CAN INTRUSION DETECTION SYSTEM USING GAN

A Project Report

Submitted by

Ms. SREELAKSHMI S BAIJU

REG NO : TKM21MEAI10

SEMESTER : IV

In partial fulfillment for the award of the degree of

MASTER OF TECHNOLOGY

IN

Mechanical Engineering (Artificial Intelligence)

Under the guidance of
Dr. ADARSH S

Thangal Kunju Musaliar College of Engineering
Kerala

MAY 2023

DECLARATION

I undersigned hereby declare that the project report "CAN INTRUSION DETECTION SYSTEM USING GAN", submitted for partial fulfillment of the requirements for the award of degree of Master of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of Dr Adarsh S. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Place: Kollam
Date: SREELAKSHMI S BAIJU

Thangal Kunju Musaliar College of Engineering
Centre for Artificial Intelligence

C E R T I F I C A T E

This is to certify that,this report titled ***CAN INTRUSION DETECTION SYSTEM USING GAN*** is a bonafide record of the **Project** presented by **SREELAKSHMI S BAIJU (TKM21MEAI10)**, under our guidance and supervision, in partial fulfillment of the requirements for the award of the degree, **M.Tech in Mechanical Engineering (Artificial Intelligence)** in **APJ Abdul Kalam Technological University** .

Project Guide	Project Coordinator	Head of the Department
Dr Adarsh S Professor Dept. of Civil Engineering	Prof. Chinnu Jacob Assistant Professor Centre for AI	Dr. Imthias Ahamed T P Professor Centre for AI

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

A successful Project is a fruitful culmination of efforts by many people, some directly involved and some others indirectly, by providing support and encouragement. Firstly I would like to thank the almighty for giving me the wisdom and grace for making my Project a memorable one. I thank him for steering me to the shore of fulfillment under his protective wings.

I express my sincere gratitude to **Dr. T A Shahul Hameed**, Principal of T.K.M College of Engineering for his constant support. I would like to thank **Dr. Imthias Ahamed**, Professor and Head of the Department, Centre for Artificial Intelligence, TKMCE, for his constant support and encouragement throughout the project work.

With a profound sense of gratitude, I would like to express heartfelt thanks to my guide, **Dr Adarsh S** , Professor, Department of Civil Engineering, TKM College of Engineering, Kollam and my Project Coordinator, **Prof. Chinnu Jacob**, Assistant Professor, Centre for Artificial Intelligence, TKM College of Engineering, Kollam for their valuable suggestions, expert guidance and immense encouragement. I am grateful to **Prof. Sumod Sundar**, Assistant Professor, Centre for Artificial Intelligence, TKM College of Engineering, Kollam, for his valuable feedback and suggestions, which helped me to complete project to the best of my abilities. I would like to express my gratitude to **Mr. Sreejith Pai P S**, Project Manager and **Mr. Tadepalli Venkata Manikanta Sai Ram**, Project Mentor, Tata Elxsi, for their expert guidance, and cooperation. I also extend my thanks to the entire faculty and staff of the Centre for AI, TKMCE, who has encouraged me throughout this work.

I also express my thanks to my loving parents, brother and friends, for their support and encouragement in the successful completion of this project work.

SREELAKSHMI S BAIJU

Abstract

Throughout the history of automobiles, advancements have been made to improve the safety and comfort of driving. One of the latest developments involves replacing the wiring between electronic control units (ECUs) with a networking standard called a Controller Area Network (CAN). While CAN has proven to be an effective communication protocol, it lacks security features that could not prevent malicious activities on the network. Therefore, there is a need for an Intrusion detection system (IDS) that can monitor CAN network traffic and identify any suspicious behavior. This work proposes an IDS for in-vehicle network communication that can detect both known and unknown malicious activities using deep learning techniques. The proposed IDS is based on Generative Adversarial Networks (GANs) which offers several novel features compared to traditional IDS techniques. The proposed IDS GAN model is evaluated using the Real ORNL Automotive Dynamometer (ROAD) CAN Intrusion Dataset, which contains many network traffic samples. The results shows that the model achieves high accuracy of 99%. Also had done a comparison with different enhanced CNN models to detect known attacks. It is evident from the above experiments that the model based on GANs can effectively detect network attacks and has the potential to be applied in real-world scenarios to enhance network security.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

Technology has made a substantial contribution in recent years to many aspects of our everyday life, particularly in the areas of increased convenience and the simplification of routine activities. Since the beginning of the 20th century, automobiles have played an essential role in modern society and have been subject to a plethora of fundamental shifts. For instance, the use of steam-powered engines in automobiles gave way to the development of gasoline-powered engines, and today the use of electric-powered engines is becoming increasingly widespread. In the past, automobiles were purely mechanical systems, made up only of mechanical components and a few electrical units that were connected to one another by wires. The incorporation of relatively simple embedded systems known as Electrical Control Units (ECUs) was one of the most important developments brought about by this evolution .

