cooperative Driving Challenge (GCDC) illustrated the latest inventions in the realistic cooperative driving domains [12][13]. Therefore, the capacity of a vehicle's computing systems improved dramatically.

To promote communications, various protocols have been developed. One of the most popular protocols is a network interface, Controller Area Network

Chapter one: Introduction

(CAN), a simple protocol that helps link sensors and motors to ECU as a standard for in-vehicle network communication, used to provide an effective interface between electronic control units, does not have a protection mechanism for detecting improper packets. CAN bus has been commonly used due to its low wiring costs, weight, and complexity [3].

However, because of its simplicity and flexibility, there are some weaknesses to the CAN bus. When the attacker exploits the weakness of the CAN protocol and can control cars network by injecting a malicious message, and is able to monitor the other ECU, thus it triggers an incorrect operation in cars. Attackers usually gain access via the on-board diagnostic (OBD-II) port or through an external interface (wireless communications) such as cellular, Bluetooth, Wi-Fi [14]. The adoption of CAN promotes the development of automotive applications[15]. In most cases, from a CAN bus, necessary information like infotainment, diagnostic, and controlling data are gotten to help the car services, such as driverless vehicles shown in Figure 1.1 [16].

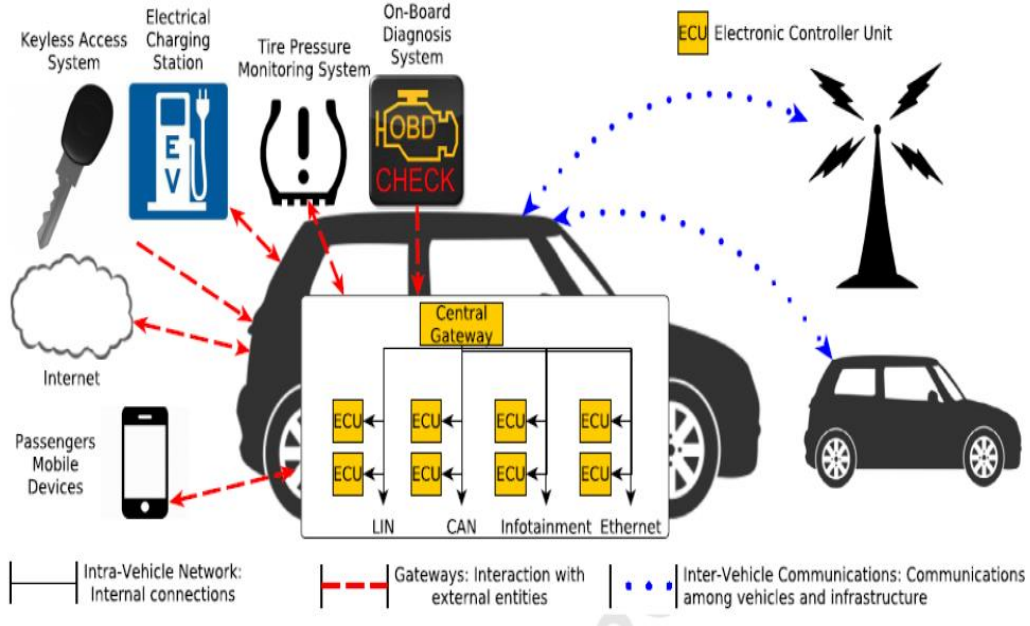


Figure 1.1 communication of self-driving vehicles[16].

However, information must be safe, but the rapid growth of networks has led to challenges. ECU can get messages from the same bus by broadcasting messages ECU-to-ECU without knowing the sender's identity [17]. There is no mechanism

Chapter one: Introduction

for authentication and message encryption involved, to protect the network from attacks. The lack of protection mechanisms in CAN bus opens the way for the attacker to access goals by injection fabrication message.

A lot of research works have considered safety issues for communication between and within cars [17][18]. Specifically, an Intrusion Detection System “IDS” achieved much attention because of the simplicity and efficiency in identifying the damage occurred [17][20].

Previous methods of intrusion detection can only be efficient for specific attack systems already considered in design phases[21][22].

Consequently, machine learning-based IDS strategies are used mainly for conventional communication networks to deal with the problem[23]. The aim is to collect the basic features of the data and use them to detect any hacking attacks[24]. Methods for detecting intrusions have been developed by using the ANN. ANN is effective in the classification of statistical patterns, security as well as in intelligent vehicular systems [25][26].and support vector machine to classify the type of attacks[27].

In this thesis, the anomaly detection system is proposed to secure internal communication systems for self-driving and semi self-driving vehicles. However, it is heavily based on ANN. Thus, intelligent detection scheme plays a vital role to protect information/ control data between ECUs. In addition, the suggested methodology trains CAN packet information to work out the fundamental statistical properties of normal packets and attack packets and, in defense, extracts the related attribute to classify the attacks.

1.2 Related Works

Although there have been many efforts to build, develop effective detection systems, there is still an open area due to various aspects in this area such as a highly dynamic environment, large traffic data size, ambiguous boundaries between normal and abnormal behavior.

Intrusion Detection techniques have been studied and developed against malicious attacks. Using these techniques, the activity of the system is monitored and classified depending on specific rules, methods, and algorithms. This part offers an account of past researches on intrusion detection systems using different techniques. The next researches effort focused either on external or internal security. In our work, we try to interest with internal security.

Larson et al. (2008) [28] proposed a specification-based method of detection of attacks by creating a set of security specifications from information extracted from the CAN v2.0 and CANopen v3.01 specifications. And considered the message an attack unless followed the protocol level or ECU behavior. The applicability of the detector has evaluated by deriving an attacker model and shown that most attack actions readily can be detected.

Neeraj et al. (2014) [29] proposed a collaborative learning based IDS using the Markov Chain Model (MCM) to build and represent states and their transitions through the network. Collaborative Trust Index (CTI) can detect any kind of attack. Also proposed an algorithm to detect anomalous events using the specified classifier. The results show that IDS is able to detect the intrusion in more than 90% and perform better than other in relation to parameters such as false alarm ratio, detection ratio.

Taylor et al. (2015)[30] presented a method based on measuring the time between CAN message data. They used frequency and average times to evaluates the parameters of CAN's message. The experience demonstrated the efficiency of detect anomalies with high-confidence.

Kang M-J, Kang J-W (2016) [31] used intrusion detection by methods of a Deep Neural Network (DNN) for the safety of the vehicle network. DNN was shown to be efficient in the classification of statistical patterns. The parameter train

Chapter one: Introduction

efficiently by initializing the deep neural network depends on the extraction of feature vehicular network packet used in training and testing to reach a good rating through the unsupervised pre-training of Deep Belief Networks (DBN) to construct a classifier and tested it on a simulated dataset to detect normal and hacking packets and thus identify a malicious attack on the cars.

Adrian Taylor et al. (2016)[32] suggested an anomaly detector based on long short-term memory using a neural network to detect anomalies of the CAN network. This detection system works by training to assess the next value of CAN message created by each sender on the bus, and used the error as the detection of an attack, if the next message has a higher bit error than the normal domain, it may be considered as an anomaly.

Mabrouka et al. (2016) [33] easy intrusion detection technique IDS has been suggested to handle the safety issue in the CAN bus by mechanism-based on the calculation of time intervals for CAN messages by checks the CAN ID of the transmitted message .The main benefit of the approach is that it is not important to change the hardware layer and to implement each ECU.

Song et al. (2016) [34] presented a flexible intrusion detection algorithm for in-vehicle networks based on the evaluation of time intervals for CAN messages. CAN messages from the cars were captured and three types of message injection attacks were carried out. Therefore, the time interval found to be a significant characteristic for detecting assaults in CAN traffic. The intrusion detection system also recognizes all injection assaults without making false-positive mistakes and can detect word injection attacks effectively in a millisecond.

Markovitz and Wool (2017) [35] discussed a method for detecting malicious messages by inspection of the real CAN bus. The authors demonstrated the messages divided into four fields. This field is Multi-Value fields, Constant, Counter, and Sensor fields. The researchers constructed a classifier that recognizes fields and types of those fields automatically. The system evaluated on simulated CAN bus traffic with a median false positive rate of 1%.

H. Lee et al. (2017) [36] suggested an intrusion detection technique based on CAN offset ratio and time interval analysis between request and reaction messages with rapid detection of intrusion with high precision. The message

Chapter one: Introduction

injection attack and impersonating node attack can be effectively detected by Offset Ratio and Time Interval based Intrusion Detection System (OTIDS), which consider the most hazardous vehicle attack.

Shadi et al. (2018) [37] discussed an anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. Intrusion detection systems require a set of data. The NSL-KDD dataset was used for a clearer understanding of the analyzed data as well as for testing and simulating IDS performance. A hybrid approach with a vote algorithm was used to identify important features and filter data to be an accurate model. This hybrid approach also addressed high false-negative and low false rates. As a result, this approach has an important impact on reduced detection time and achieve a good accuracy rate.

Kuwahara. et al. (2018) [38]discussed the applicability of statistical approach (anomaly detection system) for identifying malicious messages in -cars networks through formalizing features, they focused on the number of messages observed in a fixed time window to illustrate a message sequence that includes the number of messages related with each receiver ID. The author collected CAN message data from the real car. The result of the experiment proved the fast and effective detection achieved by their methods.

Alex et al. (2018) [39] suggested an intelligent detection systems using artificial neural networks for the recognition of shellcode patterns in the network (identifying all instances of shellcode). ANN was able to differentiate between normal and malicious network traffic precisely for use in deep packet test-based intrusion detection systems. The classification technique used is powerful and accurate, plus the minimal numbers of false positive less than 2% in repeated 10-fold cross-validation.

Mee Lan. et al. (2018) [40] suggested the detection of car network intrusion using the Survival Analysis Model (SAM). The primary goal was to identify malicious CAN messages and detect normal and abnormal messages for the network without knowing CAN IDs. An unknown attack can be detected, and this does not require a regular update of signatures for attack patterns as opposed to the abuse detection method. Analyzes data, collect targets inside the vehicle. Unstable power and low controller performance made it difficult to implement network

Chapter one: Introduction

security solutions within vehicles. The important attributes for real-time processing will be high detection accuracy and low computational costs.

Priyanka et al. (2018) [41] provided a new approach for intrusion detection and prevention systems (IDPSs) using ANN for connected cars and using deep-learning anomaly detection algorithms as part of intrusion detection and adaptation to recognize the rapid change in data transfer. The characteristic of IDPS was real-time detection and less false-negative rate.

Wang et al. (2018) [42] focused on Controller Area Network (CAN) security, which is an ECU communication standard. Suggested a new IDS based on CAN messages entropy of identifier bits. CAN injection assaults need to change the CAN ID bits and analyzing such bits ' entropy can be an efficient way to detect attacks. The system limited the injection of the number of harmful messages with larger priority IDs by attackers. Data are collected from real a vehicle (2016 Ford Fusion) and perform message injection attacks. The experimental findings showed that entropy-based IDS is capable of effectively detecting all the injection assaults without interrupting CAN communication.

Sydney et al. (2019) [43] suggested a Deep Learning (DL) IDS using Feed Forward Deep Neural Networks (FFDNN) and a filter-based selection algorithm and provided FFDNNs to design, implement and test a DL-based intrusion detection scheme. The FFDNN models used in this research were combined with a Feature Extraction Unit (FEU) using information obtained to reduce the input dimension while increasing the classifier's accuracy. The performance of the FFDNNs models with both full and FEU-reduced feature space is superior to Support Vector Machines (SVM), Random Forest(RF), Naive Bayes (NB), Decision Tree (DT) and K-Nearest Neighbor(KNN).

In this thesis, the anomaly intrusion detection system based on ANN is used to achieve a more secure and reliable internal communication of the cars.

1.3 Problem Statement

Recently, the problem of computer system intrusions has increased, and intruders have become flexible sources. The reason is that the use of electronic device in modern vehicles has increased the number of attacks targeting in-vehicle networks. Car network security is a major concern and CAN that is used as a basic standard for in-vehicle networks, does not have enough security.

CAN is a broadcast-based bus network and the receiving node does not verify the origin of a CAN message, there is no authentication, any node can be linked to the bus and all messages can be received. Many injection attacks on network traffic are possible. The intruder can also access data on a CAN bus easily. The access to internal data enables the attacker to analyze the CAN data of the intended vehicle and use the message injection attacks to monitor the target vehicle.

This problem can be resolve by using Intrusion detection (ID) is convenient software, an attractive concept that can deal with the complexity of computer systems and protect them from attacks.

Accordingly, we proposed an intelligent detection model using Artificial neural networks to distinguish between abnormal behavior (intrusion) and normal behavior based on features that will be collected from CAN bus.

1.4 Motivation

Modern cars' internal networks lack security mechanisms; therefore, it is susceptible to attacks. An intruder can be exploiting vulnerabilities in the car' controllers to penetrate the system. There are several reasons that motivate to used intrusion detection system as shown in the following:

- The CAN bus is the most common internal car network communication protocol and lacks enough security features, such as message encryption and sender authentication, to secure the network against cyber-attacks.
- Embedded interfaces in new cars could allow wireless communication such as Bluetooth, Wi-Fi, and wired connection (USB) to connect to the outside world. So, there are some security threats that fight their way to infiltrate the system.

Chapter one: Introduction

1.5 Research Objectives

The main aim is to detect intrusions into the CAN bus. assume the attacker has been already gained access to the CAN bus, for example by attacking one of the connected ECUs. An intruder is now attempting to manipulate messages to impacts vehicle behavior. Our purpose is to detect signals that deviate from their normal behavior by present a new intrusion detection system that can reduce security risks in internal communications of self-driving cars and protect from the potential attacks.

In order to fulfill the main objective of the study the following sub-objectives are defined

- 1) Studying the IDS system design and understanding the behavior of CAN bus in-car network and analyze its vulnerabilities.
- 2) To presented various potential attacks that could be deployed against the CAN bus system.
- 3) To implement secure internal communication, using an Artificial Neural Network(ANN) in self-driving cars to protect from potential attacks.
- 4) Train and test data to measure the accuracy of the model.

1.6 Contributions

The followings are the main contributions of this thesis:

- 1) **Enhancing the space (Generate new features)**, creating new input from existing features to improve the accuracy of the model.
- 2) **Feature Clustering using (k-means) algorithm** for identifying similar samples and considering them as a single class, labeled the dataset to a normal class and abnormal class.
- 3) **Evaluation the features** by measuring the discrimination ability of each feature intra-class (in a class) and inter-class (between class discrimination).
- 4) **Feature Selection**, select the best features to improve computational efficiency, enhancing classification accuracy by eliminating irrelevant feature.

Chapter one: Introduction

1.7 Thesis structure

This thesis is divided into five chapters. In addition to this chapter, briefly stated as follows:

Chapter Two: presents more details about the concept of the intrusion detection system, controller area networks such as characteristics, challenges, and attack types.

Chapter Three: this chapter explains the proposed system and all its details of the design and system requirements to implement this system.

Chapter Four: contains the results that are obtained through the proposed system and the discussion and analysis of the results that are obtained through this work.

Chapter Five: includes the work conclusions and recommendations for future work.

Chapter Two
Theoretical Background

Chapter Two

Theoretical Background

2.1. Introduction

Modern technology becomes an integral part of our daily lives where it uses have many aspects. One aspect of this technology is autonomous systems. These systems play an effective role in the development and applicability of modern technology [44]. Such technologies, for example, autonomous vehicles and robotics, have a huge contribution to improving society's quality of life in various areas including scientific research, warfare, intelligence, and recognition. Moreover, autonomous vehicles certainly compete largely in the developed country markets and bring a radical change in the way they are driven in the streets. They also have a great impact on reducing gas emissions, traffic, maximum collisions, and saving time and energy with better safety [45].

Generally, these systems have many positive traits however, their security is still a concern for their users to protect them from any kind of attack. The car network system can be accessed through its wireless network and attackers can implement number of attacks, such as replay, DoS, frame injection attacks [17]. Therefore, the current study is interested to improve the security of cars' networks to avoid attackers' access to their systems [46].

This chapter explain intrusion detection systems and its types, and introduces the concept of self-driving systems, Internal/ external communication of driverless cars, and the types of attacks on the car's networks. More details about the types of internal communication protocol and describes security issues that emerge in internal car networks.

Chapter two: Theoretical Background

2.2 Self-driving Car

Autonomous vehicles or as sometimes called self-driving cars are those cars which designed for humans to have self-controlling systems without any human supervision or intervention. They are driven in unstable, unpredictable, and open traffic environments [47].

There are many applications of autonomous systems. One well-known application is driverless cars. This application is famous for being a brand-new revolution of the automotive industry and considered the most significant application of autonomous systems. More importantly, such cars have many positive impacts on both users and society. They provide maximum safety to users in reducing the number of potential accidents, road congestions. They have many economic benefits in terms of energy and applying what is called platoon behaviour in narrow roads [48]. In addition to that, these vehicles have an important role in improving transport methods to avoid accidents made by human errors [49]. Therefore, driverless cars are seen to have revolutionized the methods of transportation [50].

Vehicles that use self-driving systems are highly dependent on communication systems, both internal and external; to check the environment they drive-through and also to anticipate the events. These cars are fitted with an array of integrated network sensors and equipped with On-Board Diagnostics (OBD) for communication [51].

Generally, the self-driving vehicle usually contains various information systems (software) and hardware, however, the important ones to consider are the ECUs or also known as nodes. It is the unit that is responsible for the computation in the intra-vehicular network. It is used in an autonomous vehicle with multiple purposes such as controlling the vehicle in general[52].

Such a system is programmed to do multiple functions, for instance, to accelerate, slow down, halt, steer, etc. Moreover, they also control the airbags, doors, and audio in a vehicle. Simply, each ECU responds to something that requires control. The ECU uses CAN communicate with each other.

However, it is worth mentioning that there are many units of them in a single-vehicle. More importantly then, it is wise to keep the ECUs of a vehicle secure from both physical and remote access attacks [53].

Chapter two: Theoretical Background

Multiple ECUs in car form the intra-vehicle network by interconnection and consequently make them liable to attacks. As such, there is a need for security measures and this can be done by considering how these parts work, what parts they communicate with.

2.3 Communication system for self-driving vehicles

The communication system of the car plays an important role in its safety and security. It helps in detecting the intermediate threat in the cars' surroundings on roads. It is argued that when it comes to autonomous vehicles, anything that is transmitted is at risk. Any information that is transmitted inside or outside the vehicle can be hacked or compromised. The fact that this study is concerned with security, it is wise to mention safety as it is a necessity for intact communication in vehicles. Moreover, there is a direct relation between communication and safety in implementation and function. Generally, there are two main types of communication systems: the internal and external infrastructure of the communication system [54].

2.3.1 External Communication system

To communicate with external vehicles, autonomous vehicles must be provided with wireless connectivity such as broadband communications to gain access to external resources and share information with other vehicles. Such communications, for example, allow passengers to connect with others and reach local services such as tourist information, traffic updates and gas situation [55].

The following Figure 2.1 shows the external communication scenarios in autonomous vehicles.

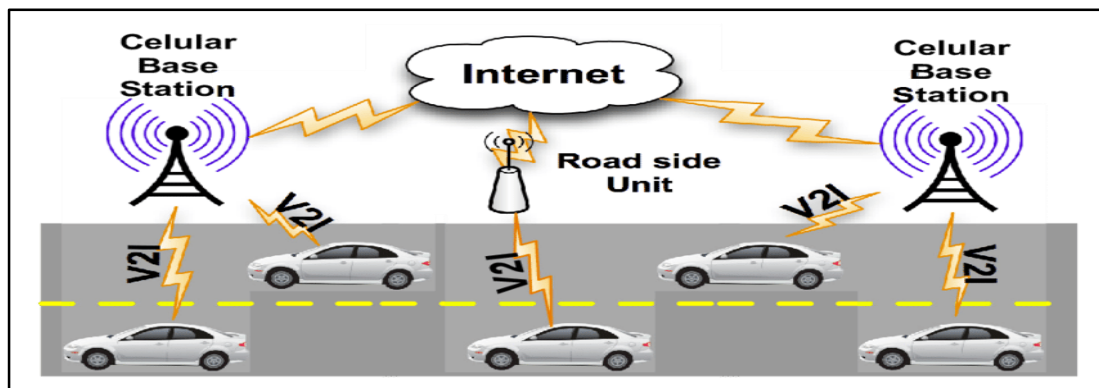


Figure 2.1 Vehicle-to-Infrastructure (V2I) communication [56].

Chapter two: Theoretical Background

2.3.2 Internal Communication system

In an automotive area, there is a range of different vehicle communication systems already in use. These applications vary on their immediate purpose. Such applications range from multiple driving assistant, electronic engine control, safety features, and different applications for infotainment. An example of a typical modern car generally incorporates a range of networked components comprising of actuators, sensors, ECUs, and intra-vehicle network devices for communication [57]. The intra-vehicle network works for data sharing among actuators, sensors, and ECUs to make the car run its engine. However, it is difficult to connect two electronic engine control units directly with other controlling units in cars because additional costs and space are required for the quantity of wire in a single car. Therefore, several ECUs are connected to a bus, and the bus transmits some messages to all associated nodes. As such, several subnetworks of a car are mainly interconnected via ECU gateways [58].