In 1983, Bosch was the first manufacturer to introduce the Controller Area Network, or CAN, which allows electronic control units to communicate with one another over a serial bus. Several message formats, error handling techniques, and bus arbitration procedures are outlined in the CAN specification. The CAN bus, on the other hand, was not designed with security concerns in mind when it was developed, and as a result, it is unable to guarantee any security aspects, including non-repudiation, authenticity, availability, integrity, or secrecy. Due to the fact that messages on the CAN bus are sent out in a broadcast fashion, the system is vulnerable to spoofing attacks. Therefore ECUs are unable to validate the authenticity of the messages.

Automobiles are among the most important systems that must adhere to the strictest safety standards because passengers and bystanders could be seriously hurt or killed by a malfunctioning of ECU. As a result, it is critical to develop techniques for detecting such attacks and protecting the in-vehicle network from destruction. Intrusion Detection Systems (IDS) are a standard method for protecting data transmissions in the field of traditional information technology security.

An intrusion detection system (IDS) is a software that focuses entirely on a system or network in order to identify intrusion attempts using a set of rules or patterns. As a result of factors such as limited processing power of memory, compatibility requirements and backward compatibility, engineers working with ECUs must deal with a number of resource constraints. In order to overcome these obstacles, this study suggest integrating a deep

learning-based intrusion detection system (IDS) into vehicle network communication which can detects both known and unknown malicious behavior.

As a result of its effectiveness against all sorts of intruder attacks, intrusion detection systems (IDS) have become an indispensable component in the construction of computer networks, with the aim of preventing early detection of malicious activity. IDS makes decisions about every packet that passes across the network by employing different algorithms to determine whether or not the packet is a normal packet or an attack packet. A computer network can be protected from unwanted users, including potentially malicious insiders, by using software designed to detect network breaches. The objective of the learning task for the intrusion detector is to develop a predictive model that is able to differentiate between so-called "bad" connections (also known as intrusions or attacks) and "good" normal connections.

Therefore this work proposes a Controller area netwoek based IDS using a Generative Adversarial Network(GAN) model capable of detecting the intrusions that occurs in the system. The proposed IDS uses a GAN model to generate normal CAN traffic, which is then compared to the actual CAN traffic using a discriminator. Any deviation between the generated and actual traffic indicates a potential attack. Evaluated the performance of the proposed IDS using Real ORNL automotive dynamometer CAN intrusion detection dataset which contains various attack scenarios. Also it has the potential to enhance the security of in-vehicle networks and ensure the safety of passengers and pedestrians.

1.1 Objectives

- To transform tabular data to images using quantile transformation for identifying the different patterns.
- To propose a GAN-based network intrusion detection system, which allows for the creation of synthetic data that closely resembles real network traffic data which identifies unknown attacks.
- To compare the performance with different deep learning algorithms such as Inception, VGG16, Resnet50, Efficientnet and CNN-BiLSTM.

In this work, a comprehensive hands-on experiment is conducted on the ROAD dataset to evaluate and compare different methodologies for data analysis and processing. The experiment is designed to provide a thorough understanding of the dataset and to identify the strengths and limitations of various analytical approaches. The results of the comparison are presented in detail, including the evaluation metrics used, the performance of each methodology, and the strengths and weaknesses of each approach. This analysis provides valuable insights into the optimal methods for analyzing the ROAD dataset and can inform future research in the field. Overall, this study represents an important contribution to the field of data analysis and provides a foundation for further research in this area.

1.2 Organization of the Report

The rest of this report is organized as follows. Chapter 2 discusses various conventional and deep learning methods used in intrusion detection systems and provides a review of their effectiveness. Chapter 3 is a study of the background information related to the topic being discussed. Chapter 4 presents the proposed model. Chapter 5 provides the implementation details. Chapte 6 presents the results and discussions. Chapter 7 presents the conclusions.

Chapter 2

LITERATURE REVIEW

The following section includes a review of different research works that have been carried out on intrusion detection systems. The studies discussed in this section specifically investigate the application of deep learning and other approaches to the field of intrusion detection.

Shahriar et al. [1] proposed to strengthen the resistance of vehicles to intelligent cyber attacks and to identify sophisticated and stealthy attacks using high-dimensional signal-level CAN data. To deal with missing data in the high-dimensional CAN signal stream, it makes use of a data processing technique that employs a temporary data queue and forward-filling mechanism. The multidimensional signal-level time series data is transformed into numerous images so that the detection problem can be approached as a computer vision problem. This is done so that convolution neural network (CNN)-based models can be used to solve the problem. It is necessary to train multiple CNN-based autoencoders (AE) models so that they can learn the temporal and spatial dependencies present in the data. During the process of reconstruction, any variations from the previously learned patterns might be identified and flagged as possible indicators of an attack. Using the SynCAN datasets, the effectiveness of the proposed approach is examined, and the results are compared with a baseline model to illustrate the improvements in identifying a wide variety of fabrication, masquerade, and suspension attacks on the CAN bus. The results of the experiments reveal that CANShield is highly effective and sensitive when it comes to spotting attacks of this kind.