The following different vehicle communication types are classified according to their technical features and field of application. Mainly, each network of the five widely used types of intra-vehicle networks in intra-vehicle communication systems has its advantages and limitations [59]. These five networks include the following types: Control Area Network (CAN), Local Interconnect Network (LIN), Flex Ray, Media Oriented Systems Transport (MOST), and Ethernet are shown in Figure 2.2

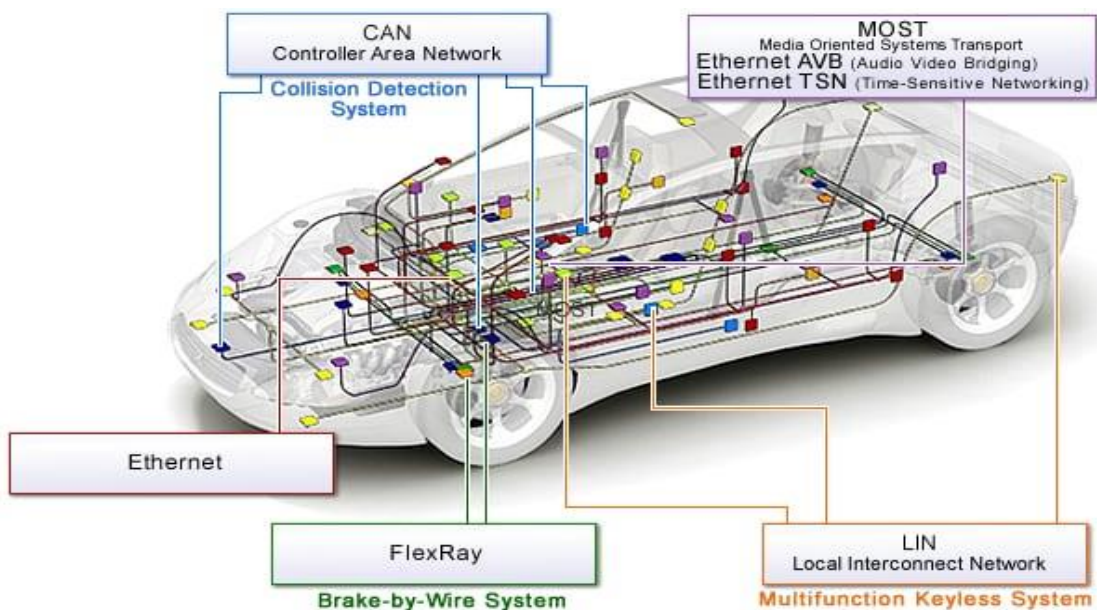


Figure 2.2 Internal Communication Protocols in self driving cars [60].

Chapter two: Theoretical Background

1. Control Area Network (CAN)

This network provides up to one Mbps of data rate. It is a serial bus designed for cars use. CAN is known as the most common communication system protocol for the internal vehicle network. Its multi-master architecture, which are able to operate even if some nodes are defect. CAN is used to transmit messages through the ECU to the network as data packets. CAN protocol solve one of the major problems in cars by link different ECUs thus, reduce complexity, cost and the weight of the car. CAN uses reliable, priority driven CSMA/CD (Carrier Sense Multiple Access / Collision Detection) access control method to guarantee every time the transmission of the top priority message always first. However, it is vulnerable to potential threats [57] [61] .

2. Local Interconnect Network (LIN)

This network uses serial communication between actuators and smart sensors. It transmits data rates with up to 20Kbit/s. Generally, it is a single-wired sub network for low-cost. It is meant to be used everywhere in a car from 2001, where there is no need for a CAN networks' bandwidth. This type of network usually uses the principle of master-slave. Upon order from the master, messages are sent from the slave at a speed of 20 kbps. The data transmission of this network is low, yet, its protocol is considered cheap where it has low tolerance for faults. This network offers a low speed of communication, which is ideal for applications that do not require strict time efficiency performance, such as battery surveillance management, window lifter control, etc. The LIN network, which is usually used within the CAN network as a sub-network therefore, usually used for controlling a cars comfort element , such as power- windows, side mirrors modification, and also the lighting [62].

3. Flex Ray

This network is used to promote faster and more efficient communication than CAN. It has a high-speed bus that is deterministic, invariant, and error tolerant. Generally, it supports two-channel communication rather than one-channel as in CAN network. Offering maximum speed up to 10Mbps. Specifically, this network is used only in critical function because it has a higher security threat and system cost [63].

Chapter two: Theoretical Background

4. Media Oriented System Transport (MOST)

This network is more expensive. It is used carry multimedia data into the car for transmitting video, voice through fiber optic cable with a high-speed communication up to 24 Mbps and synchronous, asynchronous data transmission. An internal MOST system usually handles the error management, detecting errors over parity bits, checksums, status flags and disconnect erroneous if needed.

5. Ethernet

Finally, the use of Ethernet in internal network is being considered in near future.

Generally, vehicles currently are networked and connected wirelessly with the outside world, other vehicles, or/and infrastructure. Vehicles' internal networks are complemented with external networks to carry out tasks. Therefore, electronic attacks cannot be done if the car is a closed network, an attacker can only perform an attack by cutting the wires in the vehicle. However, if the car is open to the outside world through networking, it can be reached and hacked from a remote computer.

Moreover, modern vehicle must also have many networks placed in them to provide communication between the ECUs. This number of combinations is used to address different network needs. For instance, infotainment systems require a higher bandwidth while other systems need fault-tolerant networks. Therefore, as shown in Figure 2.3 below, several network topologies are used in cars[64].

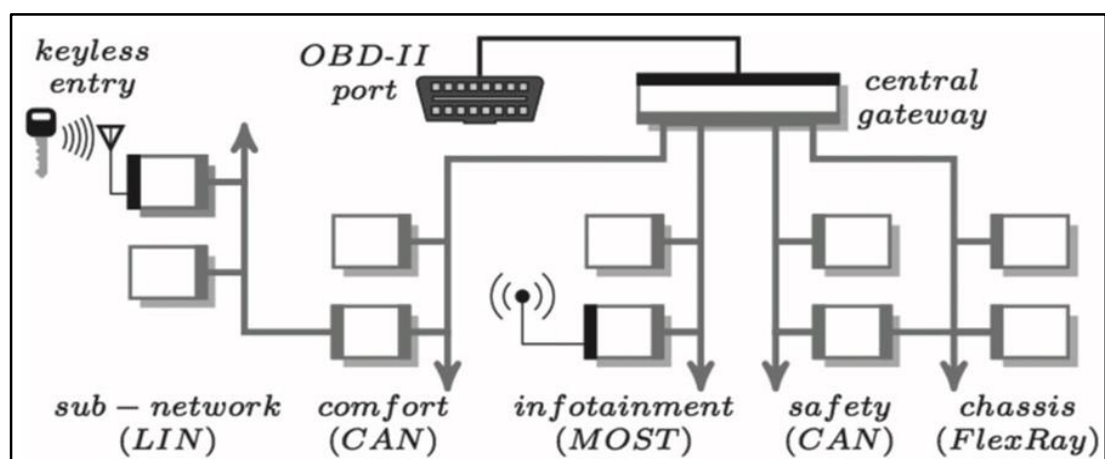


Figure 2.3 Car Network [64].

Chapter two: Theoretical Background

2.4 CAN network

It is essential to know how the internal system works to understand its vulnerability. Therefore, this section is mainly introduced to describe the CAN's protocol structure.

- **CAN data frame structure**

Autonomous cars use CAN bus topologies, CAN bus network is a broadcast protocol designed to handle the internal communications among the electronic control units, for example, devices and actuators, sensors, its permit the communication with each other. CAN is adopted and used widely for in-vehicle networks because it is reliable and can reduce wiring cost, weight, and complexity. It also offers an effective error detection mechanism (stable in transmission and speed in response)[65].

Moreover, the frame in CAN packet has a structure and its carriers sequence of CAN data (e.g. bytes) in the network. The packet's priority is indicated by the CAN frame transmitted in the field of the arbitrariness identifier (ID). The higher priority of the packet is signified by the lower the ID bit value. As such, a collision in the CAN bus traffic is avoided by this protocol. Generally, the standard structure of a frame normally contains the following fields: data, acknowledge, arbitration identifier, and a few others. In the current study, both the CAN ID data frame and the data fields are the only ones. The bit value of the CAN packet ID field is normally 11 bit or can be expanded to 29 bits. Although the CAN data field comprises 0 to 64 bits of data [66].

The meaning of each field in the CAN frame is described in the following Figure 2.4:

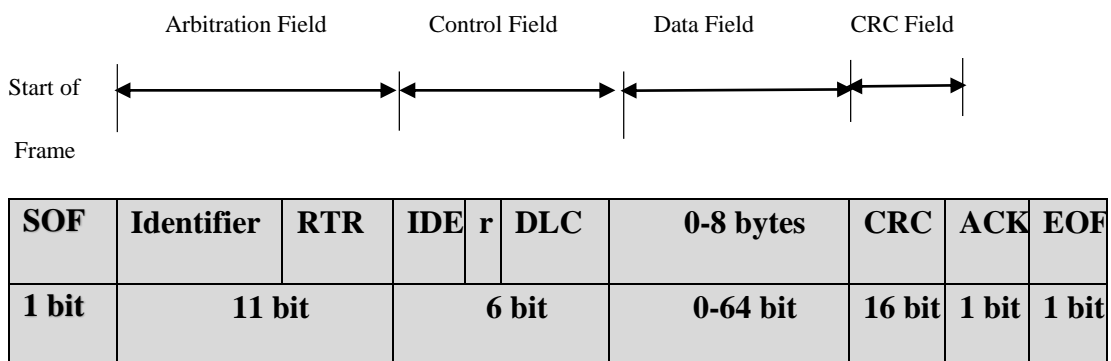


Figure 2.4 CAN frame format.

Chapter two: Theoretical Background

A detailed description of the abbreviations used in Figure 2.4 is given in the following items:

- a. **Start of the frame: (SOF)** (1 bit) this bit signifies the beginning of a new message and it is used to synchronize the nodes of the bus.
- b. **Arbitration:** This consists of two main components; the identifier field that defines an 11-bit ID that can be expanded up to 29 bits and the Remote Transmission Request (RTR) that is dependent on the CAN frame type.
- c. **Control:** This field is also referred to as the test field; this provides the recipient with details to ensure that all planned packets are successfully received. It does have two reserved bits as well as four Data Length Code (DLC).
- d. **Data:** This category of data contains actual details about actions performed by CAN nodes. It can be between 0 and 8 bytes.
- e. **CRC:** The cyclic redundancy code and the protection field is called, a 16-bit fault identification system that guarantees packet validity. All the other nodes getting the message use this tag to check the document.
- f. **Acknowledge (ACK):** Often recognized as the field of confirmation. This field guarantees that the CAN packet is handled properly by the recipient nodes. Whenever an error is found during the transmitting phase, the receiver must automatically alert the sender to send back the data packets.
- g. **End of the Frame (EOF):** This domain displays an end of the CAN frame through the flag of a recessive bit.

2.5 Attack Types

The insertion of many ECUs has resulted in complex in-vehicle networking, consequently creating communication with the outside world through interfaces such as Wi-Fi, Bluetooth, radio, etc. Such assault surfaces open the way for attackers to access a car's internal network. There are several types of techniques that may be used by an attacker to utilize these attack surfaces and take control over the car. The following common cyberattacks that are mostly used: -

Chapter two: Theoretical Background

2.5.1 Denial of Service Attack (DoS)

DoS attack, it is one of the most common technique. It aims to unavailability of an ECU or network resource to its intended users. This kind of attack is meant to stop, disturb, or reduce the functionality of the system. An attacker, for instance, could send multiple valid requests beyond the target systems' ability to process them, exhaust the systems' energy, and paralyze the function. DoS well-known types include smarter cheat attack, jamming attack, and flooding attack [67], [68]. In CAN, all nodes share a single bus. The attack may interrupt the arbitration of CAN protocol to stop messages transmission and making other ECUs stop transmission as well through sending messages of high priority, targeted at a specific ECU. That results in the system (vehicle malfunction)[58].

Injection of messages with a higher priority leads to bus overloading, and this leads to delay or refuse to receive valid messages. It should be noted that in this attack DoS can be initiated by an attacker with primitive or limited knowledge, so the probability and impact of such attack are expected to be high [69].

2.5.2 Message Injection and Replay Attack

As shown in Figure 2.5, this kind of attack happens when the attacker uses the network to sends a valid message then gains access and control of the ECU(D). The intruder then injects fabricated messages into the CAN bus. In a replay attack, the intruder stores a valid message at a certain time and uses it at a later time. [70]. For instance, an attacker may store the speedometer and send it back to the network.

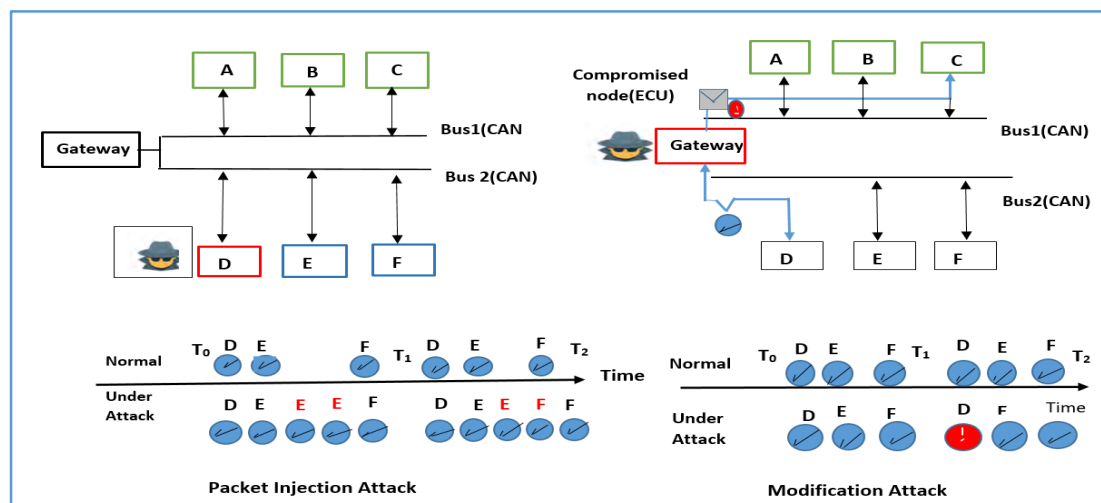


Figure 2.5 Example of an attack in the internal car network [57].

Chapter two: Theoretical Background

2.5.3 Message Manipulation

This attack affects the integrity of the data by deleting messages or altering / modifying data. For example, a message's content may be altered or modified by an attacker. The attacker, as shown in figure 2.5, succeeds in breaching the Gateway, which connects the two CAN bus and changes the source of direction emanating from ECU (D) and intended to ECU (C).

2.5.4 Masquerade Attack

In this kind of attack, which is usually called an impersonation attack, the intruder must compromise the two ECU (A and B) to mount the masquerade attack. The messages sent via CAN bus are monitored by the attacker to know the frequency and the messages which are sent by A. Then, once the IDs and frequency of the message sent by A are identified, the intruder interrupts A's communications and exploits B to produce and insert messages on behalf of A [71].

2.5.5 Malware Attack

With this kind of attack, there is malware in different forms, such as spyware and viruses that can be inserted into the network system by exploiting the vulnerabilities of the user interface. An attacker, for example, can use vehicle media player firmware input vulnerabilities to apply malware to music files. After that, the malware injected can run to send malicious messages to the vehicle's CAN when these music files are played [72].

2.6 Internal Communication Access

The CAN protocol's most popular security issue is the shortage of security features, for instance, authentication and encryption. There is no authentication to the messages and nodes. Because CAN is a broadcast-based bus network, all messages can be received and it is also possible to connect any node to the bus. Therefore, the attacker can conveniently steal data from the CAN bus as well. The attack surfaces are classified as physical access and wireless access. An intruder can access the CAN bus physically via the OBD-II port or remotely via Bluetooth, Wi-Fi, or telematics services, such as 3G. This internal data disclosure causes the target vehicles' CAN data to be accessed by an attacker to control the target vehicle. Thus, these attack surfaces exist in car

Chapter two: Theoretical Background

system eventually have brought to security researchers attention that modern car systems have be designed with no security in mind. Figure 2.6 shows the two separate scenarios for the attack: physical access and remote access.

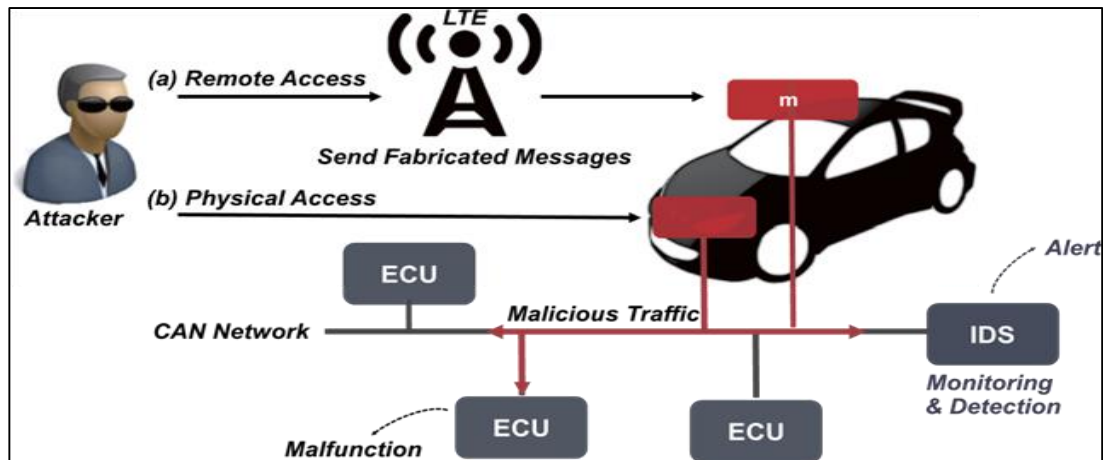


Figure 2.6 Typical scenarios for the attacker access [65].

2.6.1 Local Access

This kind of attack refers to the packets sent directly on a built-in bus (in CAN bus). This attack can be performed through plugged in a device to the targeted bus or by On-Board Diagnostics(OBD) port. The OBD port is the one in a vehicle to diagnose or identify any problem existing in the infrastructure. More specifically, the OBD port can be used to send message or to hack on the bus traffic. It is now widely known that hacks and attacks accessing the internal network are available, as documented examples [17][73].

A malicious ECU can then repeat previous messages and therefore transmit controlling instructions to other ECUs [74].

Alternatively, the attacked ECUs may be on different bus segments from the entry point of the attacker. As such, an intruder may control only one node in the network and available protocols and architectures allow the attacker to get control of any other ECU in the vehicle. here may be faster and easier non-electronic approaches (for instance, cutting brake wires instead of breaching the corresponding ECUs) depending on the goals of the intruder.

2.6.2 Remote Access

In modern cars, wireless communication systems allow an attacker to gain access to a vehicle remotely without the need for a physical attack [75].

Figure 2.7 shows the potential attacks of self-driving cars, categorized and sorted according to their types:

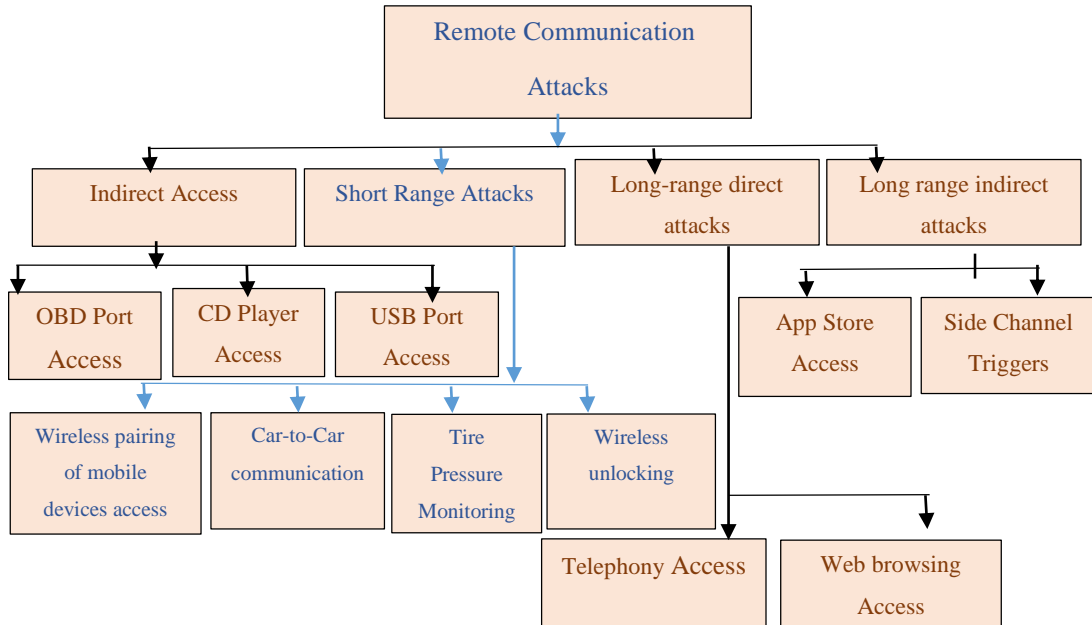


Figure2.7 Remote Access .

1. Indirect Access: modern cars provide many physical interfaces that access the car's internal networks either directly or indirectly. We treat the overall physical surface of the attack under the restriction, that the opponent may not have direct access to these physical interfaces but must work through some intermediary instead. therefore, this access refers to the third-party device that is connected to the cars and poses threats.

- **OBD port:** a diagnostic port can be applied to hack or attack the automotive network. Therefore, the actual assault on the diagnostic system that is attached to the port is passed on. Then, a pass-thru device controlled remotely by a Wi-Fi from a laptop is connected to the OBD port and the hack occurs [76]. A different computer connected to the same Wi-Fi network can then interrogate or program the car's ECUs via this device.
- **CD Player:** two weaknesses are identified on the CD player. The first one is that inserting a CD with a different name file trick the player into thinking that it was a software upgrade, installing new, malicious code. The second one

Chapter two: Theoretical Background

is that decoding the documents enabled the player to create a digital audio file that the player while reading emit the texts on the bus.

- **USB Port:** this attack happens when a media player of a car accesses a compromised file on a USB key. Another attack that can happen to the ECU is that through the connection of the compromised device, for example, a mobile or mp3 player that is attached.

2- Short Range Attacks: these kinds of attacks refer to the use of wireless communications devices that are within a short-range. This kind of attack can occur directly or indirectly if an intruder tries to target and reach a communication module of a vehicle. For instance, a smartphone of a driver links to the target car.