Hossain et al. [2] proposes a Long Short-Term Memory (LSTM)-based intrusion detection system for in-vehicle Controller Area Network (CAN) bus communications. The proposed system aims to detect abnormal activities, such as attacks, that may occur in the CAN bus communication, which could affect the safety and security of the vehicle.The proposed system utilizes the LSTM neural network architecture to capture the temporal dependencies among the sequential CAN bus data. The system first preprocesses the raw CAN bus data to extract relevant features, which are then fed into the LSTM network for classification. The LSTM network is trained on a labeled dataset of both normal and abnormal CAN bus data, and the trained model is used to detect anomalies in real-time. The proposed system is evaluated on a publicly available dataset of CAN bus data, and it achieves a high accuracy rate in detecting abnormal activities. The paper also compares the proposed LSTM-based system with other intrusion detection systems, such as rule-based and machine learning-based systems, and shows that the proposed system outperforms them in terms of detection

accuracy. Overall, the paper presents a novel approach to intrusion detection in in-vehicle CAN bus communications using LSTM-based neural networks. The proposed system can enhance the safety and security of vehicles by detecting abnormal activities.

Nam et al.[3] developed a generative pre-trained transformer (GPT) model is utilized to learn the pattern of a regular Controller Area Network (CAN) identifier (ID) sequence. Configured a CAN identifier (ID) sequence, it's necessary to collect the identifiers of various CAN signals. This sequence, which can be thought of as a sentence composed of words that take the form of CAN IDs, can be thought of as a sentence. When two GPT networks are combined in a bi-directional way, both historical CAN IDs and upcoming CAN IDs can be used for intrusion detection simultaneously. The proposed model is trained to minimize the negative log-likelihood (NLL) value of the bi-directional GPT network for a normal sequence and to classify a CAN ID sequence as an intrusion if its NLL value exceeds a pre-specified threshold. This allows the model to determine whether or not the sequence is an intrusion. The suggested model achieves greater performance in detecting CAN ID sequences that contain a small number of attack IDs when compared to models that are based on long short-term memory (LSTM) that are already in existence. The suggested model's bi-directional structure maintains the estimated performance for the majority of CAN IDs, despite their positions in the sequence; as a result, it outperforms a single unidirectional GPT model with the same level of complexity.

Hanselmann et al. [4] proposed a new neural network design called CANet for detecting intrusions on the Controller Area Network (CAN) bus, which lacks appropriate security procedures. CANet is based on unsupervised learning and is capable of detecting both known and new intrusion scenarios. The paper evaluates the performance of CANet using both actual and synthetic CAN data and compares it with other machine learning-based approaches. The results show that CANet outperforms the other methods significantly. The paper contributes to the field of intrusion detection for the automotive industry, particularly in the context of the CAN bus, which has been a vulnerable target for cyberattacks. However, the study does not compare the performance of CANet with deep learning-based intrusion detection systems, which have shown promising results in recent years. The paper could benefit from a comparative analysis to identify the strengths and weaknesses of CANet compared to other methods. Additionally, the study does not discuss the impact of the hyperparameters used in the network, which could affect the performance of the model. Further research is required to identify the optimal hyperparameters for CANet.

Nie et al. [5] proposed a data-driven intrusion detection method for the Intelligent Internet of Vehicles (IoV) using a deep convolutional neural network (CNN). The proposed approach uses a time-frequency feature representation of CAN bus signals and a CNN-based classifier for detecting attacks. The authors also proposed a new dataset called IV-IDS, which contains different types of attacks on the IoV network. The experimental results show that the proposed approach achieves higher accuracy and F1 score compared to other traditional and deep learning-based methods. Moreover, the proposed approach is also robust against noise and missing data. The paper concludes that the proposed approach can be used as an effective and reliable intrusion detection system for the IoV.

Laghrissi et al.[6] proposed a new approach to improving the performance of intrusion detection systems (IDS) by incorporating attention mechanisms. The author suggested that attention mechanisms can help IDS to focus on relevant features in network traffic data, while ignoring noise and irrelevant information, leading to more accurate and efficient detection of potential threats. They introduce a new algorithm called IDS-attention, which uses an attention mechanism to prioritize features in network traffic data and improve the accuracy of intrusion detection. From experiments conducted using the NSL-KDD dataset, which demonstrate that IDS-attention outperforms several other commonly used IDS algorithms in terms of both detection accuracy and efficiency.

Hnamte et al. [7] proposed a new approach to network intrusion detection using a two-stage deep learning model that incorporates a Long Short-Term Memory Autoencoder (LSTM-AE). Using traditional intrusion detection systems often rely on signature-based detection, which can be ineffective against new or unknown attacks. To address that limitation, the author proposed a two-stage deep learning model that can effectively identify network intrusions based on patterns and anomalies in network traffic data. The first stage of the proposed model involves using an LSTM-AE to extract high-level features from raw network traffic data. The LSTM-AE is trained on a large dataset of normal traffic data, and is able to identify anomalous traffic patterns based on differences between the trained data and incoming traffic data. The second stage of the model involves using a deep neural network to classify the extracted features as either normal or anomalous. The authors present experimental results from testing the model on several benchmark datasets. This paper is a valuable contribution to the field of intrusion detection and could be useful for researchers and practitioners in the area of network security.