- **The wireless pairing of mobile devices:** call discussions can eavesdrop when phones are connected to modern cars through Bluetooth. This can lead to challenges and the stealing of information stored in the communication device or unit. This can happen between travellers or between phone calls. Or, sometimes even could occur on the ECU where it is compromised.
- **Car-to-Car Communication:** communication between a car and the infrastructure of another car is usually essential. As such, an attacker can use this to eavesdrop or send false data.
- **Tire Pressure Monitoring System (TPMS):** a system has sensors for pressure inside the tires, sends information about the pressure of the tire to an ECU through radio frequency emitter. An intruder can use this to send signals to the control unit to start the alarm light.
- **Wireless Unlocking:** modern cars nowadays use unlocking technology to open doors remotely through some encryption applied to these signals transmitted via the air. However, this can be exposed to signal a violation of the encryption.

3- Long-Range Direct Attacks: in this category, assaults can be implemented through techniques of wireless communication of long-range.

- **Telephony:** the telematics unit could have many vulnerabilities. The attackers can use the 3 G network to carry our assaults.

Chapter two: Theoretical Background

- **Web browsing:** vehicles nowadays have web browsers and attackers could exploit them to create threats through injecting malicious software.

4. Long-Range Indirect Attacks: this type of attack is performed from a remote distance and is considered as very impactful and also detrimental. It is indirect and done through an intermediate device.

- **App Store:** vehicles that download apps for the multimedia unit may be liable to assaults such as Trojan Horse.
- **Side Channel Triggers:** the Radio Data System (RDS) signals can broadcast a risk to electronic units [76][75].

2.7 Security Technique

Security service is important to keep the internal network functions properly because any breach could lead to a major loss. Security systems are necessary to human lives in most cases. Henceforth, any security breach in a network could lead to a serious risk for humans. Therefore, any data transmitted through the network has to be protected against any interception or change in most cases [77].

Specifically, for car security, malicious behavior may have serious effects and bad results on car safety. Especially with the evolution that occurred on modern cars' networks. Their internal network design becomes exposed to many security breaches and challenges. However, due to the vulnerabilities on the bus, it focused on the vulnerabilities of cars' network systems with the analyses of buses' network protocols, and particularly on CAN [62] [73].

However, although if some ECUs are secured, such as the immobilizer are protected by protection mechanisms (authentication), CAN do not guarantee by itself the following protection attribute [58]:

- **Confidentiality:** By design, each message sent through CAN to each node physically and logically. As a result, a malicious node can easily eavesdrop into the bus and read the content of each frame.
- **Authenticity:** The CAN frame does not provide a sender authentication field. Therefore, any node may send messages that only other nodes can send. therefore, any node can be connected to the bus.

Chapter two: Theoretical Background

- **Availability:** An intruder can easily trigger a denial of service on the bus through CAN arbitration rules. For instance, an ECU can overwhelm the bus with high-priority frames causing any other ECU to interrupt its transmissions.
- **Integrity:** CAN uses a CRC to test whether a message has been changed due to a transmission error and that is insufficient to deter an intruder from altering the correct message or creating a false message because it would be easy to forge the correct CRC for a false message.
- **Non-Repudiation:** Currently, there is no way for a legitimate ECU to prove that a stated message has not been received or sent.

It is more evident that the self-driving vehicles' success depends heavily on their networks and networks' efficiency, which in response depends on their security's scalability. The security systems are generally split into two sections. Firstly, the intrusion prevention methods, such as authentication and encryption (utilizing passwords or biometrics) that are the first line of defences. Secondly, intrusion detection strategies ("any collection of acts that seek to compromise the confidentiality, integrity, or availability of a property"[78])

Security requirements vary from system to system, so this depends on the type of device and the system design's main goal. Thus, in this thesis, the new Intrusion Detection System (IDS) is proposed using the Artificial Neural Network (ANN) to monitor the car's status by collecting information from internal buses and ensuring the car's internal network's safety.

2.8 Intrusion Detection System(IDS)

An IDS is an application that is used to monitor activities in the network and protect it from the attacks. It is a series of actions aimed at compromising the confidentiality, availability, or integrity of the resource. Intrusion detection raises many interests largely due to its simplicity and the ability in detecting the attacks efficiently. Generally, IDS tries to detect inappropriate or inconsistent behavior occurred on network or on a host by monitoring and discovering certain types of data. This behavior may be triggered by external intruders or internal attacks[79].

2.9 Type of Intrusion Detection Systems

There are many classifications of IDS. These systems vary largely as to architecture used, and which Detection Methods performed. IDS can be categorized into the following general categories as shown in Figure 2.8 [80].

1. Based on Architecture (Where an analysis of data is made)

- Host-based Intrusion Detection Systems (HIDS).
- Network-based Intrusion Detection Systems (NIDS).
- Hybrid Intrusion Detection System.

2. Based on Detection Methods (How the analysis of data is achieved)

- Anomaly detection system.
- Signature detection system.
- Hybrid detection (Anomaly and Signature).

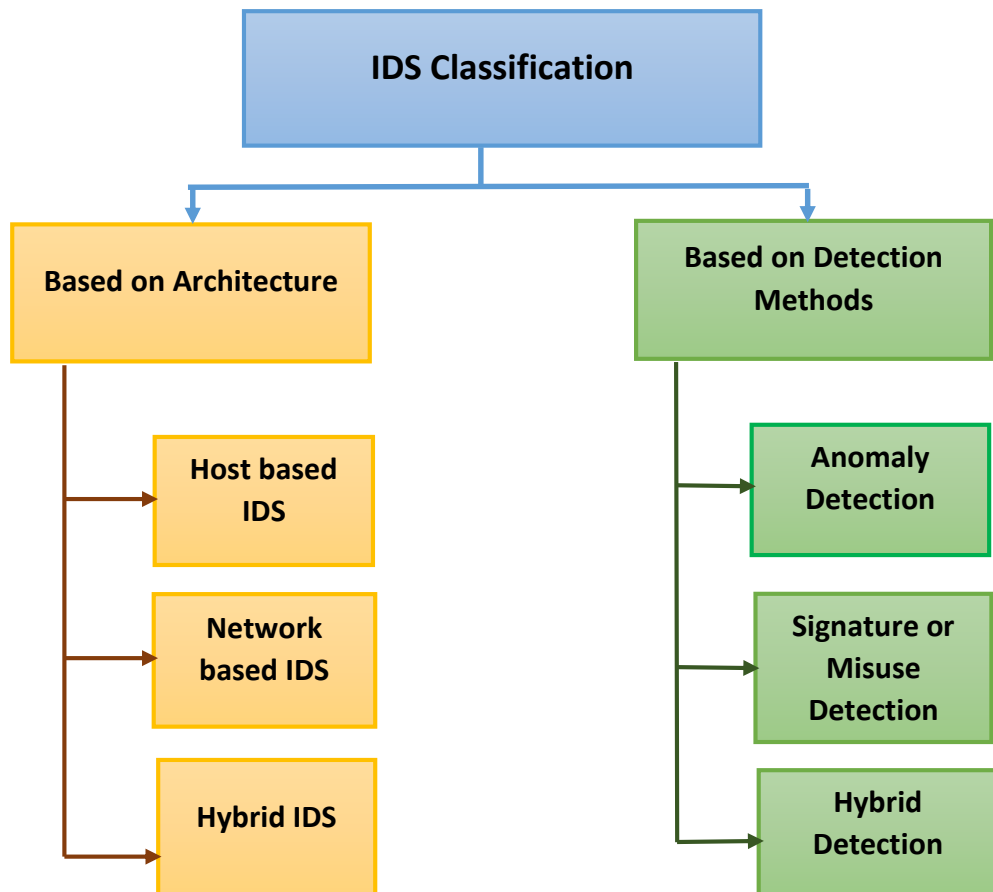


Figure 2.8 Classification of IDS based on used Methodology.

Chapter two: Theoretical Background

2.9.1 Host-Based Intrusion Detection System (HIDS)

It's typically a software application installed on a single device or server, known as the host, and controls the operation only on that system. It can be categorized into two types: anomaly-based detection techniques and signature-based (i.e. misuse detection). HIDS often tracks the status of key device files and detects when the attacker establishes, modifies, or removes the files being monitored. The HIDS will trigger a warning when one of the following changes occurs: the file attributes will be changed; new files will be generated, or existing files will be removed. Figure 2.9: show a network using HIDS on servers and host computers.

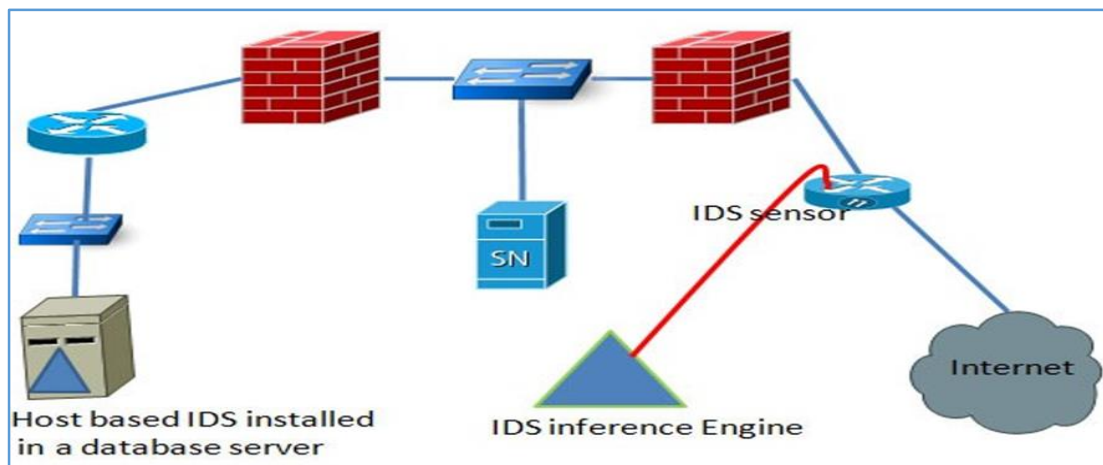


Figure 2.9 Host-based IDS[81].

2.9.2 Network-Based Intrusion Detection (NIDS)

It is a stand-alone platform that tracks network backbones and looks for attack scenarios by evaluating and tracking network traffic information. NIDS is installed at a specific location on the network (the router is an example) and it possible to access network traffic by linking to a hub. NIDS monitors network traffic using strategies such as deep packet inspection to collect network data and attempts to detect suspicious activity such as a denial of service attacks[23]. It re-assembles and analyzes all network packets that link the network interface card. Not only should deal with packets moving to a specific host, like all machines in the network segment benefit from NIDS protection. The major difference between the NIDS and the HIDS is that the NIDS can access encrypted information when traveling over the network [82].

Chapter two: Theoretical Background

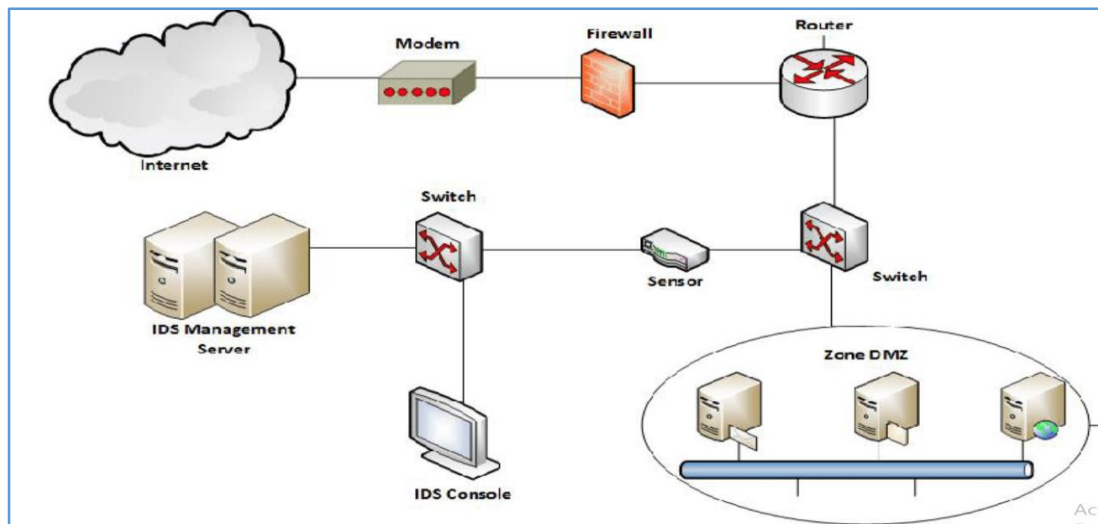


Figure 2.10 Network-based IDS [83].

2.9.3 Hybrid-Based Intrusion Detection

In the modern IT environment, hybrid IDS is often called a Distributed Intrusion Detection System (DIDS). It combines many IDS techniques such as network-based IDS and host-based IDS across a large, interacting network, controlled by a central server [84]. The host IDS elements on the network capture and transform controlled system information into a standard format and relay the information to the central server. The main system collects and processes the information obtained from several IDS.

Host agent data is combined with network information to provide a universal view of the network. The key explanation for the implementation of such a hybrid IDS is the need to operate together with encrypted networks and their information on a single host (Only the source and destination of the network will see decrypted traffic). The hybrid intrusion detection system is versatile and increases the degree of protection. It combines the locations of the IDS sensor with warnings of attacks targeting specific segments or the entire network [80][82].

2.10 Classification of IDS according to detection method

Intrusion detection has become an important field of network security. It is a software program that tracks a network or system for malicious activity or policy violations. The process for detecting intrusions in the automotive domain depends on how the detection mechanism is used within the system[85]. It can categorize into four types, anomaly, signature, specification and hybrid-based IDS. Compared to the signature- and specification-based approach, the anomaly IDS is the most popular and promising method used in the cars.

2.10.1 Anomaly-based IDS

This method monitors the behaviors of the real-time system and compares it to the normal behavior documented in the profile. Once the deviation from standard profile activity crosses a certain level, the alarm is lifted [86]. Such an approach can also easily recognize new attacks following a training phase. However, this approach assumes that anything that deviates from normal behavior is a snooping message[87].

Consequence, this method represents a challenge where it already produces false-positive warnings for normal packets.

The inclination of CAN packet IDS towards the anomaly-based approach over other methods is due to the constraints and limitations that it possesses. The proprietary characteristic of CAN protocol makes it difficult to adopt a signature or specification-based approach, as these approaches required semantic knowledge of CAN packet, and the protocol may change frequently.

The main advantage of anomaly-based systems to signature-based systems is the first approach capable of detecting novel forms of anonymous attacks[88].

2.10.2 The signature-based IDS

This approach uses a set of specified signatures, rules, or malicious events stored in the database module to detect an attack. Bad types such as malware and viruses can be detected. This form can detect only known attacks. This strategy compares the behavior of the network or device to the attack patterns stored in IDS and generates the warning if it matches the malicious patterns stored.

The pattern-based approach is promising because it is not complex to construct and can increase the efficiency of detecting known attacks. As this method focuses on

the database, the identification of new or unknown attacks is ineffective. Therefore, the IDS needs to update new attack signatures regularly [89].

2.10.3 Specification-based IDS

The specification is usually a set of rules and thresholds that define the well-known behavior of network elements for example protocols, nodes, and routing tables. The specification-based approach is used to detect attacks whenever the predictable behavior of the network varies from the specified specifications. The purpose of a specification approach is, therefore, similar to the anomaly approach, everything that differs from the well-known defined behavioral profile is described as anomalies. However, the only major difference between the two methods is that each rule and definition require to be manually defined by a human expert.

Nevertheless, a training phase for this method is not needed as it will work immediately once setting up specification is finished. Furthermore, setting the parameters manually will be mistake-prone and take too much time, as it may be difficult for the system to adapt to different domains [84].

2.10.4 Hybrid-based approach

Hybrid detection now uses to combine the two types of intrusion detection systems. Hybrid systems can use signature-based IDS to detect only known attacks and an anomaly -based approach to detect novel attack. Hybrid IDS is developed that try to combine the advantages of anomaly and rules-based IDSs while eliminating some of their disadvantages[90].

However, in comparison to the CAN bus environment, hybrid anomaly detection incorporates further than one approach takes into consideration the CAN timing interval, CAN ID field, the CAN specification, the CAN data payload, or its frequency when detecting attacks[91][92].

2.11 Features Clustering

Clustering is a way to reasonably classify raw data and to search for hidden patterns that may exist in datasets. [7]. clustering is an important technique of exploration and data mining which has been widely used. It is a process of clustering data objects into grouping so that we can achieve a high degree of similarity between the elements from the same cluster and dissimilarity in other clusters.

Cluster analysis aims to find natural clustering within a set of information (e.g., patterns, objects, or points). The three main purposes for data clustering usage are: to find out the data structure, to detect anomalies, and to discover salient features, for identification the degree of similarity among forms, cluster analysis can be used as a method for compression by organizing and summarizing the data.

The K-means clustering algorithm is an effective method. It's an unsupervised, numerical, non-deterministic, and iterative technique. It is easy and very fast, so in many practical applications, it is a very efficient way of producing good clustering results. This method used to classify the features in k different clusters by the iterative, where the error between the mean of the cluster and the points belong to it is minimized as possible. So, the results of generated clusters are compact and independent. The inputs are a record of data and clusters K. The data set consists of a collection of features for each data point. The algorithm begins with initial estimates for the K centroids, which can either be generated randomly or selected randomly from the data set.

K-Means clustering algorithm steps:

- a) Initialize, arbitrarily K, centroids.
- b) For each set of point $X_1, X_2 \dots X_i$, compute its distance with each centroid $C_1, C_2 \dots C_k$ and assign it to cluster K_j whose centroid show minimum distance to the point such as using measuring distance (Euclidean distance).
- c) For each cluster, K_j , re-compute its new centroid from all data point's X_i belong to this cluster.
- d) Repeat steps (b) and (c) until the clusters memberships stabilize, when none of the cluster assignment change (no data points change clusters).

Figure 2.11 shows the basic steps as a flow chart of K-means clustering algorithm.

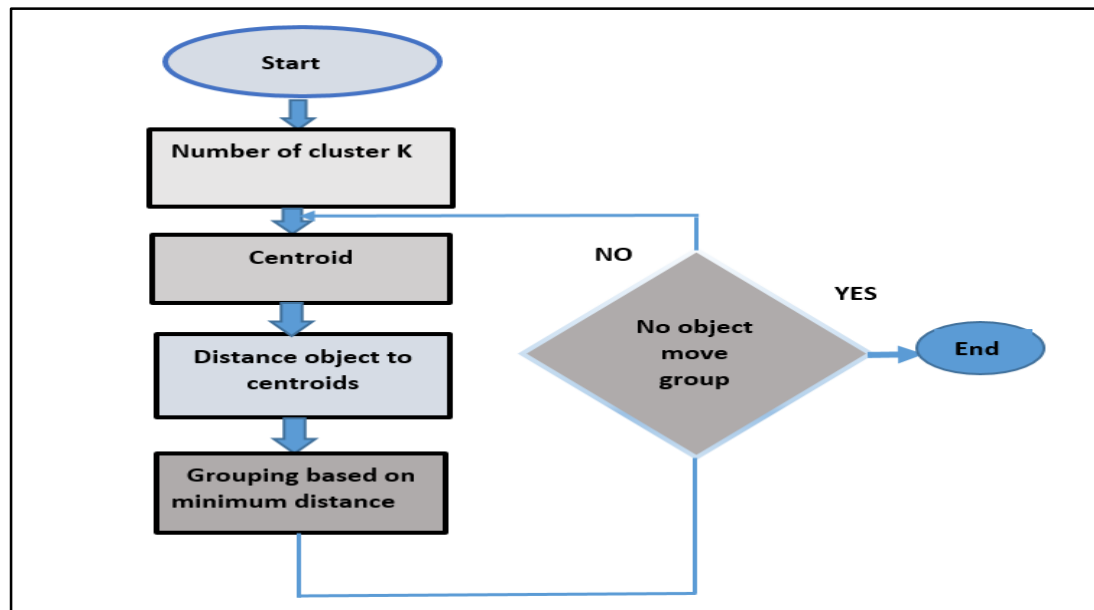


Figure 2.11 The flow chart of the K-means algorithm (applied with K=2).

2.12 Features Selection

Datasets contain great number of redundant and irrelevant attributes[93]. Irrelevant attributes can lower the efficiency and high misclassification rates by the classifier. Therefore, any intrusion detection system is to reduce the dimensionality and select the feature subset of the utilized dataset, especially if the size of the dataset is large. Dimensionality reduction means the transformation of the high dimensional data into reduced dimensionality with a meaningful representation of the data. Ideally, the reduced representation should observe the properties of the data with a minimum number of parameters. However, feature selection gives an advantage over dimensionality reduction methods because it provides the ability to distinguish and select the best existing features in the feature pool which in turn yields to dimensionality reduction[94].

The aim of feature selection is to find a subset of the attributes from the original dataset to reduce the time complexity by selecting only useful features to build a model for classification. Feature selection is an effective approach which further helps in building an efficient classification system and improving the accuracy of model[95].

2.12.1 Feature Selection methods

The feature selection can be categorized into three standard methods:

1. **Filter:** Filter methods use variable ranking techniques as the principle criteria for variable selection and evaluate the relevance of features by looking at the essential properties of the data [96][97]. A suitable ranking criterion is used to record the variables and a threshold is used to remove variables below the threshold[98]. This method evaluate the relevant of feature by measures information, distance, consistency, similarity, and statistical measures[96]. The filter methods are found to be fast, simple, and independent of the classifier. The disadvantages of this methods are that they shortage interaction with the classifier, as a results lower classification accuracy[99].
2. **Wrapper:** The wrapper methods function almost similar to the filter methods except that they make use of a predefined classification algorithm instead of an independent measure for the subset evaluation. These algorithms are called sequential due to the iterative nature of the algorithms. The most effective feature selection technique is Sequential Forward Feature Selection (SFFS) method[100]. starts from the empty set and adds one feature for the first step which gives the highest value for the objective function. From the second step onwards the remaining features are added individually to the current subset and the new subset is evaluated. The individual feature is permanently included in the subset if it gives the maximum classification accuracy. The process is repeated until the required number of features are added. A Sequential Backward Selection (SBS) algorithm can also be constructed which is similar to SFS but the algorithm starts with a full set of features and removes one feature at a time whose removal gives the lowest decrease in predictor performance[100].
3. **Embedded:** Embedded methods used to reduce the computation time taken up for reclassifying than the wrapper approach. The main approach is to incorporate the feature selection as part of the training process make better use of the available data by not needing to split the training data into a training and validation set; they reach a solution faster by avoiding retraining a predictor from scratch for every variable subset investigated[101][102].