The above mentioned researches summarizes about intrusion detection systems for in-vehicle Controller Area Network (CAN) bus communications. It involves identifying, analyzing, and synthesizing information about different techniques. Also it shows wide varirty of different techniques or methods to identify sophisticated and stealthy attacks. It explains about different types of dataset which helps to indicate whether the traffic contains known malicious patterns or not. Also, it represents a promising direction for improving the accuracy and efficiency of intrusion detection systems, and could have important implications for the development of more effective network security technologies.

While several studies have explored the use of deep learning models for intrusion detection in networks, there is still a research gap in the development of lightweight models that can be implemented on resource-constrained devices. In the above mentioned researches it only focussed on detecting the known attacks on the in-vehicle network, it does not consider the impact of external network attacks or unknown attacks. Also traditional machine learning methods often struggle to effectively analyze complex tabular data with multiple features, which can lead to suboptimal results. So by using an image-based approach, it overcomes limitations and provide a more accurate and efficient method for analyzing complex tabular data. It does not consider the impact of real-world scenarios, such as the presence of noise or variations in the signal, on the performance of the methods.

Chapter 3

INTRUSION DETECTION SYSTEM

This chapter presents information on the background for the work presented in this thesis. It introduces the concepts and technology relevant to the application of intrusion detection system. These concepts include Classification and detection methods of intrusion detection system.

3.1 Classification of Intrusion Detection System

3.1.1 Network Intrusion Detection System (NIDS)

Network intrusion detection systems, also known as NIDS, are installed at a predetermined point inside of the network in order to monitor the traffic coming from all of the connected devices. It does an observation of the traffic that is passing on the entire subnet and then compares the traffic that is being carried on to the collection of known attacks. The administrator can be notified of the warning once an assault has been recognised or once anomalous behaviour has been observed. One example of an NIDS is installing a network intrusion detection system (NIDS) on a subnet that contains firewalls in order to determine whether someone is attempting to breach the firewall.

3.1.2 Host Intrusion Detection System(HIDS)

Host intrusion detection systems, often known as HIDS, are security measures that are implemented on individual hosts or devices that are connected to a network. Only the incoming and outgoing packets from the device are monitored by a HIDS, and the administrator is notified of any suspicious or malicious activity that is uncovered by this monitoring. It then compares the current snapshot with the one that was taken previously after taking a snapshot of the currently active system files. In the event that the analytical system files were altered or removed, the administrator is notified immediately so that they can conduct an investigation. In mission-critical machines, which are not expected to change the arrangement of their components, one can see an example of the utilisation of HIDS.

3.1.3 Protocol-based Intrusion Detection System (PIDS)

A protocol-based intrusion detection system, also known as a PIDS, is made up of a system or agent that always lives at the front end of a server. This agent is in charge of managing and interpreting the protocol that is communicated between a user or device and the server. It does this by continuously checking the HTTPS protocol stream while also accepting the related HTTP protocol in an effort to make the web server more secure. Given that HTTPS does not encrypt data and that it takes some time for a website's presentation layer to load, a system that makes use of HTTPS will need to have its interface located between the two layers.

3.1.4 Application Protocol Intrusion Detection System (APIDS)

The APIDS stands for Application Protocol-based Intrusion Detection System. It refers to a system or agent that often resides within a collection of servers. It does this by monitoring and analysing the communication that takes place on application-specific protocols in order to find the intrusions. For instance, this would carry out a transactional monitoring of the SQL protocol that is explicit to the middleware while it is interacting with the database that is hosted on the web server.

3.1.5 Hybrid Network Intrusion Detection System (HNIDS)

The creation of a hybrid intrusion detection system involves combining two or more distinct methodologies utilised by traditional intrusion detection systems. In a hybrid intrusion detection system, the data from the host agent or system is mixed with the information gathered from the network in order to generate a comprehensive view of the network. When compared to other types of intrusion detection systems, hybrid intrusion detection systems provide superior levels of protection. One example of a hybrid intrusion detection system is "Prelude."

3.2 Detection Methods of IDS

3.2.1 Signature-based Method

Signature-based IDS can identify potential threats in network traffic based on unique characteristics, like a predetermined pattern of bytes or bits. The malware is identified on the basis of the known sequence of malicious instructions it employs. Signatures are the patterns identified by an IDS. While signature-based IDS works well for assaults whose pattern (signature) already exists in the system, it has a much harder time detecting new malware attacks because their pattern is unknown.

3.2.2 Anomaly-based Method

An anomaly-based intrusion detection system was developed with the intention of detecting unknown malware attacks in light of the constant development of new malware. The anomaly-based intrusion detection system makes use of machine learning to develop a trustworthy activity model. Everything that comes in is compared to the model, and an event is deemed suspicious if it is not present in the model. In comparison to signature-based IDS, a solution that is based on machine learning has a more universal property. This is because the models used in machine learning can be developed according to the specific hardware and software configurations.

3.3 Need for IDS

Considering the wide variety of possible threats, constructing a secure network is no easy feat. These days, the usage of computer networks and the services they provide permeates in every aspect of society. Intruder assaults are a major source of anxiety for network administrators and anybody else tasked with keeping users safe online.

Accounts of users, resources of the network, personal information, and passwords are all given the utmost priority by network administrators and security officers when it comes to creating a secure environment. Attackers can carry out their attacks on networks in one of two ways; one of these ways is to make a network service inaccessible for users or to violate personal information. The other option is to compromise the network itself.

Chapter 4

METHODOLOGY

This chapter provides an overview of various deep learning algorithms and outlines a comprehensive approach for detecting attacks.

4.1 Vehicular communication technologies

Different components within the vehicle rely on the collaboration of other devices and sensors to execute their assigned jobs. This necessitates a one-way or multi-directional communication channel between these devices. There are two types of communications.They are **Internal communication** and **External communication.** Internal vehicular communication refers to all communications between the vehicle's internal components. There are other communication technologies that provide a communication link for external devices to execute activities like diagnostics or firmware updates. Furthermore, features that allow passengers to stay connected to the internet are becoming increasingly popular. External communication refers to these types of communications as well as all communications that incorporate an outside party.

4.2 Internal communications

An in-vehicle network is made up of many buses and ECUs that work together to form a network. Almost every function in a contemporary automobile is controlled by one or more Electronic Control Units (ECUs). An ECU is a compact embedded computer system with real-time computing capabilities, time constraints, and minimal power consumption. The primary responsibility of the ECU is to collaborate in order to share sensor data via messages. Multiple communication protocols, such as CAN, LIN, FlexRay, MOST, and Ethernet, connect the ECUs in the car. Figure 4.1 shows how an in-vehicle network can be constructed.

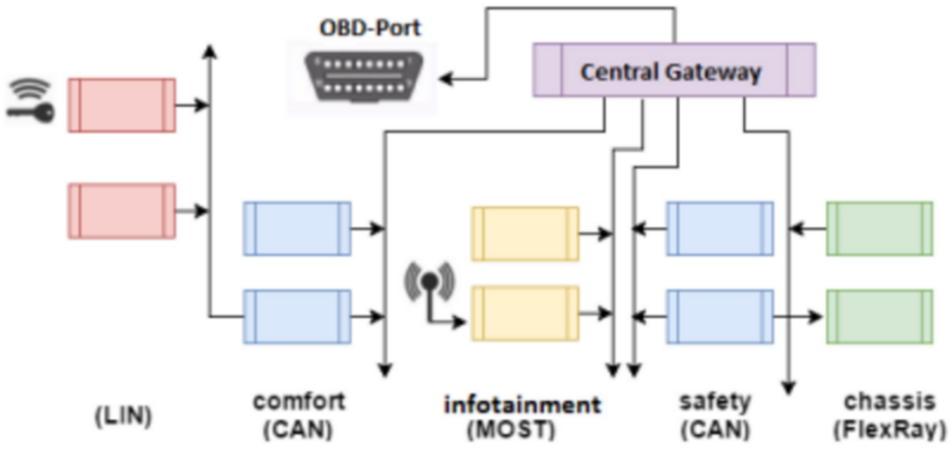

Figure 4.1: Internal communications

1. Controller Area Network(CAN)

The Control Area Network (CAN) is the most widely used bus technology in modern cars, providing a cost-effective and efficient way for electronic control units (ECUs) to communicate with each other. CAN allows ECUs to transmit and receive messages over a single channel, enabling reliable and fast communication between different systems in the vehicle.

2. Local Interconnect Network (LIN)

The Local Interconnect Network (LIN) is a serial bus communication technology that operates similarly to CAN. It is a cost-effective alternative to other bus technologies and is commonly used for applications that do not require precise real-time communication. LIN allows devices to communicate with each other over a single channel, making it an efficient and reliable method for transmitting data in various systems.

3. Media Oriented Systems Transport (MOST)

The Media Oriented Systems Transport (MOST) is a high-speed bus communication standard that is commonly used in multimedia networks within vehicles. MOST employs a ring topology to transmit video and audio data, providing a high bandwidth for efficient data transmission. It is specifically designed for multimedia applications and is optimized for the transmission of large volumes of data.

4. FlexRay

FlexRay is a protocol that offers faster and more reliable communication than CAN, making it an ideal solution for time-sensitive functions like drive-by-wire, brake-by-wire, and backbone systems. Unlike CAN, FlexRay has two-channel communication and supports multiple topologies such as star and ring. However, FlexRay is more expensive compared to CAN. Despite the higher cost, FlexRay provides superior performance and is designed to meet the

demands of high-speed communication in critical applications.

4.3 Controller Area Network(CAN)

The Controller Area Network (CAN) is a bus communication protocol that is commonly used in modern cars and other industrial applications [8] . It was originally developed by Robert Bosch GmbH in the 1980s as a way to simplify wiring and improve communication reliability between different electronic control units (ECUs) within a system. It is standardized as ISO 11898 and ISO 11898-2.

CAN uses a multi-master message broadcast system to enable communication between different ECUs in real-time, allowing for efficient and reliable data transmission. This enables various systems within a vehicle, such as the engine control unit, transmission control unit, and anti-lock brake system, to communicate with each other over a single network, reducing wiring complexity and improving overall system performance. Additionally, CAN is designed to be fault-tolerant, meaning that it can continue to function even if certain components or nodes within the network fail.