The Sequential Forward Feature Selection (SFFS) method has been used in our proposed system.

2.13 Limitations

One major problem with an IDS is that they alert you to false positives on a regular. False positives in several cases are greater in frequent than real threats. To decrease the number of false positives, an IDS can be tuned, but if we are not careful to monitor the false positives, real attacks can intrude.

There were certain limitations that need to be taken into account when designing proposed solutions within the CAN bus network system explained as follows:

1. Restricted resources: all ECUs have several limitations related to memory storage, and data transmission within a car (bandwidth).

2. Real-time limitation: The CAN packets are transmitted in real-time from the ECUs to other node, so delaying and queuing packet buffers becomes dangerous. thus, when proposing any security solutions, should be taken into account of the real-time limitation.

3. Traffic conduct: The CAN bus protocol behaviors for the automotive communication system differ from those of the traditional Internet Protocol (IP) networks. For example, CAN packets are broadcasting in nature. In addition, in order to perform software update and wireless diagnostics on cars, need to establish a temporary connection from in-vehicle to infrastructure(V2I) that requiring various solution for traffic protocols and communication models. Therefore, the adoption of an IP network solution is not feasible.

4. connections unstable: As vehicles move, irrespective of speed, they might also move to an area that does not have network connection. thus, the IDS solutions for the self-driving cars domain should be knowing that the linking to a third party will not always be available.

5. Cost, size, and weight: The IDS solutions might also differ depending in size and weight, and may also require some small or possibly significant changes that will ultimately affect the cost. For instance, removing a CAN bus network that adopts line topology, replacing it with an Ethernet bus network that uses a different topology, requires great number of wiring and may be ineffective.

Chapter THREE

Proposed System Design and

Implementation

Chapter Three

Proposed System Design and Implementation

3.1 Introduction

IDS greatly raise many concerns due to its simplicity and its efficient detection of attacks. Generally, it monitors behavior in the network, in case any unexpected events occur in the system it raises the alarm. Such unusual events, defined as intrusions, battle their way into the network to gain unauthorized access. Intrusion operations may be internal or external intruders, may come from outside the target network, to obtain system components illegally. The IDS may be either active or passive, depending on how it has been set up. Active IDS takes preventative behavior against an attack while Passive IDS only detects an attack.

In this chapter, an intelligent anomaly intrusion detection system (IDS) is designed for the internal communication system of the self-driving car. This system based on (ANN) to monitor the state of the car by information collected from internal buses and to achieve security, the safety of the internal network against Denial of service attack. In addition, the ANN structure is trained CAN packet information to select the basic attribute of normal and attacking packets and to classify the associated attribute to the proper intrusion type. Moreover, we discuss data preprocessing, clustering, features evaluation, and selection using simulation and real dataset and more details about the proposed system.

3.2 Intrusion Detection system architecture

The security systems are proposed in this thesis is composed of nine stages namely, data collection, pre-processing, feature generation, Feature Clustering, normalization, evaluation, selection, training, and classification (detection stage). However, the basic architecture of the proposed system is shown in Figure 3.1.

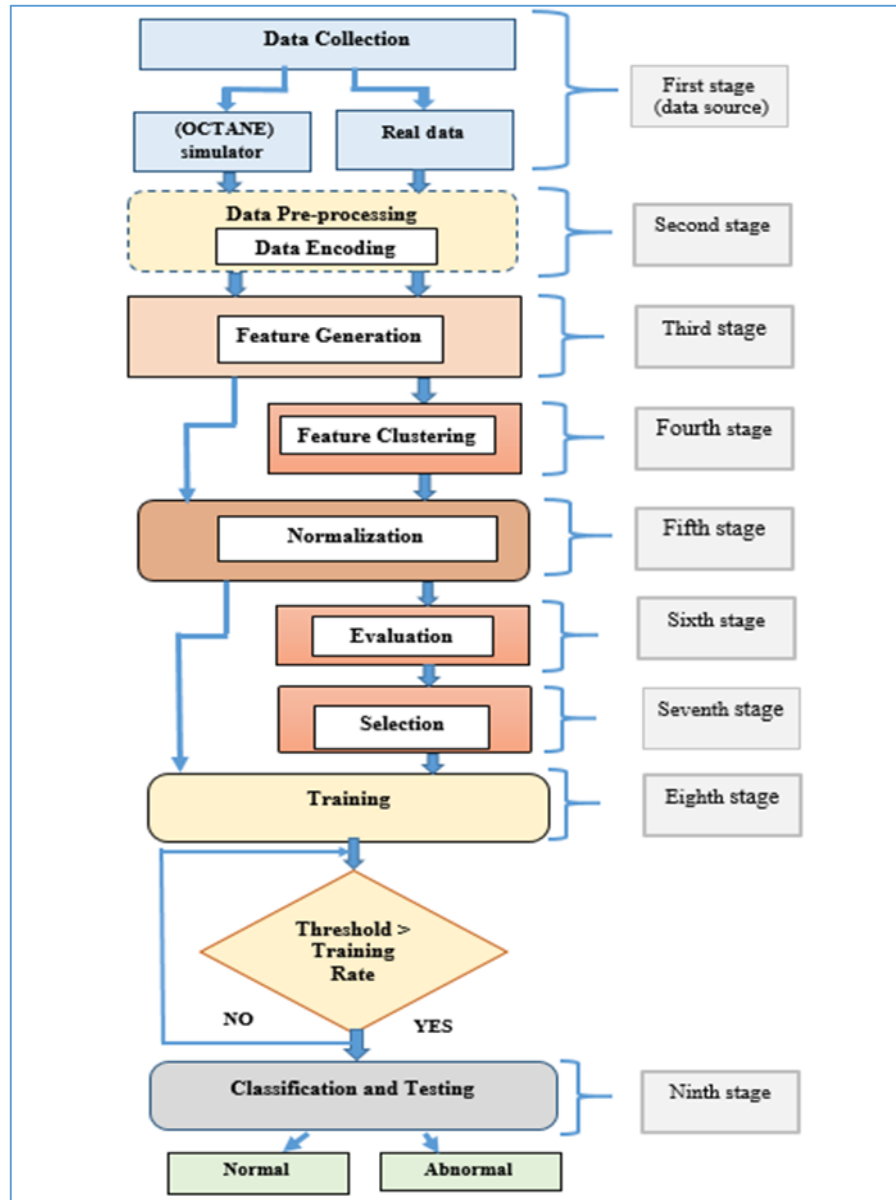


Figure 3.1: Block diagram of the Proposed System.

Chapter three: Proposed System Design and Implementation

3.2.1 Data Collection

Intrusion detection depends mainly on observing data exchanged among connected entities/nodes (e.g., ECUs). A dataset comprises a set of records, which represent events, objects, or processes.

The first step in the detection process for anomalies is to collect messages from the bus. Intelligent detection systems can only be designed if an appropriate data set is available. Collecting data with a large amount of quality information can help train and test the IDS.

Two sets of data are required for the proposed model:

1. Simulated data: A simulated test environment Table 3.1 was generated to obtain normal and abnormal data. The simulator is based on the well-known Open Car Test-Bed and Network Experiments (OCTANE)[103]. OCTANE is a software tool used for cyber-physical security to create the packets. CAN messages on the bus are transmitted at periodic intervals of time. The initial parameters are one of the important elements of the simulation that determined efficiency and behavior in OCTANE.

Table 3.1 Sample of simulation data

A	B	C	D	E	F	G	H
ID	ECU_ID	Priority	DLC	Flags	Data	Time	Class
242	240	2	2	2	01 3E	6856	1
242	240	2	2	2	01 3E	7856	1
242	240	2	2	2	01 3E	8856	1
242	240	2	2	2	01 3E	9856	1
242	240	2	2	2	01 3E	10856	1
242	240	2	2	2	01 3E	11856	1
242	240	2	2	2	01 3E	12866	1
242	240	2	2	2	01 3E	13866	1
242	240	2	2	2	01 3E	14866	1
242	240	2	2	2	01 3E	15866	1
242	240	2	2	2	01 3E	16866	1
242	240	2	2	2	01 3E	17866	1
242	240	2	2	2	01 3E	18866	1
242	240	2	2	2	01 3E	19866	1
242	240	2	2	2	01 3E	20866	1
242	240	2	2	2	01 3E	21866	1
242	240	2	2	2	01 3E	22866	1
242	240	2	2	2	01 3E	23866	1
242	240	2	2	2	01 3E	24866	1
242	240	2	2	2	01 3E	25867	1
242	240	2	2	2	01 3E	26868	1
242	240	2	2	2	01 3E	27868	1
242	240	2	2	2	01 3E	28868	1

Chapter three: Proposed System Design and Implementation

2. Real data: The dataset used in this thesis was obtained from a real vehicle and is available online for public use [104]. It is the open dataset that includes normal and attack packets with labels. However, these features reflect normal and abnormal behavior on the network. CAN Dataset was used for intrusion detection by Hacking and Countermeasure Research Lab for measuring the performance of the security system. Moreover, it is built from real data on a network, include DoS attack; were constructed by CAN traffic. The data is loading into a file and then analyzed.

3.2.2 Feature sets

The IDS is based on features that describe internal events in the car. IDS monitor and extract the behavior from the data set because it consists mainly of several different features that can be used for analysis. Characteristics identify behaviors that are normal or abnormal and used to measure the effectiveness of the suggested IDS. The type and number of features of the IDS are very significant that affect the number of alarms and detection rates.

The features set for simulation and real data are shown in Table 3.2

Table 3.2 Feature type of CAN

Features	Type
ID	Discrete
ECU-ID	Discrete
Priority	Discrete
DLC	Discrete
Flags	Discrete
Data	Continuous
Time	Continuous
Class	Discrete

3.2.3 Data Preprocessing

The dataset is utilized to assess the performance of the proposed IDS. Network traffic includes numerous types of data (continuous, symbolic, and discrete) that vary considerably in ranges. To handle such type of data, the pre-processing stage of data is important.

- **Data Encoding**

Some features were represented by symbols such as “ID” “DLC”. The proposed intelligent detection system deals only with numerical values. In this case, the security system needs to convert symbolic attributes to numeric values before making any changes to the data set. The conversion is done by assigning an encoding number for each symbol. The encoding starts at 1 and increases one for each symbol within each feature.

3.2.4 Feature generation

It is the process used to select one or more attributes from the dataset and generate new features from them, having more features should result in a more discriminating power. The objective of creating new input from existing features is to create ones that do a better job of representing a machine learning problem to the model and to improve the accuracy of the model.

Nevertheless, enhancing the space of features in this way is often an important step in the processing and classification of data by generating new features from an existing feature. Because the raw feature space is generally filled with an overwhelming number of irrelevant and unorganized data, don't has the features needed to perform machine learning tasks (i.e. some data have predictive value and some data do not have predictive value). therefore, we enhanced the feature space and the important data was better to identify, which has predictive value in analysis.

3.2.5 Feature Clustering for real data

The cluster analysis step is needed to explore the features that can be effective if they are clustered in more than one cluster.

In the proposed model, labeled each sample of the original data set, by dividing the samples into two clusters (using the K-Means algorithm) for identifying similar samples and considering them as a single class, the cluster with larger samples will have a normal class and the cluster with have fewer samples will have abnormal class because normal instances should form large clusters compared malicious intrusions.

3.2.6 Normalization

The data attributes are scaled to be within the range [0, 1]. The main advantage is to avoid attributes that have greater ranges from dominate those in smaller ranges. The scaling is performed using the following equation[105].

$$A = \frac{a - \mathit{min}_{old}}{\mathit{max}_{old} - \mathit{min}_{old}} \mathit{max}_{new} - \mathit{min}_{new} + \mathit{min}_{new} \quad (3.1)$$

where **A** and **a** are new and old data values, respectively. min_{old} and max_{old} are the old range confines while min_{new} and max_{new} are the new range confines. Normalizing data also allows the detection rate to be increased and the performance of ANN to be improved.

3.2.7 Feature Evaluation for real Data

Evaluation of features is achieved by calculating the capabilities of each feature intra-class (in a class) and inter-class (between class discrimination). The intra-class value of the feature indicates how interlinked the feature within-class samples and discrimination indicates how much the feature will overlap between classes. Therefore, the best features are those that have the highest variations within-class and have the lowest variations between classes. Inter-class (Bs) is calculated using the following equation[105]:

$$Bs(j) = \frac{\frac{1}{Nc} \sum_{i=1}^{Nc} \sigma_i(j)}{\frac{2}{Nc(Nc-1)} \sum_{i=1}^{Nc-1} \sum_{j=i+1}^{Nc} |M_i(j) - M_j(j)|} \quad (3.2)$$

where,

Nc is the number of classes.

$\sigma_i(j)$ is the standard deviation of j^{th} feature belong to i^{th} class.

$M_i(j)$ is the mean of j^{th} feature belong to i^{th} class.

i and j class number and feature number, respectively.

Normal Distribution is defined precisely by its mean and standard deviation. The mean value changes the distribution spatially and standard deviation controls the spread. After measuring the discrimination ability of each feature, the best M feature is passed to the next step in feature selection stage.

3.2.8 Feature Selection for real Data

Feature evaluation makes preparation for feature selection; nevertheless, precise feature evaluation greatly relies on the preprocessing quality of feature values. The feature selection stage in the proposed IDS model consists of a feature evaluation step.

The objective of feature selection is to improve computational efficiency while preserving or enhancing classification accuracy by eliminating irrelevant data.

The first step of constructing the ANN as a classifier algorithm to prepare CAN Dataset using Algorithm 3.1:

Algorithm 3.1: Preparing the data set
<p><i>Input: Original Dataset</i></p>
<p><i>Output: The dataset is ready for Training, Testing.</i></p>
<p><i>Step 1: Delete the feature which does not affect the real dataset.</i></p>
<p><i>Step 2: Convert symbolic Features to numeric values:</i></p> <ul style="list-style-type: none">• <i>Convert from hexadecimal to binary representation.</i>• <i>Convert binary to decimal.</i>• <i>Convert the result string to double representation to facilitate the next steps.</i>
<p><i>Step 3: Feature generation</i></p> <ul style="list-style-type: none">• <i>Generate new feature from the original feature based on ID, the number of times the packet appears in the next 10 Ms as $d=1, d=2 \dots d=10$, where d represents the number of times the packet appearance.</i>• <i>The number of the packet appears to have the same ID with data that has zero value in the last 10 packages.</i>• <i>Generate a new feature from the number of times the same ID appeared with a different data length in the last 10 packages.</i>• <i>Generate a new feature from the time difference between the current package and the previous package.</i>

Chapter three: Proposed System Design and Implementation

- Adding new features to the features found in the original dataset.

Step4: The class label for each sample with the original data set by dividing the samples into two clusters (using the K-Means algorithm) for identifying similar samples and considering them as a single class, the cluster with larger samples will have a normal class and the cluster with have fewer samples will have abnormal class.

Step5: Convert feature values to numeric. (original dataset contains a categorical value, it must be converted to numerical within the range [0, 1]).

Step6: Feature evaluation measuring the discrimination ability of each feature.

Step7: Selected the best M feature from discrimination.

The steps to prepare the real data set are shown in Figure 3.2.

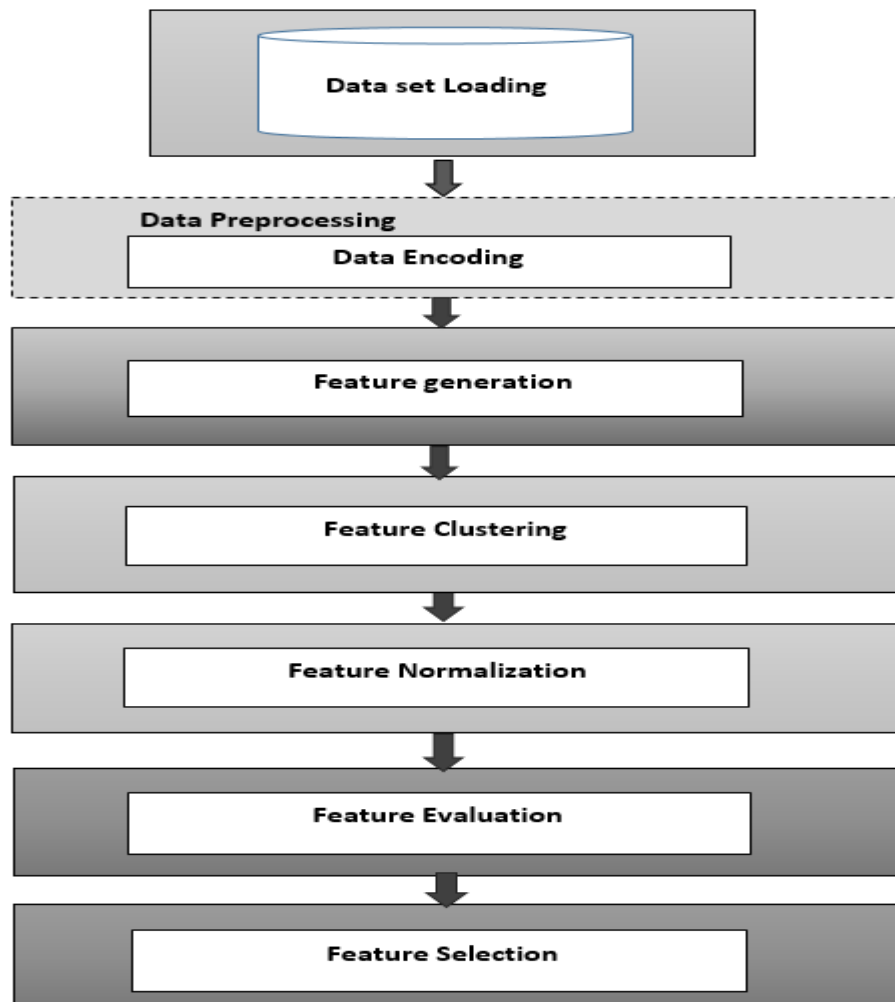


Figure 3.2 Real Dataset Preparing.

3.3 The ANN Architecture

3.3.1 Real dataset

ANNs are a data processing unit that is inspired by neurons in the human brain. It consists of several layers of simple interconnected processing units called nodes or neurons. An artificial neuron is a mathematical function designed as a biological neuron model. A growing neuron is linked to other neurons by a weighted relationship that defines the cross-node impacts. These networks entering input data from the input layer to the output layer to measure the values. ANNs learn to achieve the desired results by changing the weights of the links between the network nodes[106].

The intelligent detection system uses a Feed-forward and back-propagation learning, these are two methods of ANN learning; furthermore, back-propagation performs perfectly on tasks of classification and prediction[107]. The proposed security system used ANN to classify malicious attacks. ANN includes three layers, input, hidden, and output.

The input layer is made up of (15) neurons equal to the number of features. The hidden layer contains (1-15) neurons. The number of neurons used in the hidden layer influences ANN output, so the higher the number of neurons, the greater the efficiency. Nevertheless, an increase in the number of neurons has resulted in a significant increase in the cost of computation. In the output layer, there are two neurons (normal and abnormal).

Chapter three: Proposed System Design and Implementation

Figure 3.3 shows the ANN architecture for real data that has been used in the proposed system.

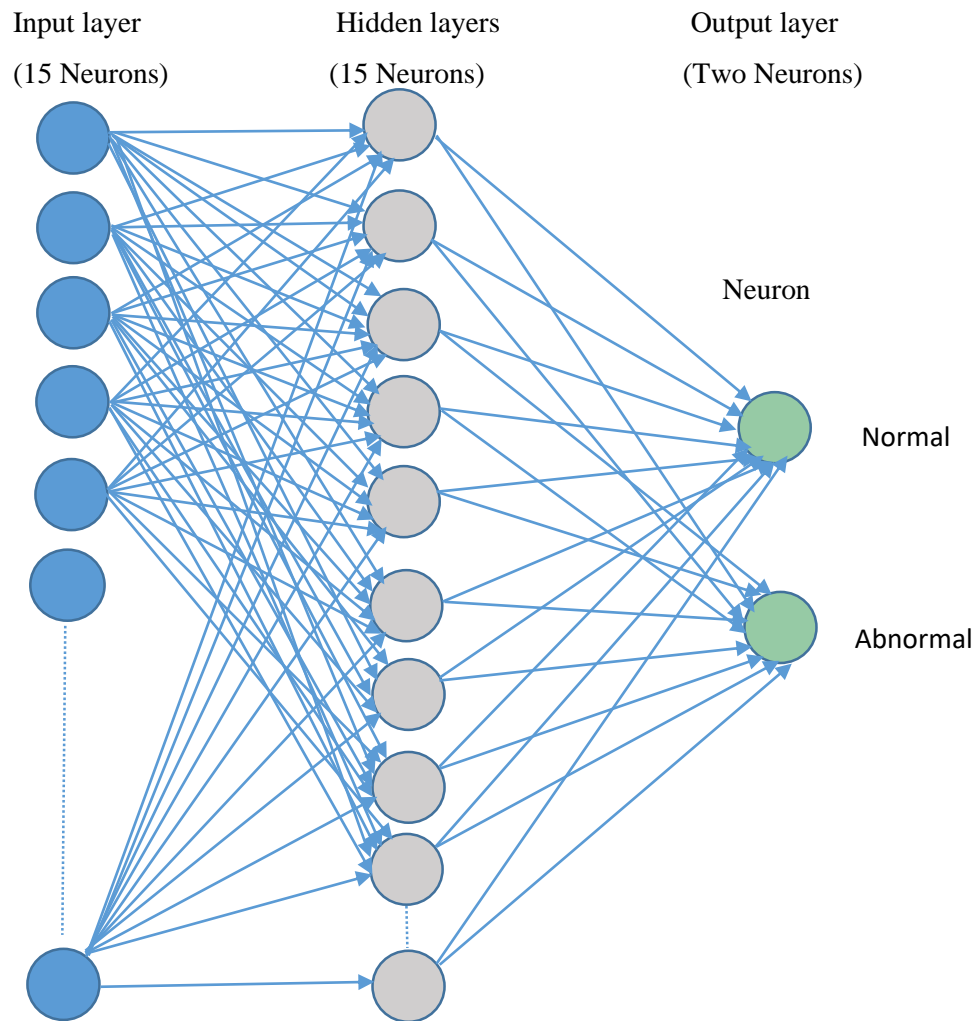


Figure3.3 Structure of the Artificial Neural Network for real data.