The CAN has a data frame structure is divided into different fields, each with a specific function shown in Figure 4.2. The data frame consists of seven fields:

1. **Start-of-Frame (SOF) :** This field marks the beginning of the CAN frame and is represented by a dominant bit (0).
2. **Arbitration Field :** This field is used for message arbitration, where nodes on the bus compete for bus access. The arbitration field contains the identifier (ID) of the message and determines the priority of the message. The ID is either 11 bits (standard frame) or 29 bits (extended frame) long.
3. **Control Field :** The control field includes several bits that define the data length (DLC), error checking (CRC), and other control information.
4. **Data Field :** The data field contains the actual data being transmitted and can be up to eight bytes long.
5. **Cyclic Redundancy Check (CRC) :** This field is used for error detection and consists of 15 bits in the standard frame format or 17 bits in the extended frame format.
6. **ACK Field :** This field is used for error acknowledgment and is sent by the receiving node to acknowledge successful reception of the message.
7. **End-of-Frame (EOF) :** This field marks the end of the CAN frame and is represented by a recessive bit (1). The total length of the standard CAN data frame is 47 bits, while the extended frame is 95 bits.

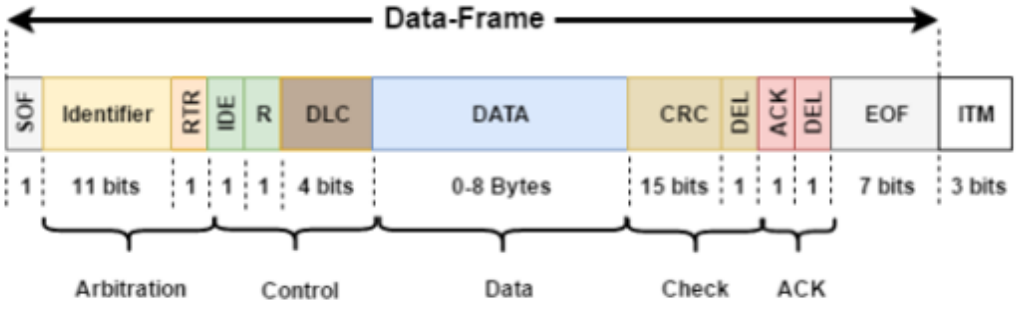

Figure 4.2: CAN Data Frame structure

4.4 Deep Learning Algorithms for Intrusion Detection System

4.4.1 Generative Adversarial Network

Generative Adversarial Networks (GANs) are a type of neural network used in deep learning that can generate new data based on input data. GANs are composed of two neural networks: a generator network and a discriminator network. The generator network creates new data based on the input data, while the discriminator network tries to distinguish between the new generated data and the original input data. During the training process, the generator and discriminator networks are trained together, and the generator network is adjusted to generate better data to fool the discriminator network. The ultimate goal of GAN is to produce synthetic data that is indistinguishable from real data. GANs have been used in various applications, such as anomaly detection, image generation, natural language processing, and drug discovery etc.

4.4.2 Inception

Inception, also known as GoogleNet, is a convolutional neural network (CNN) architecture that was developed by Google in 2014 for the purpose of image classification. It was designed to improve the efficiency and accuracy of CNNs by utilizing a unique architecture with multiple layers.

The Inception architecture is composed of multiple modules, each of which performs a different type of convolution on the input image. These modules include 1x1 convolutional layers, 3x3 convolutional layers, and 5x5 convolutional layers, as well as max pooling and average pooling layers. These modules are combined in different ways to form what is known as an inception module, which is then repeated multiple times to create a deep neural network.

One of the key innovations of the Inception architecture is the use of 1x1 convolutions , which allow the network to reduce the dimensionality of the input image before performing more computationally expensive convolutions. This helps to improve the efficiency of the network while still maintaining high accuracy.

4.4.3 VGG16

VGG16 is a convolutional neural network architecture that was proposed by the Visual Geometry Group (VGG16) at the University of Oxford. It was one of the top-performing models in the 2014 ImageNet Large Scale Visual Recognition Challenge (ILSVRC), which is a benchmark competition for image classification.

The VGG16 architecture consists of 16 layers, including 13 convolutional layers and 3 fully connected layers. The convolutional layers use 3x3 filters with a stride of 1 and a padding of 1. The pooling layers use 2x2 filters with a stride of 2. The network takes an input image of size 224x224 and produces an output vector of size 1000, corresponding to the 1000 categories in the ImageNet dataset.

One of the key features of the VGG16 architecture is its simplicity and uniformity [9]. All of the convolutional layers have the same filter size and stride, and the fully connected layers have the same number of units. This makes the architecture easy to understand and modify, and also makes it possible to train the network using relatively small amounts of data.

4.4.4 Resnet50

ResNet50 is a deep convolutional neural network architecture that was proposed by Microsoft Research in 2015. It is part of the ResNet (Residual Network) family of architectures, which are designed to overcome the degradation problem that occurs when deep neural networks are trained.

The ResNet50 architecture consists of 50 layers, including 49 convolutional layers and 1 fully connected layer. It uses skip connections (also called residual connections) to enable the network to learn residual mappings rather than directly trying to fit the desired underlying mapping. This allows for the training of much deeper neural networks, as the skip connections help to avoid the vanishing gradient problem that can occur with deep networks.

The ResNet50 architecture also includes bottleneck layers, which use 1x1 filters to reduce the number of input channels before applying the larger 3x3 filters. This helps to reduce the computational cost of the network while preserving its accuracy.

ResNet50 has achieved state-of-the-art performance on a variety of computer vision tasks, including image classification, object detection, and semantic segmentation.

4.4.5 Efficientnet

EfficientNet is a family of convolutional neural network architectures that was proposed by Google Brain in 2019. It was designed to achieve state-of-the-art accuracy while also being computationally efficient, making it practical for a wide range of applications.

The EfficientNet architecture uses a compound scaling method to balance model size, computational cost, and accuracy. This involves scaling the network's depth, width, and resolution in a systematic way. The depth and width of the network are increased using a similar approach to ResNet, while the resolution is increased using a new method called "mobile inverted bottleneck convolution."

EfficientNet has achieved state-of-the-art performance on a wide range of computer vision tasks, including image classification, object detection, and semantic segmentation. It has also

been shown to be efficient and effective for transfer learning, where a pre-trained model is fine-tuned on a new task with limited data.

4.4.6 CNN-BiLSTM

CNN-BiLSTM is a neural network architecture that combines a convolutional neural network (CNN) with a bidirectional long short-term memory (BiLSTM) network. It is commonly used for sequence classification tasks, such as anomaly detection and speech recognition.

The CNN-BiLSTM architecture first uses a CNN to extract local features from the input sequence [10], such as individual words in a sentence or individual frames in an audio signal. The CNN applies a set of convolutional filters to the input sequence to capture local patterns and create a set of feature maps.

The output of the CNN is then fed into a BiLSTM network, which is able to capture the context and long-term dependencies in the input sequence. The BiLSTM consists of two LSTM layers, one processing the input sequence forward and the other processing it backward. The output of each LSTM layer is concatenated to create the final output of the BiLSTM.

The concatenated output of the BiLSTM is then fed into one or more fully connected layers for classification or regression. The architecture is trained end-to-end using backpropagation and gradient descent to minimize a loss function, such as cross-entropy or mean squared error.

4.5 Proposed Framework

Intrusion detection system is used as a standalone security tool or as a larger security architecture. It is an essential component for security infrastructure providing protection against different cyber threats by helping organizations to detect and respond to security incidents quickly.

The purpose of this work is to create a vehicle-based intrusion detection system (IDS) capable of identifying a wide range of potential threats. The usual attack scenario and the architecture of an IDS-protected vehicle are depicted in Figure 4.3. Using the On-Board Diagnostics II (OBD II) interface, inter vehicular networks(IVNs) are vulnerable to internal attacks by hackers. Thus, the proposed IDS should be installed in the in-car network. The suggested IDS can be installed on top of the CAN-bus in IVNs to monitor report on any suspicious CAN communications.