3.3.2 Simulation data

An Artificial Neural Network (ANN) is used by the intelligent detection system to classify malicious attacks. Current self-driving car studies confirm that ANN is effective and easy in designing the internal systems for these cars. ANN used includes three layers, input, hidden, and output. The input layer is made up of seven neurons equal to the number of features. Used two hidden layers to reduce the number of false alarms and improve the accuracy of the detection system. The first hidden layer contains three neurons, while the second hidden layer contains seven neurons. In the output layer, there are two neurons (normal and abnormal).

Chapter three: Proposed System Design and Implementation

However, the classifier decides which group the packet belongs to divides the datasets into normal packets and attack packets. Figure 3.4 shows the ANN structure for the simulation data used to train the features.

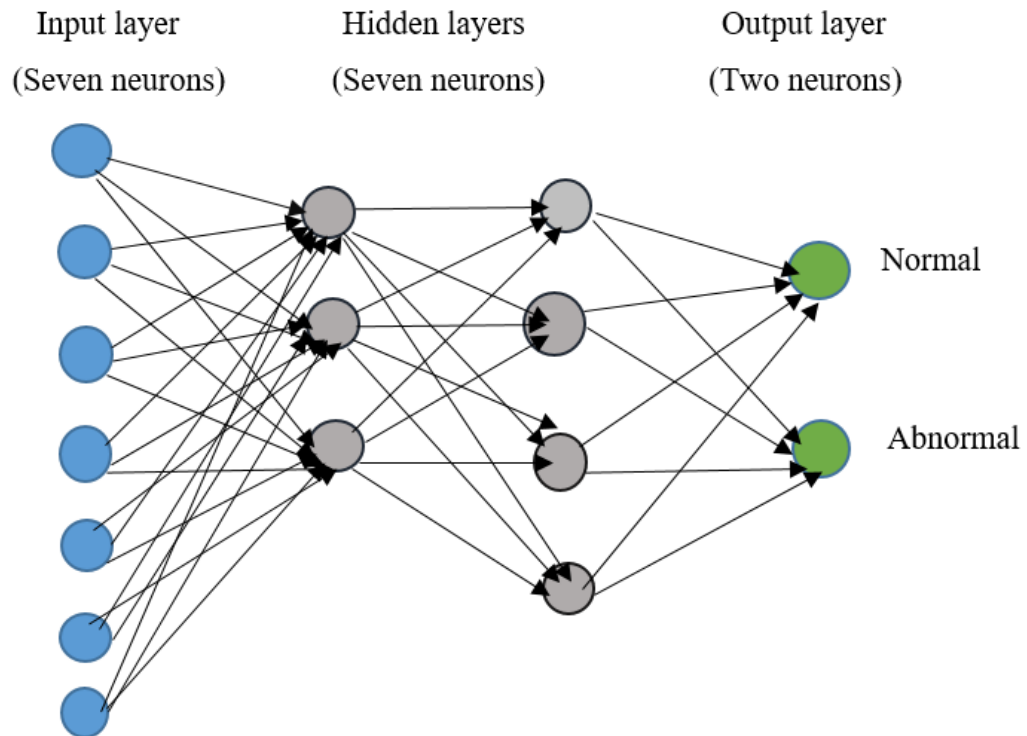


Figure3.4 Structure of the Artificial Neural Network of simulation data.

IDS design includes two primary phases:

- ❖ The training phase: The ANN is trained with the extracted data set (features).
- ❖ The testing phase: The ANN is tested with data attributes that define behavior, normal and abnormal.

Malicious vehicles can perform many types of attacks, such as DoS attacks. The detection system is built to detect the DoS attack which detects through its behavior.

3.4 ANN performance Steps

3.4.1 Simulation Data

Figure 3.1 shows the methodology used. All categorical data that are in the textual form are converted to a numerical form in the preprocessing step. The data set is split into three subsets: the first is a training set 50%, the second 25% is the validation, and the third 25% testing set. The initial parameters have an important role in the performance of ANN and have a direct impact on performance. The training phase parameters used in the ANN are shown in Table 3.3 below:

Table 3.3: Artificial Neural Networks (ANN) Parameters.

Parameters	Value
Train Parameters Epoch	41
Train Parameters learn	10^{-7}
Train Parameters goal	0
Train Parameters min-grad	10^{-8}
Time	0.4s

In this system, the train parameters epoch is the stop condition uses to enhance the result of the training and according to Table 3.3 we note that the ANN received an acceptance training rate of 41 epochs. The value of train parameters learn is 10^{-7} used to update weight.

The principle of trial and error applied to select the best FFNN configuration. The ANN is tested with data features that describe malicious and normal behavior. The labels of the test data are predicted using the model. However, Accuracy, False-negative Rate (FNR), False- Positive Rate (FPR), and True -Positive Rate (TPR) are computed to evaluate the proposed model performance (the key factors to evaluate any IDS).

The model is used to build the following steps.

1. Establishing data from simulator.
2. Data is pre-processed.

Chapter three: Proposed System Design and Implementation

3. The data are split up into training data and test data.
4. Build the model to training data
5. Read the test data
6. validation the model.
7. Compute and compare all models using TPR, FPR, FNR, and Accuracy.

3.4.2 Real Data

The implementation of the ANN algorithm is described in this section. After all dataset analysis and preprocessing operations, we tried to implement individually ANN algorithm with 15 features. We are then implemented with various parameters to choose the best options for the ANN algorithm in the domain of the intrusion detection problem. In this way, we built an ANN algorithm with different parameters to achieve different results and to choose the appropriate one. The learning rate, hidden layers, the number of neurons in the hidden layers are the parameters used to construct the ANN architecture. This parameter will be described in (1,2,3,4). Figure 3.5 shows the ANN that uses algorithm 3.1 output data to measure the model by training and testing data to gain accuracy.

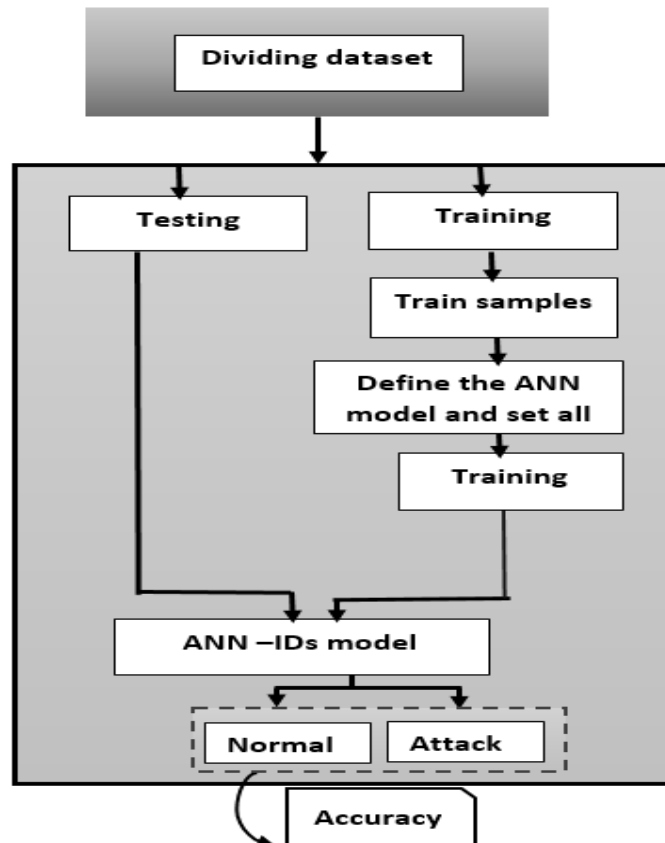


Figure 3.5 ANN Implementation for the proposed system.

Chapter three: Proposed System Design and Implementation

As shown in Figure 3.5 dataset was divided into two parts by a certain percentage, the dataset was split into an 80% training set and a 20% testing set. After the input layer data format to be managed by the ANN algorithm (samples, time-steps, features) the training data is processed to train and generate a trained model. The tested data will then be analyzed to measure the accuracy of the model.

1. Activation functions

An artificial neuron (AN) is the most basic element of an ANN that is inspired by biological neurons within the human brain. (AN) measures and forwards the sum of the information received on its input side. Due to the non-linearity of real-life problems and in an attempt to improve learning and approximation, each AN uses an activation function before generating output. An activation function performs a mathematical operation on the signal output. More sophisticated activation functions can also be utilized depending upon the type of problem to be solved by the network. The most common activation functions are presented.

- **Sigmoidal Activation function**

This nonlinear function is the most common type of activation used to construct neural networks. It is mathematically well behaved, differentiable, reduce the computational capacity for training. A sigmoidal transfer function can be written in the following formula [108]:

$$\mathit{sigmoid} = \frac{1}{1 + e^{-x}} \quad (3.3)$$

The activation functions received input node and give output node. However, it produces output values in the range of [0, 1].

- **Tangent hyperbolic function**

The hyperbolic tangent function can be used as an activation function in neural networks. The estimation of error effects on weights requires a derivative of an activation function. Hyperbolic tangent function derivative has a simple form that makes it common in neural networks.

This transfer function is described by the following mathematical form:

$$\mathit{tanh}(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})} \quad (3.4)$$

Chapter three: Proposed System Design and Implementation

The functions produce outputs in $[-1, + 1]$ scale. It is a continuous function, in addition. In other words, for every x value, the function generates output. The two functions were used in this work.

2. Learning Rate

The learning rate is a parameter that determines how much modulation must be applied to the weights of a given network to get some desired goals, controls how faster or slower a neural network. A small value of the learning rate η should be applied to convergence the network. Nonetheless, a high learning rate is preferred for fast convergence and local minimal avoidance. For instance, a much too high learning rate would prevent the model from learning effectively. To resolve such two conflicting requirements, trying with a small learning rate and modifying it adaptively. The learning rate must be within the range of 0.01 - 0.1 to produce satisfactory results. In this work, we used the value (0.1-0.9) learning rate, which is good for our problem.

3. Hidden Layers

A hidden layer in an ANN is the layer between the input and output layer, where the artificial neurons perform computations through taking a set of weighted inputs and produce output through apply the activation function. Choosing the number of hidden layers and nodes relies on the network application. Selecting the number of hidden layers is a vital aspect of network architecture and is not as straightforward as layers of input and output. There is no mathematical approach to obtaining an optimum number of hidden layers, as such selection falls within the implementation-oriented classification. Usually, the number of hidden layers should be between the size of the input and output layer, using too few neurons in the hidden layer will result in something called underfitting. however, using too many neurons can result in overfitting this takes place when the neural network has too much information processing ability that the limited amount of data found in the training set is not enough to train all of the neurons in the hidden layer. According to use of supervised learning techniques (ANN algorithm), for real data, one hidden layer with 15 neurons has been tested and established to achieve the best architecture that leads to the best results, and two hidden layers used in simulation data to identify intrusion problem depending on the trial and error approach.

4. Number of Epochs

An epoch represents one cycle through the full training dataset and the iteration is the number of batches or steps through partitioned packets of training data name to indicate the training step. For example, if we have 2000 training examples and the batch size is 1000, then it will take two iterations to complete one epoch.

Moreover, the number of epochs is high, often hundreds or thousands, enabling the learning algorithm to run until the model error has been adequately minimized.

In the proposed system 4000 epochs were used to trained and to achieve accuracy with 0.00001 thresholds.

All parameters and configurations previously described form a set of connected layers that are trained and tested by using data obtained from algorithm 3.1, thus that final accuracy is obtained through the implementation ANN algorithm. Algorithm 3.2 shows the implementation steps of the ANN algorithm.

Algorithm 3.2: ANN Algorithm for Intrusion Detection

Input: Training and Testing datasets.

Output: foretelling Accuracy.

Step 1: Initialize a network n- Input, n-Hidden, n-Output.

`bnn = new BackPropNeuralNet1(numInput, numHidden, numOutput);`

Step 2: Randomly initializing the weights and biases of the neural networks.

`bnn. Set Weights(initWeights);`

Step3: Choose the network structure (like a number hidden layers and the number of neurons for each layer).

Step4: From the training set select the training pairs. Apply an input vector to the network input (calculate sum of product).

Step5: Calculating the activation function for both the hidden layer and the output layer.

Chapter three: Proposed System Design and Implementation

Step6: Compute hidden layer weighted sums.

$$hSums[j] += this.inputs[i] * ihWeights[i][j];$$

Step7: For output of the hidden layer apply the tan activation function.

$$hOutputs[i] = HyperTanFunction(hSums[i]);$$

Step8: Compute output layer weighted sums.

$$oSums[j] += hOutputs[i] * hoWeights[i][j];$$

Step9: Apply log-sigmoid activation to Compute Outputs layer.

$$outputs[i] = Sigmoid Function(oSums[i]).$$

Step10: Calculate the error between network output and the desired output.

Step11: Propagate error backward and change the weights so that the error is reduced. Start from the output layer and return to the input layer.

Step12: Update network weights with error (back-propagation).

- compute output gradients (derivative of log-sigmoid).
- compute hidden gradients (derivative of tan).
- update hidden to output weights.

Step13: Update input to hidden weights (gradients must be computed right-to-left).

Step14: Update hidden to output weights.

Step15: Repeat steps 5–14 for each vector in the training set until the error reaches the minimum required.

Step16: Return Accuracy.

Classification is a procedure of discriminating class data from other data sources in the feature space. In the proposed IDS model, Back-propagation neural network is used to classify the resulted best feature into the appropriate IDS class. Training feedforward neural networks consist of three major steps: -

- 1) Forward propagation.
- 2) Back-propagation of the calculated error.

Chapter three: Proposed System Design and Implementation

3) The weights are updated.

In these studies, we train FFANNs using a backpropagation algorithm. During the training stage, the data is passed into the input layer. The data is transferred to the hidden layer then into the output layer. This is called the forward propagation, in this phase, every node in the hidden layer will get input from all the neurons in the input layer, each input is multiplied by the associated weight. Output of hidden node is non-linearity transformation of that resulting sum. Then compared the output value with the desired values. The error between the actual output values and the desired output is calculated to adjust the weights of the output layer. In a similar way, the network error is also propagated backward by stochastic gradient descent (SGD) to update weights of the previous layers via the network.

Chapter Four
Results and Discussion

Chapter Four

Results and Discussion

4.1 Introduction

In this chapter, the results of the proposed model are present for real data and simulation data, how to evaluate and select the features from the dataset in addition to the values of the parameters used in the experiments to achieve the best possible accuracy. Also, this chapter consist the comparison with other work and discuss the results of the implementation.

The real dataset used in this research has been obtained from a real vehicle and is openly available online. [H.M. Song, H.K. Kim, can network intrusion datasets, <http://ocslab.hksecurity.net /Datasets /car-hacking -dataset.>]

This dataset is the only open dataset that contains normal and attack CAN traffic with labels. In all the experimental results, the run of the system implementation has been done on a Hp laptop Core(TM) i5-7200U CPU @ 2.50GHz, the processor with 8 GB RAM. uses C# language visual studio 2012.

For simulated data using simulated test environment well-known Open Car Test-Bed and Network (OCTANE) [Borazjani P, Everett C, McCoy D, OCTANE: An Extensible Open Source Car Security Testbed, Proceedings of the Embedded Security in Cars Conference 2014)] uses the Mat lap language.

4.2 Datasets for real data

A real vehicle had been used to construct a data set. Datasets were created by logging CAN traffic from a real car via the OBD-II port while message injection attacks were being performed out.

Table 4.1 lists the number of normal messages and messages injected into the data sets of the experiment.

Table 4.1 Overview of datasets

Attack Type	# of messages	# of normal messages	# of injected messages
DoS Attack	3,665,771	3,078,250 (84%)	587,521 (16%)

Dataset was generated by logging CAN traffic while the faked messages were injected in the monitored environment. The dataset included 300 intrusions of messages injected. Each intrusion was performed for 3 to 5 seconds. Dataset included CAN traffic gathered for 30 to 40 minutes. For the DoS attack, designers injected 29-bit CAN ID messages, the most common CAN ID on the CAN protocol, every 0.3 ms.

The primary aim of the DoS attack is lacking the availability of the network. Since the ECU, which tries to send a message with controlling on CAN ID, always success to gain the bus during the arbitration phase, all these other ECUs are stopped from sending their messages.

4.3 Software –OCTANE (Open Car Testbed and Network Experiments)

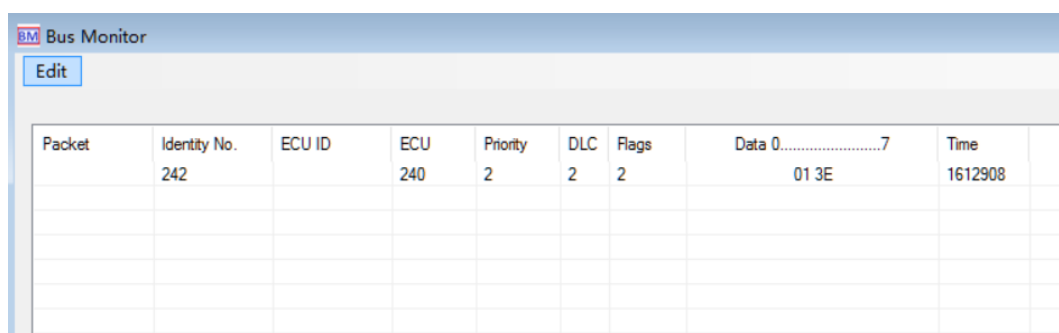
The automotive network testbed consists of a software package that can be used for testing and reverse of automotive networks. It provides a packet monitor and a customized packet transmitter that enables to verify the packet functionalities by injecting the monitored packets to the network.

We can download OCTANE directly from the OCTANE website. OCTANE is a software tool used for cyber-physical security for researchers and students. It reduces the barriers to access to safety research and the teaching of

Chapter four: Results and Discussions

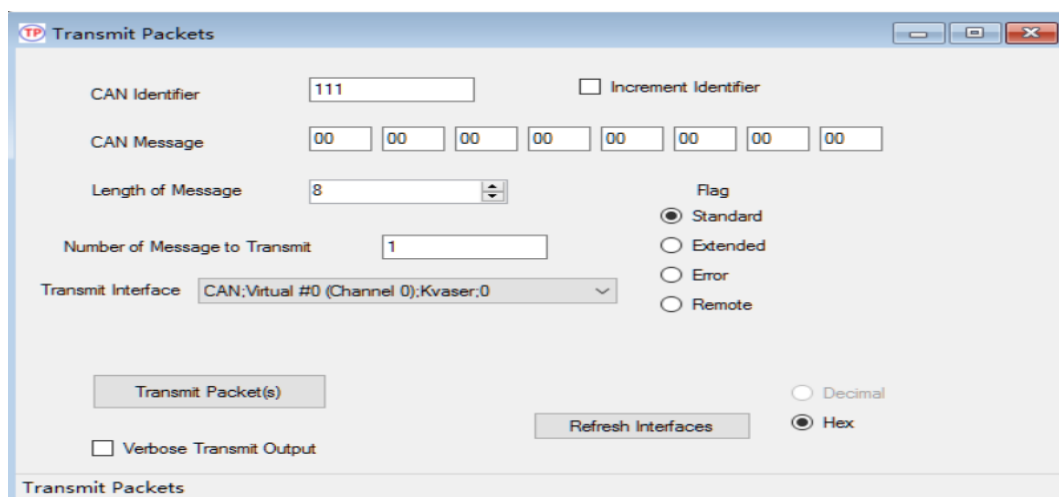
automotive networks through the provision of software packages and testing of automotive networks.

New proposed system offers for security research and teaching by replicating the interactions between the system's hardware components and control software, so that the user may concentrate on the automotive network's security aspects instead of configuring and setting up the device. The package supports monitoring shown in Figure 4.1 and transmitting of CAN protocol shown as Figure 4.2.



Packet	Identity No.	ECU ID	ECU	Priority	DLC	Flags	Data 0.....7	Time
	242		240	2	2	2	01 3E	1612908

Figure 4.1 OCTANE (Bus Monitor).



CAN Identifier: 111 Increment Identifier

CAN Message: 00 00 00 00 00 00 00 00

Length of Message: 8

Number of Message to Transmit: 1

Transmit Interface: CAN:Virtual #0 (Channel 0);Kvaser:0

Flag: Standard Extended Error Remote

Verbose Transmit Output

Decimal Hex

Buttons: Transmit Packet(s), Refresh Interfaces

Figure 4.2 OCTANE (Transmit Packet).

4.4 Performance Evaluation Criteria for IDS

Several evaluation criteria are used to determine the efficiency of the proposed methods. We used the most common performance metrics to determine IDS performance in intra-car networks.

1. Accuracy: (ACC) discloses the ability of the classifier to correctly classify normal and intruder traffic. To determine the feasibility of the proposed system by calculating the accuracy metric based on equation (4.1)[40].

$$\text{Accuracy} = \frac{\text{Number of correctly classified}}{\text{Total number of patterns}} \times 100\% \quad (4.1)$$

The measurements will also be computed as follows to evaluate the classification performance and for comparison with other algorithms[65]:

2. True-Negative (TN): attack connection record classified as an attack. TNR is given as follows:

$$\text{TN Rate}(\text{specificity}) = \frac{TN}{TN + FP} \quad (4.2)$$

3. False-Negative (FN): attack connection record classified as normal. FNR is given as follows:

$$\text{FN Rate}(1 - \text{sensitivity}) = \frac{FN}{FN + TP} \quad (4.3)$$

4. False-Positive (FP): normal connection classified attack. FPR is given as follows:

$$\text{FP Rate}(1 - \text{specificity}) = \frac{FP}{FP + TN} \quad (4.4)$$

5. Error Rate (ER): is the fraction of incorrectly classified frames which are calculated as:

$$\text{ER} = \frac{(FN + FP)}{TP + TN + FP + FN} \quad (4.5)$$

6. True-positive (TP): normal connection record classified as normal. The recall is measured using the following equation[109]:

Chapter four: Results and Discussions

$$\text{Recall or TP Rate} = (1 - FNR) = \frac{TP}{TP + FN} \quad (4.6)$$

For real data, we computed precision and F1-score for compare with other strategies. So, precision is defining as the portion of the real frames of attack between the frames discovered as an attack. High precision corresponds to low FP. Thus, the efficiency of the service should be increased. Precision is measured using the following equation [110]:

$$\text{Precision or positive predictive value}(ppv) = \frac{TP}{TP + FP} \quad (4.7)$$

The F1-score is an equilibrium between precision and recall. So, precision is defining as the portion of the real frames of attack between the frames discovered as an attack. The F-measure value is between 0 and 1, where the value 1 is a good classifier and the value 0 is a bad classifier. Typically, the F1-score is used to evaluate the efficiency of classification that is computed as[111]:

$$F1 = \frac{2 \times (\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})} \quad (4.8)$$

4.5 Evaluation Performance of proposed system

The performance of the suggested IDS model was verified using the data set. The performance of the model is changed by modifying the value of the parameter. One of the parameters is the learning rate, the lower the value of the learning rate leads the model to be in local minimum. In contrast, when the learning rate is too high, the descent gradient does not converge. Therefore, we choose the value of the learning rate 0.1-0.9 for training the IDS model. However, before the training stage, we evaluated and select the feature for real data shown in table 4.2,4.3. Then, we divide the dataset into the training and test data, 80% for training, and 20% for tests.

Table 4.2 illustrates the results of the feature evaluation step, in which the discrimination ability of each is tested within the level of classification. 19 features are shown in the table and FNo. represents feature index in the dataset.

Chapter four: Results and Discussions

Table 4.2 Feature evaluation for real data.

#	FNo.	Bs
f1	0	0.4127
f2	16	0.5081
f3	15	0.5266
f4	3	0.5426
f5	9	0.5473
f6	8	0.5490
f7	7	0.5572
f8	12	0.5628
f9	13	0.5630
f10	10	0.5638
f11	11	0.5638
f12	14	0.5701
f13	6	0.5863
f14	18	1.5811
f15	4	1.9577
f16	5	2.6609
f17	2	3.5368
f18	1	4.8418
f19	17	9.9805

Table 4.3 Feature selection result for real data.

#	FNo.	Bs
f1	0	0.4127
f2	3	0.5426
f3	4	1.9577
f4	6	0.5863
f5	7	0.5572
f6	8	0.5490
f7	9	0.5473
f8	10	0.5638
f9	11	0.5638
f10	12	0.5628
f11	13	0.5630
f12	14	0.5701
f13	15	0.5266
f14	16	0.5081
f15	18	1.5811

Chapter four: Results and Discussions

The confusion matrix from the test on the dataset are shown in Table 4.4. The result in the table reflects the value of TN, FP, FN, and TP, and the number of records applied in the proposed system.

Table 4.4: Classification results for real data of the proposed IDS.

DoS	Predicted normal	Predicted attack
True normal	4804	810
True attack	2	949

Attack Class	IDS			
	ANN	Match Records	Miss Records	Accuracy
Normal	5614	4804	810	85.57%
Abnormal	951	949	2	99.78%
Total	6565	5753	812	87.63%

The most common performance metrics has used to determine IDS performance in intra-car networks, the accuracy, True-Negative (TN), False-Negative (FN), False-Positive (FP), True-positive (TP) and the number of records applied in the proposed system. The result in the table reflects the value of TP (4804), FP (810), and TN (949), FN (2). (6565 samples) collected from real car. the proposed IDS achieves efficient accuracy of 87.63%.

Table4.5: Indicates the classification accuracy and number of records of features used in the simulation.

Attack Class	Simulation Record	IDS			Accuracy
		ANN	Match Records	Miss Records	
Normal	209	256	193	63	92.34%
Abnormal	791	741	728	13	92.03%
Total	1000	997	921	76	91.1%

Chapter four: Results and Discussions

The result in the table 4.5 reflects the value of TP (193), FP (63), and TN (728), FN (13), (1000 samples) collected from simulator. the proposed IDS achieves efficient accuracy of 91.1%.

The detection system computed four alarms rate of internal communication of self-driving cars that were calculated based on Equation (4.2,4.3,4.4, and 4.6) as shown in Table 4.6.

Table4.6: Recognition Rate for simulation.

Alarm Type	Accuracy	Numbers of Connections
True positive	75.39 %	193
True negative	98.24%	728
False-negative	24.60 %	13
False-positive	1.75 %	63

4.6 Comparison with previous work

There is some previous work that used different methods and algorithms. Table 4.7 Describes the classification performance of our ANN-based system that compared to other machine learning methods.

Table4.7: Comparison with other algorithms.

Authors and Literature Reference	Method used	Precision	Recall	FNR %	ER %	F1
	Proposed system	0.855	0.999	0.16	0.006	0.9214
(2011)[112]	Entropy	0.955	0.8884	0.11	NA	0.9205
(2015) [30]	Frequency	0.9373	0.99	0.01	0.01	0.9629
(2016) [71]	CIDS	0.7959	1.0	0	0	0.8864
(2016)[32]	Sequence (LSTM)	0.4571	0.199	0.80	0.81	0.2682
(2018)[113]	Heuristics and Recurrent Neural Networks (RNNs)	0.993	0.992	NA	NA	0.9929

Chapter four: Results and Discussions

(2018)[92]	Hierarchical temporal memory(HTM)	0.997	0.712	NA	NA	0.8307
(2019)[65]	Deep convolution neural network(DCNN)	1.0	0.998	0.1	0.3	0.9995

In table 4.7 The results demonstrate that through the evaluation of the ANN model based on the precision measure is better than Clock-based IDS (CIDS), Sequence (LSTM) whereas Entropy, Frequency, HTM and Deep convolution neural network(DCNN) show high precision. On the other hand, the proposed system has the best Recall compared with other machine-learning models.

while ANN showed better F1-Score compared with Entropy, CIDS, Sequence (LSTM)and Hierarchical temporal memory(HTM). whereas Heuristics and Recurrent Neural Networks (RNNs) and Deep convolution neural network(DCNN) show better but converging result compared with the proposed system.

ANN algorithm, demonstrated good performance with low FNR and ER against DoS attacks.

Deep neural networks such as DCNN showed significant detection performance with low ER and FNR. also compared with other algorithms such as Support Vector Machine (SVM), k-nearest neighbors (kNN), and Naïve Bayes in [65].

Chapter four: Results and Discussions

The Comparison of evaluation results with the previous study is shown in Figure4.3.

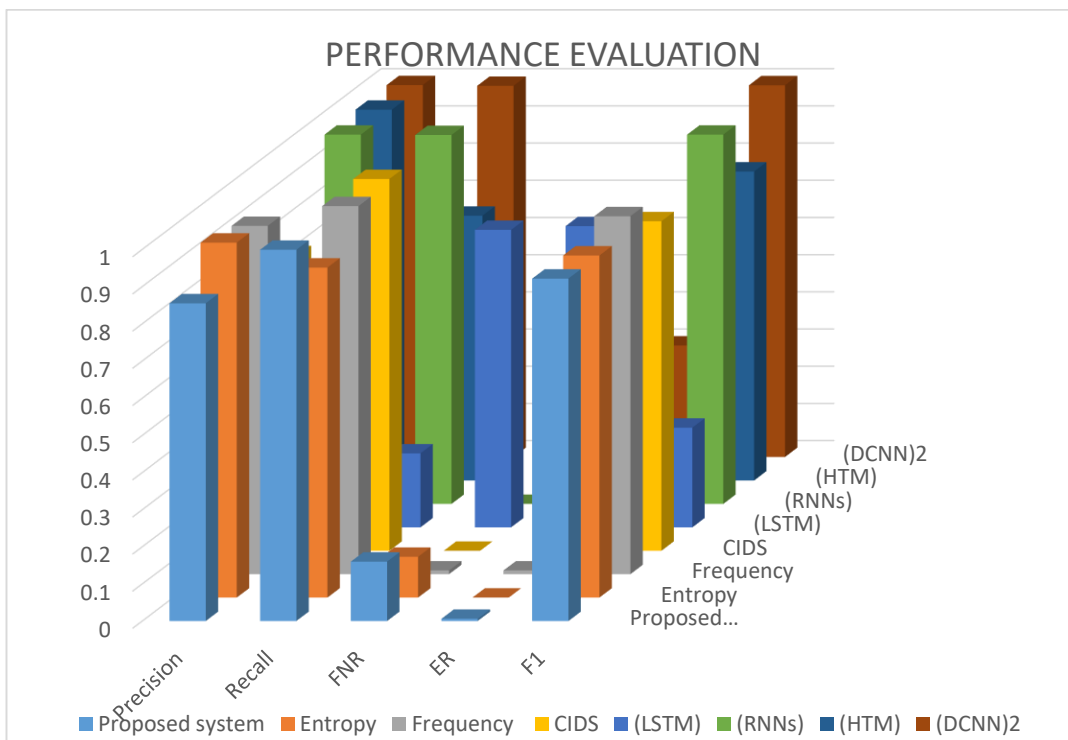


Figure 4.3 Performance Evaluation Comparison between the proposed system and the previous study

Figure 4.4 shows a comparison between the proposed system (simulation) and another system used the (OCTANE) given in[31].

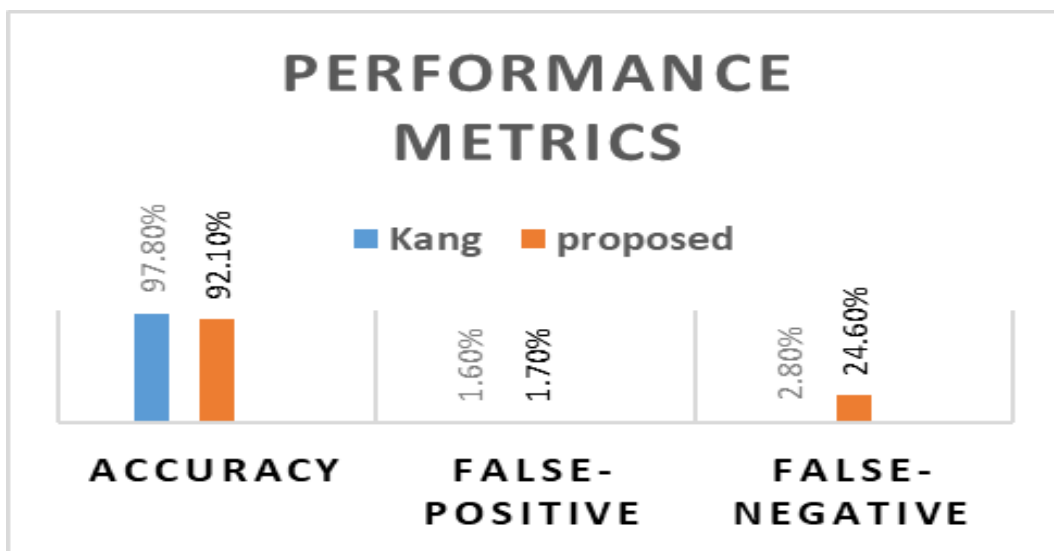


Figure 4.4 Performance Comparison between simulation data.

Chapter four: Results and Discussions

In the proposed system, the accuracy is 92.1% while Kang obtains 97.8%. The false-positive rate 1.6 % while we achieved 1.7 %. So we Obtaining converging and reliable detection results compared with Kang. The false negative is 2.8% while we obtain 24.6% this rate was acceptable, but there is still a need for improvement.

Chapter Five
Conclusions and Future Works

Chapter Five

Conclusions and Future Works

5.1 Conclusions

The proposed intrusion detection system is implemented successfully. The results are summarized as the following:

1. The intelligent intrusion detection system in this thesis is suitable to improve upon the performance of anomaly-based detection methods.
2. Two approaches were suggested for detecting DoS attack on CAN bus (Neural Network with simulation data, Neural Network with real data) with good classification accuracy 92.1 % and 87.63% respectively.
3. Feature clustering step using the K-Means algorithm to class label the features in two clusters normal and abnormal to improve the performance.
4. The system was designed with ANN to deal with the problem of intrusion detection. Train and test the parameters with the feature extracted from the in-vehicular networks, during the training process of real data it was found that a feature generation, enhancing the feature space in this way is a good step to improve the Accuracy.
5. The increase in the number of neurons increases the efficiency of the network. In addition, the increase in the number of an epoch does not greatly affect the accuracy of the model for simulation and real data.
6. Feature evaluation and selection techniques were used to select the best features, Using a limited number of features increases the efficiency of the ANN algorithm in terms of accuracy, the number of parameters used, and time spent on training and testing models to improve the performance.

Experimental evaluation on Real data by evaluating an extremely (6565 samples) show that the proposed IDS achieves good performance with false Negative rate of less than 0.1%, a false-positive rate of 0.9% and the error rate of 0.006

Experimental evaluation on (OCTANE) by evaluating an extremely (1000 samples) show that the proposed IDS achieves acceptable performance to detect attacks with a false-positive rate of 1.75 %, a false-negative rate 24.6%. This system is capable of providing good security for internal communication for self-driving and semi-self-driving cars.

5.2 Future Works

There are a number of works that can be applied in future to improve the proposed system. The most promising of these are:

1. To select the proper feature, machine learning methods can use such as genetic algorithm, Particle Swarm Optimization (PSO).
2. In the classification, various algorithms will have to be designed to address various attacks on the autonomous car such as Support Vector Machine (SVM).
3. Constructing the proposed IDS by using hierarchical classification techniques.
4. Apply the proposed system to other intrusion detection datasets.

References

References

- [1] T.-J. Park, C.-S. Han, and S.-H. Lee, “Development of the electronic control unit for the rack-actuating steer-by-wire using the hardware-in-the-loop simulation system,” *Mechatronics*, vol. 15, no. 8, pp. 899–918, 2005.
- [2] S. Biswas, R. Tatchikou, and F. Dion, “Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety,” *IEEE Commun. Mag.*, vol. 44, no. 1, pp. 74–82, 2006.
- [3] F. Yu, D.-F. Li, and D. A. Crolla, “Integrated vehicle dynamics control—State-of-the art review,” in *2008 IEEE Vehicle Power and Propulsion Conference*, 2008, pp. 1–6.
- [4] S. Tuohy, M. Glavin, C. Hughes, E. Jones, M. Trivedi, and L. Kilmartin, “Intra-vehicle networks: A review,” *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 534–545, 2014.
- [5] S. Yu and Z. Shi, “Fuel consumptions and exhaust emissions induced by cooperative adaptive cruise control strategies,” *Int. J. Mod. Phys. B*, vol. 29, no. 14, p. 1550084, 2015.
- [6] S. Yu and Z. Shi, “Dynamics of connected cruise control systems considering velocity changes with memory feedback,” *Measurement*, vol. 64, pp. 34–48, 2015.
- [7] T. Tang, W. Shi, H. Shang, and Y. Wang, “A new car-following model with consideration of inter-vehicle communication,” *Nonlinear Dyn.*, vol. 76, no. 4, pp. 2017–2023, 2014.
- [8] T.-Q. Tang, W.-F. Shi, H.-Y. Shang, and Y.-P. Wang, “An extended car-following model with consideration of the reliability of inter-vehicle communication,” *Measurement*, vol. 58, pp. 286–293, 2014.
- [9] W.-L. Jin and W. W. Recker, “Instantaneous information propagation in a traffic stream through inter-vehicle communication,” *Transp. Res. Part B*

Methodol., vol. 40, no. 3, pp. 230–250, 2006.

- [10] A. Kesting, M. Treiber, and D. Helbing, “Connectivity statistics of store-and-forward intervehicle communication,” *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 1, pp. 172–181, 2010.
- [11] S. Yu and Z. Shi, “An extended car-following model at signalized intersections,” *Phys. A Stat. Mech. Its Appl.*, vol. 407, pp. 152–159, 2014.
- [12] E. van Nunen, M. R. Kwakkernaat, J. Ploeg, and B. D. Netten, “Cooperative competition for future mobility,” *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1018–1025, 2012.
- [13] K. Lidstrom *et al.*, “A modular CACC system integration and design,” *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1050–1061, 2012.
- [14] M. Yli-Olli and others, “Machine Learning for Secure Vehicular Communication: an Empirical Study,” 2019.
- [15] K. Johansson, M. Törngren, and L. Nielsen, “Handbook of networked and embedded control systems,” *Boston, MA Birkhauser*, 2005.
- [16] C. Bernardini, M. R. Asghar, and B. Crispo, “Security and privacy in vehicular communications: Challenges and opportunities,” *Veh. Commun.*, vol. 10, pp. 13–28, 2017.
- [17] K. Koscher *et al.*, “Experimental security analysis of a modern automobile,” in *2010 IEEE Symposium on Security and Privacy*, 2010, pp. 447–462.
- [18] R. A. Kemmerer and G. Vigna, “Intrusion detection: a brief history and overview,” *Computer (Long. Beach. Calif.)*, vol. 35, no. 4, pp. suppl27--suppl30, 2002.
- [19] S. Woo, H. J. Jo, and D. H. Lee, “A practical wireless attack on the connected car and security protocol for in-vehicle CAN,” *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 993–1006, 2014.
- [20] M. Müter, A. Groll, and F. C. Freiling, “A structured approach to anomaly detection for in-vehicle networks,” in *2010 Sixth International Conference on Information Assurance and Security*, 2010, pp. 92–98.

- [21] P. Tyagi and D. Dembla, "Investigating the security threats in vehicular ad hoc networks (VANETs): towards security engineering for safer on-road transportation," in *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2014, pp. 2084–2090.
- [22] X. Sun, B. Yan, X. Zhang, and C. Rong, "An integrated intrusion detection model of cluster-based wireless sensor network," *PLoS One*, vol. 10, no. 10, p. e0139513, 2015.
- [23] A. J. Deepa and V. Kavitha, "A comprehensive survey on approaches to intrusion detection system," *Procedia Eng.*, vol. 38, pp. 2063–2069, 2012.
- [24] H. Chae, B. Jo, S.-H. Choi, and T. Park, "Feature selection for intrusion detection using nsl-kdd," *Recent Adv. Comput. Sci.*, pp. 184–187, 2013.
- [25] G. Vladimir and P. Kochurko, "Intrusion recognition using neural networks," *Intell. Data Acquis. Adv. Comput. Syst. Technol. Appl. IEEE*, 2005.
- [26] Z. Zhang, J. Li, C. N. Manikopoulos, J. Jorgenson, and J. Ucles, "HIDE: a hierarchical network intrusion detection system using statistical preprocessing and neural network classification," in *Proc. IEEE Workshop on Information Assurance and Security*, 2001, pp. 85–90.
- [27] W. Hu, Y. Liao, and V. R. Vemuri, "Robust anomaly detection using support vector machines," in *Proceedings of the international conference on machine learning*, 2003, pp. 282–289.
- [28] U. E. Larson, D. K. Nilsson, and E. Jonsson, "An approach to specification-based attack detection for in-vehicle networks," in *2008 IEEE Intelligent Vehicles Symposium*, 2008, pp. 220–225.
- [29] N. Kumar and N. Chilamkurti, "Collaborative trust aware intelligent intrusion detection in VANETs," *Comput. Electr. Eng.*, vol. 40, no. 6, pp. 1981–1996, 2014.
- [30] A. Taylor, N. Japkowicz, and S. Leblanc, "Frequency-based anomaly detection for the automotive CAN bus," in *2015 World Congress on*

Industrial Control Systems Security (WCICSS), 2015, pp. 45–49.

- [31] M.-J. Kang and J.-W. Kang, “Intrusion detection system using deep neural network for in-vehicle network security,” *PLoS One*, vol. 11, no. 6, p. e0155781, 2016.
- [32] A. Taylor, S. Leblanc, and N. Japkowicz, “Anomaly detection in automobile control network data with long short-term memory networks,” in *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 2016, pp. 130–139.
- [33] M. Gmiden, M. H. Gmiden, and H. Trabelsi, “An intrusion detection method for securing in-vehicle CAN bus,” in *2016 17th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)*, 2016, pp. 176–180.
- [34] H. M. Song, H. R. Kim, and H. K. Kim, “Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network,” in *2016 international conference on information networking (ICOIN)*, 2016, pp. 63–68.
- [35] M. Markovitz and A. Wool, “Field classification, modeling and anomaly detection in unknown CAN bus networks,” *Veh. Commun.*, vol. 9, pp. 43–52, 2017.
- [36] H. Lee, S. H. Jeong, and H. K. Kim, “OTIDS: A novel intrusion detection system for in-vehicle network by using remote frame,” in *2017 15th Annual Conference on Privacy, Security and Trust (PST)*, 2017, pp. 57–5709.
- [37] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, “Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model,” *J. Comput. Sci.*, vol. 25, pp. 152–160, 2018.
- [38] T. Kuwahara *et al.*, “Supervised and unsupervised intrusion detection based on CAN message frequencies for in-vehicle network,” *J. Inf. Process.*, vol. 26, pp. 306–313, 2018.
- [39] A. Shenfield, D. Day, and A. Ayeshe, “Intelligent intrusion detection systems using artificial neural networks,” *ICT Express*, vol. 4, no. 2, pp. 95–99, 2018.

- [40] M. L. Han, B. Il Kwak, and H. K. Kim, "Anomaly intrusion detection method for vehicular networks based on survival analysis," *Veh. Commun.*, vol. 14, pp. 52–63, 2018.
- [41] P. Sharma and D. P. F. Möller, "Protecting ECUs and vehicles internal networks," in *2018 IEEE International Conference on Electro/Information Technology (EIT)*, 2018, pp. 465–470.
- [42] Q. Wang, Z. Lu, and G. Qu, "An entropy analysis based intrusion detection system for controller area network in vehicles," in *2018 31st IEEE International System-on-Chip Conference (SOCC)*, 2018, pp. 90–95.
- [43] S. M. Kasongo and Y. Sun, "A Deep Learning Method With Filter Based Feature Engineering for Wireless Intrusion Detection System," *IEEE Access*, vol. 7, pp. 38597–38607, 2019.
- [44] A. M. Wyglinski, X. Huang, T. Padir, L. Lai, T. R. Eisenbarth, and K. Venkatasubramanian, "Security of autonomous systems employing embedded computing and sensors," *IEEE micro*, vol. 33, no. 1, pp. 80–86, 2013.
- [45] M. Z. Islam, M. A. Hannan, A. K. M. Waliuzzaman, and A. Ahsan, "Self-driving car: a step to the future of mobility," BRAC University, 2018.
- [46] C. Valasek and C. Miller, "A survey of remote automotive attack surfaces," *IOActive [online] Available http://www.ioactive.com/pdfs/IOActive_Remote_Attack_Surfaces.pdf [Accessed 13 Feb 2015]*, 2014.
- [47] P. M. Bösch, "Autonomous Vehicles-The next Revolution in Mobility," ETH Zurich, 2018.
- [48] T. Litman, "Autonomous vehicle implementation predictions: Implications for transport planning," 2015.
- [49] J. Muller, "No hands, no feet: My unnerving ride in google's driverless car," *Forbes Mag.*, vol. 21, 2013.
- [50] A. Nikitas, E. T. Njonya, and S. Dani, "Examining the myths of connected and autonomous vehicles: analysing the pathway to a driverless mobility

- paradigm,” *Int. J. Automot. Technol. Manag.*, vol. 19, no. 1–2, pp. 10–30, 2019.
- [51] M. S. Al-Kahtani, “Survey on security attacks in Vehicular Ad hoc Networks (VANETs),” in *2012 6th International Conference on Signal Processing and Communication Systems*, 2012, pp. 1–9.
- [52] M. Di Natale, H. Zeng, P. Giusto, and A. Ghosal, *Understanding and using the controller area network communication protocol: theory and practice*. Springer Science & Business Media, 2012.
- [53] J. Edwards and A. Kashani, “Identifying Security Vulnerabilities Early in the ECU Software Development Lifecycle,” 2017.
- [54] S. A. Bagloee, M. Tavana, M. Asadi, and T. Oliver, “Autonomous vehicles: challenges, opportunities, and future implications for transportation policies,” *J. Mod. Transp.*, vol. 24, no. 4, pp. 284–303, 2016.
- [55] H. Wu, “Analysis and design of vehicular networks,” Georgia Institute of Technology, 2005.
- [56] K. Ullah, “Vehicle-to-Infrastructure (V2I) communication,” © *ResearchGate* 2020. All rights reserved. [Online]. Available: https://www.researchgate.net/figure/Vehicle-to-Infrastructure-V2I-communication_fig1_309546589.
- [57] O. Y. Al-Jarrah, C. Maple, M. Dianati, D. Oxtoby, and A. Mouzakitis, “Intrusion Detection Systems for Intra-Vehicle Networks: A Review,” *IEEE Access*, vol. 7, pp. 21266–21289, 2019.
- [58] I. Studnia, V. Nicomette, E. Alata, Y. Deswarte, M. Kaâniche, and Y. Laarouchi, “Survey on security threats and protection mechanisms in embedded automotive networks,” in *2013 43rd Annual IEEE/IFIP Conference on Dependable Systems and Networks Workshop (DSN-W)*, 2013, pp. 1–12.
- [59] Dr.Shiple, “In Vehicle Communication Protocols V2.” [Online]. Available: http://drshiple-courses.weebly.com/uploads/5/2/9/2/52929307/blockdiagram_networking_orig.jpg.

- [60] Dr.Shiple, “Understanding and Using the Controller Area Network Communication Protocol,” 2012.
- [61] W. Zeng, M. A. S. Khalid, and S. Chowdhury, “In-vehicle networks outlook: Achievements and challenges,” *IEEE Commun. Surv. Tutorials*, vol. 18, no. 3, pp. 1552–1571, 2016.
- [62] M. Wolf, A. Weimerskirch, and C. Paar, “Security in automotive bus systems,” in *Workshop on Embedded Security in Cars*, 2004.
- [63] M. Wolf, A. Weimerskirch, and C. Paar, “Secure In-Vehicle Communication,” 2006.
- [64] C. Blommendaal, “Information Security Risks for Car Manufacturers Based on the In-Vehicle Network,” University of Twente, 2015.
- [65] H. M. Song, J. Woo, and H. K. Kim, “In-vehicle network intrusion detection using deep convolutional neural network,” *Veh. Commun.*, vol. 21, p. 100198, 2020.
- [66] S.-F. Lokman, A. T. Othman, and M.-H. Abu-Bakar, “Intrusion detection system for automotive Controller Area Network (CAN) bus system: a review,” *EURASIP J. Wirel. Commun. Netw.*, vol. 2019, no. 1, p. 184, 2019.
- [67] A.-S. K. Pathan, *Security of self-organizing networks: MANET, WSN, WMN, VANET*. CRC press, 2016.
- [68] A. D. Wood and J. A. Stankovic, “Denial of service in sensor networks,” *Computer (Long Beach, Calif.)*, vol. 35, no. 10, pp. 54–62, 2002.
- [69] F. Sakiz and S. Sen, “A survey of attacks and detection mechanisms on intelligent transportation systems: VANETs and IoV,” *Ad Hoc Networks*, vol. 61, pp. 33–50, 2017.
- [70] X. Lin, X. Sun, X. Wang, C. Zhang, P.-H. Ho, and X. Shen, “TSVC: Timed efficient and secure vehicular communications with privacy preserving,” *IEEE Trans. Wirel. Commun.*, vol. 7, no. 12, pp. 4987–4998, 2008.
- [71] K.-T. Cho and K. G. Shin, “Fingerprinting electronic control units for vehicle

- intrusion detection,” in *25th USENIX Security Symposium (USENIX Security 16)*, 2016, pp. 911–927.
- [72] Q. Luo and J. Liu, “Wireless telematics systems in emerging intelligent and connected vehicles: Threats and solutions,” *IEEE Wirel. Commun.*, vol. 25, no. 6, pp. 113–119, 2018.
- [73] T. Hoppe, S. Kiltz, and J. Dittmann, “Automotive it-security as a challenge: Basic attacks from the black box perspective on the example of privacy threats,” in *International Conference on Computer Safety, Reliability, and Security*, 2009, pp. 145–158.
- [74] T. Hoppe and J. Dittman, “Sniffing/Replay Attacks on CAN Buses: A simulated attack on the electric window lift classified using an adapted CERT taxonomy,” in *Proceedings of the 2nd workshop on embedded systems security (WESS)*, 2007, pp. 1–6.
- [75] H. A. Boyes and A. E. A. Luck, “A security-minded approach to vehicle automation, road infrastructure technology, and connectivity,” 2015.
- [76] S. Checkoway *et al.*, “Comprehensive experimental analyses of automotive attack surfaces,” in *USENIX Security Symposium*, 2011, vol. 4, pp. 447–462.
- [77] S. Gillani, F. Shahzad, A. Qayyum, and R. Mehmood, “A survey on security in vehicular ad hoc networks,” in *International Workshop on Communication Technologies for Vehicles*, 2013, pp. 59–74.
- [78] R. Heady, G. Luger, A. Maccabe, and M. Servilla, “The architecture of a network level intrusion detection system,” 1990.
- [79] T. Song, C. Ko, J. Alves-Foss, C. Zhang, and K. Levitt, “Formal reasoning about intrusion detection systems,” in *International Workshop on Recent Advances in Intrusion Detection*, 2004, pp. 278–295.
- [80] M. Bijone, “A survey on secure network: intrusion detection & prevention approaches,” *Am. J. Inf. Syst.*, vol. 4, no. 3, pp. 69–88, 2016.
- [81] “What is Intrusion Detection System (IDS),” 2019. [Online]. Available: <https://www.networkingsphere.com/2019/07/intrusion-detection-system-ids.html%0D>.

- [82] K. M. Alheeti, "Intrusion Detection System and Artificial Intelligent," *Intrusion Detect. Syst.*, p. 117, 2011.
- [83] O. Achbarou, "Network-based Intrusion Detection System architecture."
- [84] J. P. Amaral, L. M. Oliveira, J. J. P. C. Rodrigues, G. Han, and L. Shu, "Policy and network-based intrusion detection system for IPv6-enabled wireless sensor networks," in *2014 IEEE International Conference on Communications (ICC)*, 2014, pp. 1796–1801.
- [85] S. Dave, B. Trivedi, and J. Mahadevia, "Efficacy of Attack detection capability of IDPS based on it's deployment in wired and wireless environment," *arXiv Prepr. arXiv1304.5022*, 2013.
- [86] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Comput. Secur.*, vol. 28, no. 1–2, pp. 18–28, 2009.
- [87] R. Mitchell and I.-R. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 1–29, 2014.
- [88] V. Jyothsna, V. V. R. Prasad, and K. M. Prasad, "A review of anomaly based intrusion detection systems," *Int. J. Comput. Appl.*, vol. 28, no. 7, pp. 26–35, 2011.
- [89] C. Kruegel and T. Toth, "Using decision trees to improve signature-based intrusion detection," in *International Workshop on Recent Advances in Intrusion Detection*, 2003, pp. 173–191.
- [90] C. Herringshaw, "Detecting attacks on networks," *Computer (Long. Beach. Calif.)*, vol. 30, no. 12, pp. 16–17, 1997.
- [91] M. Weber, S. Klug, E. Sax, and B. Zimmer, "Embedded hybrid anomaly detection for automotive CAN communication," 2018.
- [92] C. Wang, Z. Zhao, L. Gong, L. Zhu, Z. Liu, and X. Cheng, "A Distributed Anomaly Detection System for In-Vehicle Network Using HTM," *IEEE Access*, vol. 6, pp. 9091–9098, 2018.
- [93] N. Acharya and S. Singh, "An IWD-based feature selection method for

- intrusion detection system,” *Soft Comput.*, vol. 22, no. 13, pp. 4407–4416, 2018.
- [94] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, “An efficient intrusion detection system based on feature selection and ensemble classifier,” *arXiv Prepr. arXiv1904.01352*, 2019.
- [95] M. Kezih and M. Taibi, “Evaluation effectiveness of intrusion detection system with reduced dimension using data mining classification tools,” in *2nd International Conference on Systems and Computer Science*, 2013, pp. 205–209.
- [96] A. Jović, K. Brkić, and N. Bogunović, “A review of feature selection methods with applications,” in *2015 38th international convention on information and communication technology, electronics and microelectronics (MIPRO)*, 2015, pp. 1200–1205.
- [97] M. S. Pervez and D. M. Farid, “Literature review of feature selection for mining tasks,” *Int. J. Comput. Appl.*, vol. 116, no. 21, 2015.
- [98] G. Chandrashekar and F. Sahin, “A survey on feature selection methods,” *Comput. Electr. Eng.*, vol. 40, no. 1, pp. 16–28, 2014.
- [99] V. Bolón-Canedo, N. Sánchez-Marono, A. Alonso-Betanzos, J. M. Benítez, and F. Herrera, “A review of microarray datasets and applied feature selection methods,” *Inf. Sci. (Ny)*, vol. 282, pp. 111–135, 2014.
- [100] P. Pudil, J. Novovičová, and J. Kittler, “Floating search methods in feature selection,” *Pattern Recognit. Lett.*, vol. 15, no. 11, pp. 1119–1125, 1994.
- [101] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *J. Mach. Learn. Res.*, vol. 3, no. Mar, pp. 1157–1182, 2003.
- [102] A. L. Blum and P. Langley, “Selection of relevant features and examples in machine learning,” *Artif. Intell.*, vol. 97, no. 1–2, pp. 245–271, 1997.
- [103] P. Borazjani, C. Everett, and D. McCoy, “OCTANE: An extensible open source car security testbed,” in *Proceedings of the Embedded Security in Cars Conference*, 2014.

- [104] H. K. K. H.M. Song, “Can network intrusion datasets, [http://ocslab.hksecurity.net /Datasets /car-hacking -dataset.](http://ocslab.hksecurity.net/Datasets/car-hacking-dataset)”
- [105] R. O. Duda, P. E. Hart, and D. G. Stork, “Pattern classification 2nd ed,” *John Willey Sons Inc*, 2001.
- [106] I. Ahmad, A. B. Abdullah, and A. S. Alghamdi, “Application of artificial neural network in detection of DOS attacks,” in *Proceedings of the 2nd international conference on Security of information and networks*, 2009, pp. 229–234.
- [107] M. Zamani and M. Movahedi, “Machine learning techniques for intrusion detection,” *arXiv Prepr. arXiv1312.2177*, 2013.
- [108] B. Karlik and A. V. Olgac, “Performance analysis of various activation functions in generalized MLP architectures of neural networks,” *Int. J. Artif. Intell. Expert Syst.*, vol. 1, no. 4, pp. 111–122, 2011.
- [109] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, “Deep learning approach for network intrusion detection in software defined networking,” in *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, 2016, pp. 258–263.
- [110] S. Kumar and A. Yadav, “Increasing performance of intrusion detection system using neural network,” in *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies*, 2014, pp. 546–550.
- [111] M. E. Elhamahmy, H. N. Elmahdy, and I. A. Saroit, “A new approach for evaluating intrusion detection system,” *CiiT Int. J. Artif. Intell. Syst. Mach. Learn.*, vol. 2, no. 11, pp. 290–298, 2010.
- [112] M. Müter and N. Asaj, “Entropy-based anomaly detection for in-vehicle networks,” in *2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 1110–1115.
- [113] S. Tariq, S. Lee, H. K. Kim, and S. S. Woo, “Detecting In-vehicle CAN message attacks using heuristics and RNNs,” in *International Workshop on Information and Operational Technology Security Systems*, 2018, pp. 39–

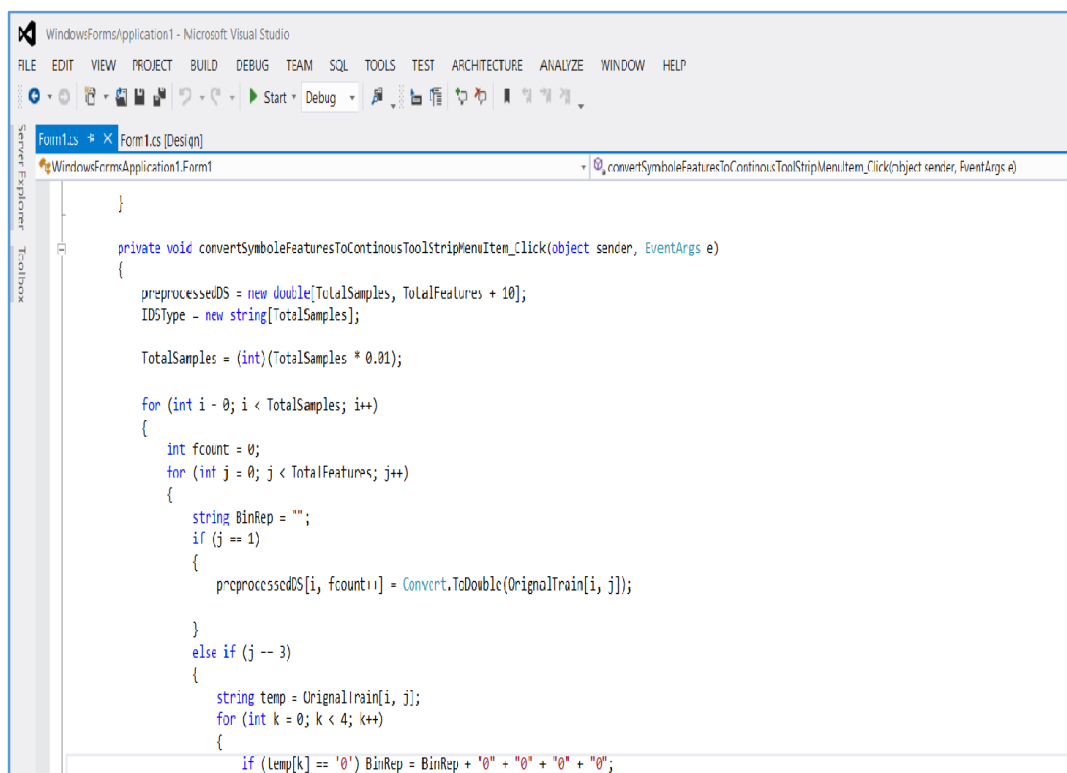
Appendix

Appendix A

A.1 Implementation Part for real data

In this section, the practical part of the proposed system that includes the user environment and so many other information relevant to the implementation part of the proposed system are described in some pictures.

The data preprocessing step is shown in Figure (A.1).



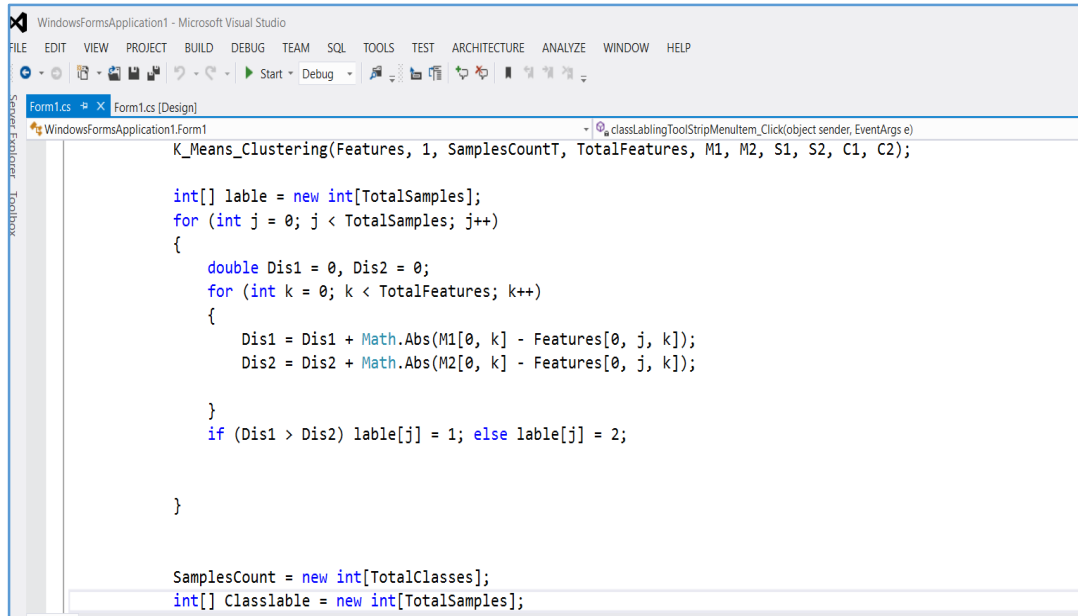
```
WindowsForms/application1 - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Start Debug
Form1.cs [Design]
convertSymbolFeaturesToContinuousToolStripMenuItem_Click(object sender, EventArgs e)
}
private void convertSymbolFeaturesToContinuousToolStripMenuItem_Click(object sender, EventArgs e)
{
    preprocessedDS = new double[TotalSamples, TotalFeatures + 10];
    IDSType = new string[TotalSamples];

    TotalSamples = (int)(TotalSamples * 0.01);

    for (int i = 0; i < TotalSamples; i++)
    {
        int fcount = 0;
        for (int j = 0; j < TotalFeatures; j++)
        {
            string BinRep = "";
            if (j == 1)
            {
                preprocessedDS[i, fcount+i] = Convert.ToDouble(OriginalTrain[i, j]);
            }
            else if (j == 3)
            {
                string temp = OriginalTrain[i, j];
                for (int k = 0; k < 4; k++)
                {
                    if (temp[k] == '0') BinRep = BinRep + "0" + "0" + "0" + "0";
                }
            }
        }
    }
}
```

Figure (A.1) Dataset Preprocessing.

Class labeling for samples using the K-Means algorithm shown in Figure (A.2).



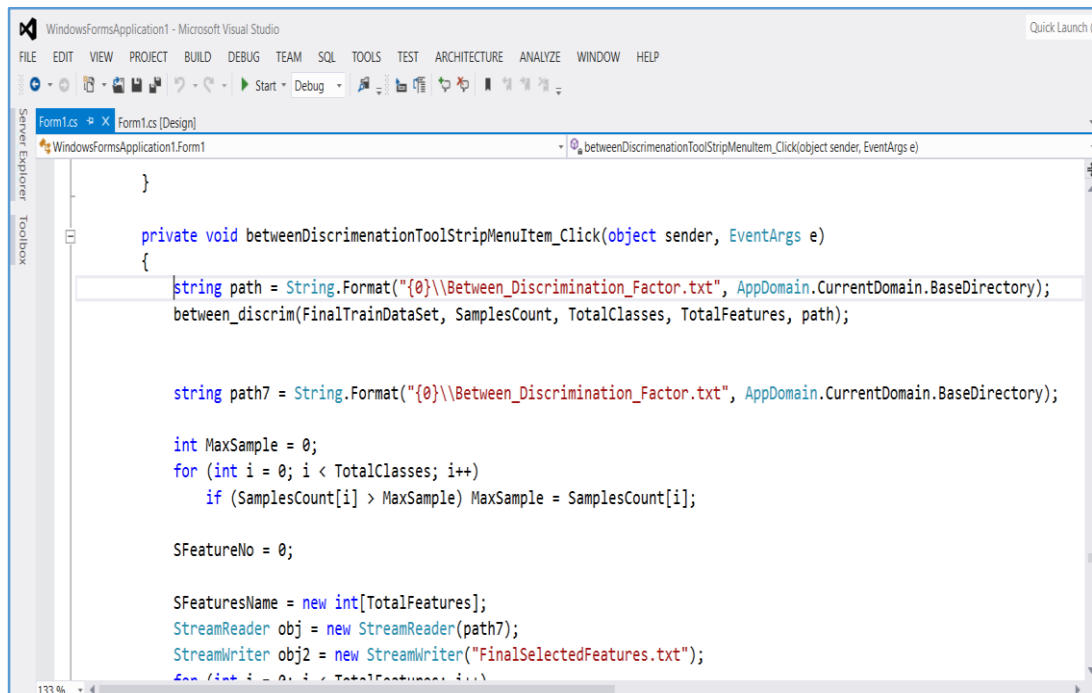
```
WindowsFormsApplication1 - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Start Debug
Form1.cs Form1 (Design)
WindowsFormsApplication1.Form1 classLabelingToolStripMenuItem_Click(object sender, EventArgs e)
K_Means_Clustering(Features, 1, SamplesCountT, TotalFeatures, M1, M2, S1, S2, C1, C2);

int[] lable = new int[TotalSamples];
for (int j = 0; j < TotalSamples; j++)
{
    double Dis1 = 0, Dis2 = 0;
    for (int k = 0; k < TotalFeatures; k++)
    {
        Dis1 = Dis1 + Math.Abs(M1[0, k] - Features[0, j, k]);
        Dis2 = Dis2 + Math.Abs(M2[0, k] - Features[0, j, k]);
    }
    if (Dis1 > Dis2) lable[j] = 1; else lable[j] = 2;
}

SamplesCount = new int[TotalClasses];
int[] Classlabel = new int[TotalSamples];
```

Figure (A.2) class label the feature to normal and abnormal class.

Figure (A.3) and figure (A.4) show the feature evaluation with some of the results.



```
WindowsFormsApplication1 - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Start Debug
Form1.cs Form1 (Design)
WindowsFormsApplication1.Form1 betweenDiscriminationToolStripMenuItem_Click(object sender, EventArgs e)
}
private void betweenDiscriminationToolStripMenuItem_Click(object sender, EventArgs e)
{
    string path = String.Format("{0}\\Between_Discrimination_Factor.txt", AppDomain.CurrentDomain.BaseDirectory);
    between_discrim(FinalTrainDataSet, SamplesCount, TotalClasses, TotalFeatures, path);

    string path7 = String.Format("{0}\\Between_Discrimination_Factor.txt", AppDomain.CurrentDomain.BaseDirectory);

    int MaxSample = 0;
    for (int i = 0; i < TotalClasses; i++)
        if (SamplesCount[i] > MaxSample) MaxSample = SamplesCount[i];

    SFeatureNo = 0;

    SFeaturesName = new int[TotalFeatures];
    StreamReader obj = new StreamReader(path7);
    StreamWriter obj2 = new StreamWriter("FinalSelectedFeatures.txt");
    for (int i = 0; i < TotalFeatures; i++)
```

Figure (A.3) Feature Evaluation for real data.

Between_Discrimination_Factor - Notepad				
File	Edit	Format	View	Help
0		0.412718514978791		
16		0.508167527619027		
15		0.526612298683166		
3		0.542647382800389		
9		0.547319086947587		
8		0.549084589806575		
7		0.557205791709108		
12		0.562815132653659		
13		0.563017156076774		
10		0.563826042242905		
11		0.563826042242905		
14		0.570129503587719		
6		0.586397819465951		
18		1.58110584149196		
4		1.95770009577474		
5		2.66096902123907		
2		3.53681932095637		
1		4.84185097891489		
17		9.98054165270702		

Figure (A.4) Result of feature evaluation

Figure (A.5) and figure (A.6) show all results of training.

```

WindowsFormsApplication1 - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Start Debug
NeuralNetworks Form1.cs* Form1.cs [Design]*
WindowsFormsApplication1.Form1 trainingNNToolStripMenuItem_Click(object sender, EventArgs e)

//Trying all Samples Training
for (int i = 1; i <= 15; i++)
{
    for (double j = 0.1; j <= 1; j = j + 0.1)
        for (double k = 0.1; k <= 1; k = k + 0.1)
            {
                path = String.Format("{0}\\AllRes.txt", AppDomain.CurrentDomain.BaseDirectory);
                StreamWriter SR = new StreamWriter(path, true);
                obj.Init(Features_Count, i, j, k, 4000, 0.00001); //Random Samples
                obj.TrainingAll(MyFeatures, Cc, SamplesCount, Features_Count);
                int S = obj.Testing(MyFeatures, Cc, SamplesCount, Features_Count);
                SR.WriteLine("Hidden No.= " + i + "\t LR= " + j + "\t Mom= " + k + "\t Result= \t" + S);
                SR.Close();
            }
}

```

Figure (A.5) Training.

*AllRes - Notepad					
File	Edit	Format	View	Help	
Hidden No.= 1	LR= 0.1	Mom= 0.1	Result=	5749	
Hidden No.= 1	LR= 0.1	Mom= 0.2	Result=	5749	
Hidden No.= 1	LR= 0.1	Mom= 0.3	Result=	5749	
Hidden No.= 1	LR= 0.1	Mom= 0.4	Result=	5749	
Hidden No.= 1	LR= 0.1	Mom= 0.5	Result=	5749	
Hidden No.= 1	LR= 0.1	Mom= 0.6	Result=	5752	
Hidden No.= 1	LR= 0.1	Mom= 0.7	Result=	5750	
Hidden No.= 1	LR= 0.1	Mom= 0.8	Result=	5750	
Hidden No.= 1	LR= 0.1	Mom= 0.9	Result=	5752	
Hidden No.= 1	LR= 0.1	Mom= 1	Result=	5749	
Hidden No.= 1	LR= 0.2	Mom= 0.1	Result=	5753	
Hidden No.= 1	LR= 0.2	Mom= 0.2	Result=	5750	
Hidden No.= 1	LR= 0.2	Mom= 0.3	Result=	5749	
Hidden No.= 1	LR= 0.2	Mom= 0.4	Result=	5750	
Hidden No.= 1	LR= 0.2	Mom= 0.5	Result=	5750	
Hidden No.= 1	LR= 0.2	Mom= 0.6	Result=	5753	
Hidden No.= 1	LR= 0.2	Mom= 0.7	Result=	5753	
Hidden No.= 1	LR= 0.2	Mom= 0.8	Result=	5750	
Hidden No.= 1	LR= 0.2	Mom= 0.9	Result=	5750	
Hidden No.= 1	LR= 0.2	Mom= 1	Result=	5753	
Hidden No.= 1	LR= 0.3	Mom= 0.1	Result=	5753	
Hidden No.= 1	LR= 0.3	Mom= 0.2	Result=	5750	
Hidden No.= 1	LR= 0.3	Mom= 0.3	Result=	5752	
Hidden No.= 1	LR= 0.3	Mom= 0.4	Result=	5750	
Hidden No.= 1	LR= 0.3	Mom= 0.5	Result=	5753	
Hidden No.= 1	LR= 0.3	Mom= 0.6	Result=	5753	
Hidden No.= 1	LR= 0.3	Mom= 0.7	Result=	5752	
Hidden No.= 1	LR= 0.3	Mom= 0.8	Result=	5753	
Hidden No.= 1	LR= 0.3	Mom= 0.9	Result=	5753	
Hidden No.= 1	LR= 0.3	Mom= 1	Result=	5750	
Hidden No.= 1	LR= 0.4	Mom= 0.1	Result=	5750	
Hidden No.= 1	LR= 0.4	Mom= 0.2	Result=	5750	
Hidden No.= 1	LR= 0.4	Mom= 0.3	Result=	5750	
Hidden No.= 1	LR= 0.4	Mom= 0.4	Result=	5752	

Figure (A.6) All Results of training.

Artificial intelligence used to improve the detection rate and reduce the false alarm rate. The ANN needs to be trained on the normal behavior of the car, which will allow it to detect malicious behavior. Normal behavior is a significant issue during the training phase of the network. However, the data set contains important features that describe normal and abnormal behavior. However, normal and abnormal behaviors are obtained by the real dataset which contains important features that characterize the normal and abnormal behavior of the car. Test /evaluate the standard data set CAN Intrusion Detection Dataset to measure performance efficiency in the detection of abnormal behavior.

Appendix B

B.1 Implementation Part for simulation data

The simulation system has preparing training and test data for measuring the performance of the proposed detection system. The training accuracy is 91.1%. Figures B.1 and B.2 show the performance of the neural network (for simulation data).

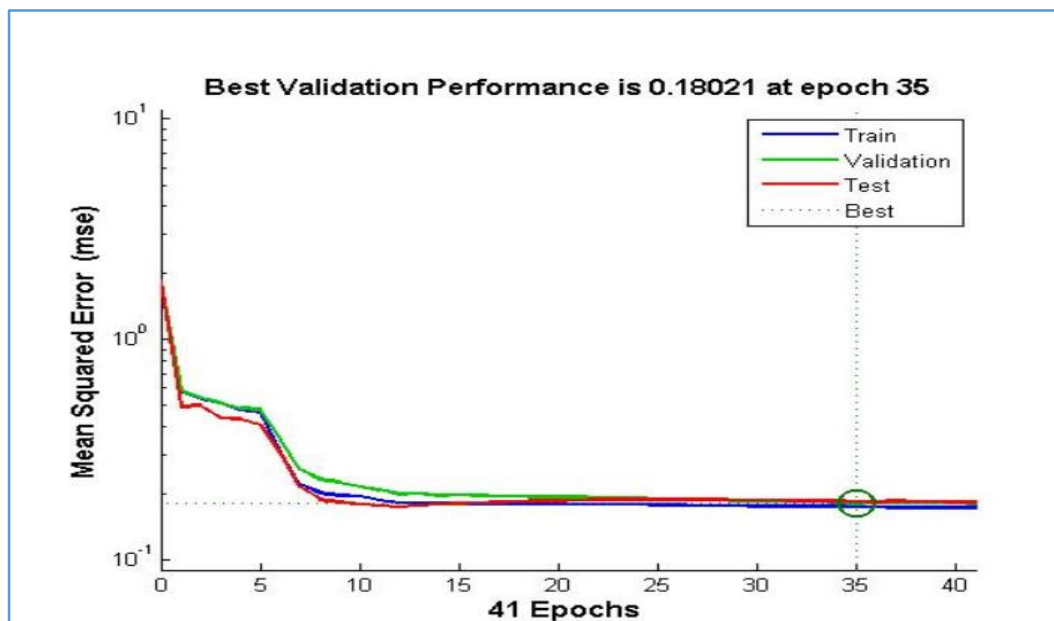


Figure (B.1) Training Performance of ANN.

Figure B.1 shows the mean square error of the neural network training versus training epochs. The drop in mistake was satisfactory. Hence good classification results were expected.

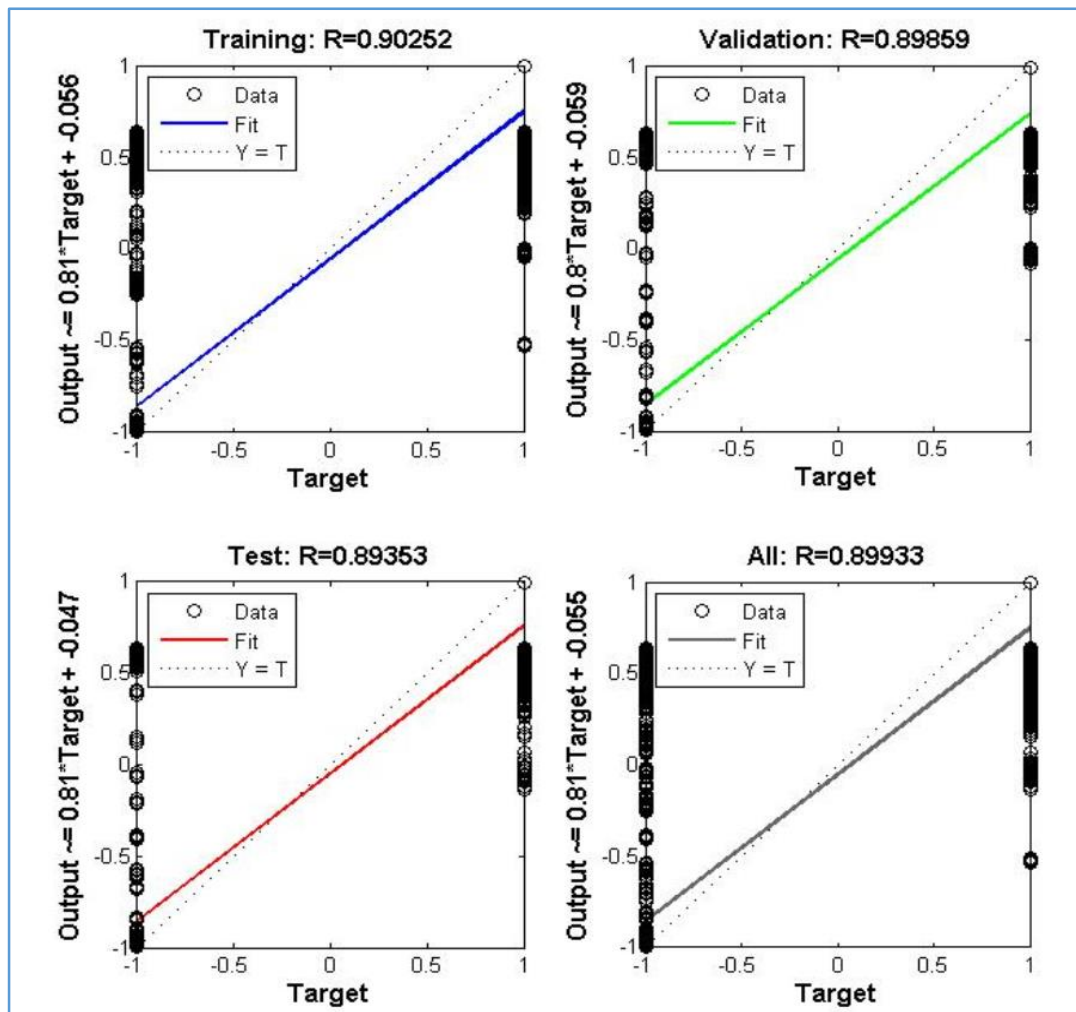


Figure B.2 Training, Testing, and Validation.

Training-validation strategy was used in the present study to improve the ANN's generalization capability. The data is divided into three sub-sets, training, validation and testing, validation set was used to monitor the error though the training. The validation error will reduce usually during the initial training phase. If the validation error for a particular number of iterations increases, the training is stopped. The FFNN was trained with simulation data to obtain the best ratio from the training. Figures B.3 shows all result of training.

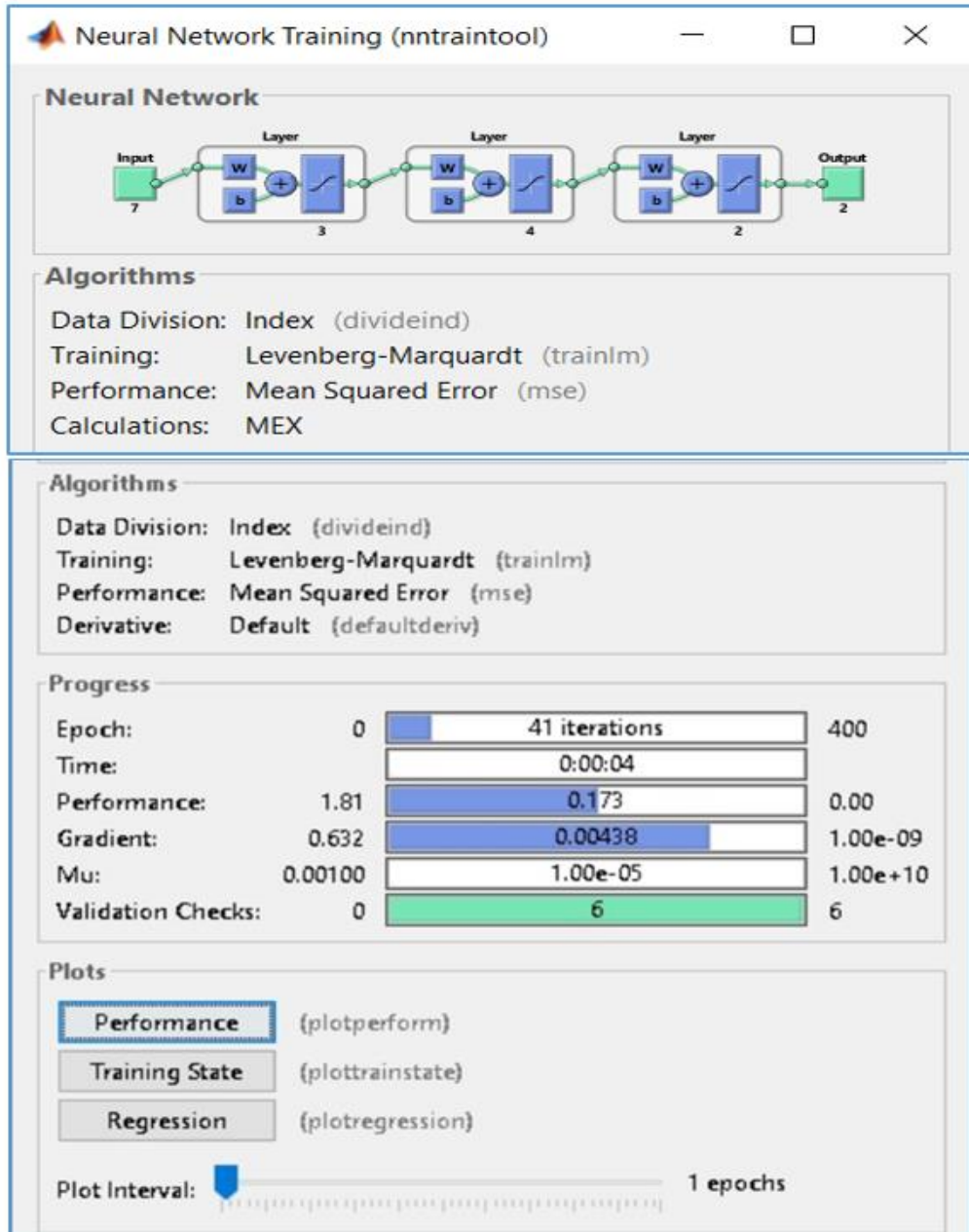


Figure (B.3) Training result for simulation data.

In the training phase of the proposed security system, the initial FFNN parameters play an important role in training accuracy and time. In this system, the epoch parameter is established as a stopping condition with 400 epochs, but according to figure B.3 we note that the ANN received an acceptance training rate of 41 epochs and the number of hidden layers is placed according to the trial and error concept.

الخلاصة

السيارة الحديثة عبارة عن نظام معقد يتكون من وحدات تحكم إلكترونية (ECUs) ، محركات وأجهزة كشف وبروتوكولات تتصل لاسلكياً ، تتواصل وحدات التحكم ECUs من خلال أنواع مختلفة من الشبكات داخل السيارة. حيث يعتمد التصميم الفيزيائي على وحدة التحكم الإلكترونية ، وقد اثبت الباحثين عن أمن المركبات انها تتعرض للعديد من الهجمات من خلال الوصول المادي وعن بعد إلى الشبكة الداخلية للسيارات. تحتوي الشبكة الداخلية على العديد من الثغرات الأمنية التي تجعل من الممكن شن هجمات عن طريق بروتوكول CAN والانتشار الى شبكة وحدة التحكم الإلكترونية بأكملها ، على سبيل المثال منع المحرك من العمل أو قطع الفرامل عن طريق حقن رسائل مصنعة . لذلك ، يمكن لتقنية الكشف عن الأخطاء ، التي تمثل الحماية الأمنية ، تقليل التهديدات الأمنية بكفاءة.

يقترح هذا العمل نظام كشف التسلسل عن الشذوذ (Anomaly Intrusion Detection System (IDS باستخدام الشبكة العصبية الاصطناعية (ANN) لمراقبة حالة السيارة عن طريق الحزم التي تم جمعها من bus لتحقيق الحماية للشبكة الداخلية من خلال تدريب حزمة CAN على اكتشاف الميزة الإحصائية الأساسية من الحزم العادية والهجومية وفي الدفاع ، استخراج الصفة ذات الصلة لتصنيف الهجوم.

إنشاء ميزات جديدة واستخدام خوارزمية التجميع K-mean للتمييز بين العينات المتشابهة ومعالجتها single class لتحسين دقة النموذج. يتم تقييم الميزات لقياس قدرتها على التمييز بين الفئات واختيار أفضل الميزات الموجودة.

أجريت الدراسة التجريبية باستخدام مجموعتين من البيانات ، بيانات المحاكاة Open Car Test-Bed and Network Experiments (OCTANE ومجموعة البيانات الحقيقية لتقييم نظام الكشف. أظهرت النتائج التجريبية لمجموعة بيانات (OCTANE) أن نظام IDS حقق أداءً جيداً بمعدل إيجابي خاطئ 1.7٪ بمعدل سلبي كاذب 24.6٪ ومعدل دقة 92.1٪. يُظهر التقييم التجريبي لمجموعة بيانات حقيقية أن النظام المقترح له معدل منخفض سلبي كاذب يبلغ 0.1٪ ومعدل خطأ 0.006 ومعدل دقة 87.63٪.



جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة الانبار
كلية الحاسوب وتكنولوجيا المعلومات
قسم علوم الحاسبات

كشف التسلل الذكي في أنظمة الاتصالات الداخلية للسيارات

بدون سائق

رسالة مقدمة الى

قسم علوم الحاسبات – كلية علوم الحاسوب وتكنولوجيا المعلومات -

جامعة الانبار

كجزء من متطلبات نيل درجة ماجستير علوم في علوم الحاسبات

قدمت من قبل

نهى عبدالله حمد

بإشراف

أ.م.د. خطاب معجل الهيتي

أ.م.د. صلاح صليبي الراوي