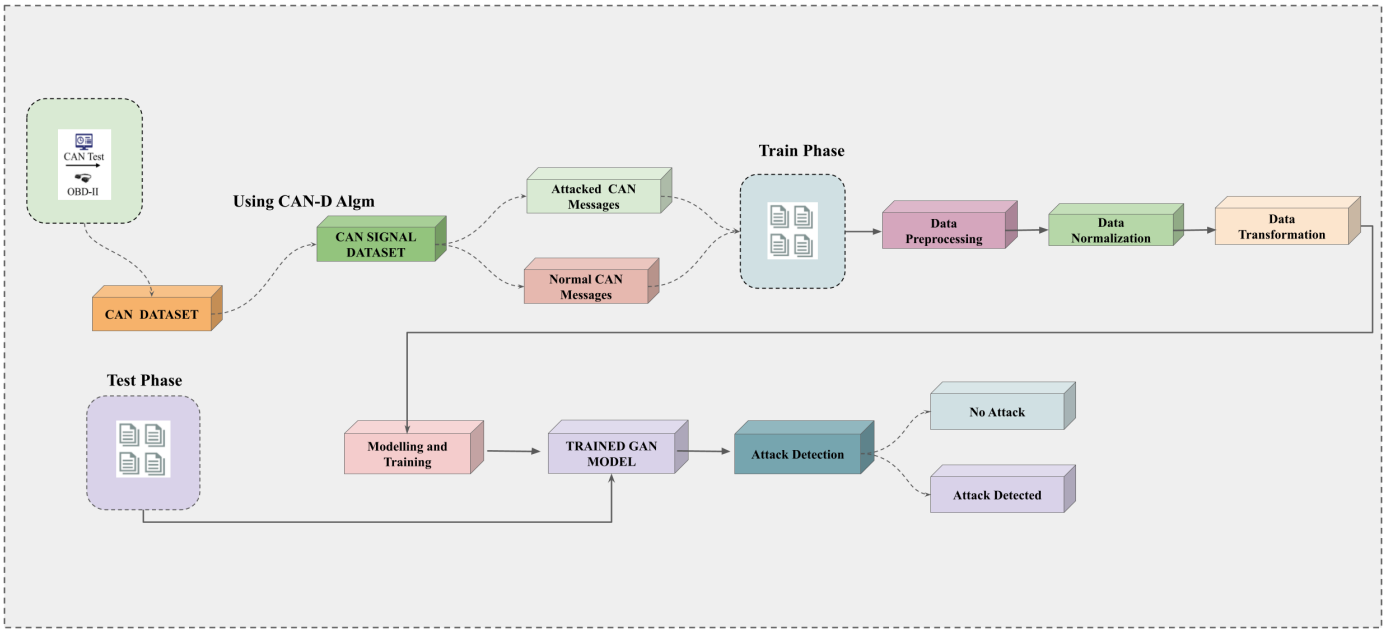

Figure 4.3: Proposed Methodology

Step 1: Data Collection

The ROAD dataset is utilised in the process of developing the proposed IDS for usage in internal vehicular networks. To create it, CAN packets are sent across the CAN-bus of an actual automobile. CAN packets,IDs and their 8-bit data fields (DATA[0]-DATA[7]) are the primary features of the data set. This raw CAN dataset is converted into signals by using CAN-D algm. The time series of captured signals will be provided alongside many of the captured signals that are signal-translated. The input is the decoded signals as time series.

The first step in developing a CAN IDS is to collect a dataset of network traffic data. The data is collected from SocketCAN software installed on a Linux computer, and a Kvaser Leaf Light V2 which is used to connect to an automobile's on-board diagnostics interface. All of the attacks were carried out while the vehicle was being driven on a dynamometer, and data on the surrounding environment was collected from both the dynamometer and the actual roads. It drove in a variety of ways, some of which were unusual, such as with the door open or the seatbelt undone.

When the data has been gathered, it needs to undergo preprocessing before it can be used as an appropriate input for the proposed IDS. The datasets for automotive network traffic are typically presented in tabular form. Inorder to detect complex patterns and anomalies it is important to convert the original network data into image formats. By converting tabular data to images it will help to improve the accuracy of attack detection by leveraging the advanced pattern recognition capabilities of deep learning algorithms.

Step 2: Data Transformation

Data transformation is one of the key aspect of Intrusion Detection Systems (IDS) that involves processing and manipulating data to improve its usefulness for analysis and detection of security threats. The data transformation techniques used in this IDS are quantile transformation and normalisation.

The first step in the process of data transformation is the normalisation of the data. it involves transforming raw data into a standardized format to facilitate accurate comparison and analysis of data. The values of the pixels that make up an image can be anywhere from 0 to 255, which is why the scale used to measure network data should likewise be 0-255. The methods that are most frequently used to transform data values to the same range are the min-max normalisation techniques which is included in the category of normalising techniques.

Quantile transformation technique is also used to convert the tabular data into images [11]. It involves representing tabular data as images by mapping the data values to different colors or intensity levels. It also ensures the distribution of data values is uniform across different ranges or quantiles. In the proposed framework, this approach adjusts all of the feature values to be based on the normal distribution when the feature distribution is converted to a normal distribution. Because of this, the vast majority of the variable's values are relatively close to the median values, which makes dealing with outliers easier.

Once the data has been cleaned and standardised, it is divided into chunks according to the timestamps and feature sizes of network traffic datasets. Hence, the result of each transformation is a three-channel, square colour image (red, green, and blue). When the images are constructed using the timestamps of the data samples, the original network's time-series correlations are preserved.

Following that, the transformed images are labelled depending on the attack patterns. When image contains just normal patterns, we call it as "Normal". If that iamges contains any abnormalities then is said to an "Attack". The most common attack type in a given chunk is then assigned to the image with the most attack samples. Figure 4.4 shows example samples from the dataset for each type of attack.

Steps for Quantile transformation

- Step 1: Apply the quantile transform to each feature in the tabular data. This will transform the data to a uniform distribution between 0 and 1.
- Step 2: Reshape the transformed data into a 2D array, where each row represents an instance and each column represents a feature.
- Step 3: Map each value in the 2D array to a colour using a colour map. Colour map is a function that maps values to colors.

- Step 4: Plot the 2D array as an image, where each pixel represents a value in the array. The color of each pixel is determined by the corresponding value in the array and the chosen color map.

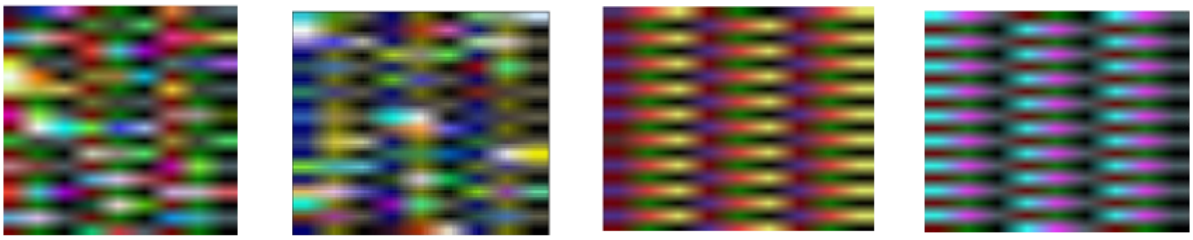

Figure 4.4: Sample images of ROAD dataset

Step 3: Modelling and Training data

In the 3rd step, the model starts training. A GAN model is trained using the pre-processed data to generate synthetic CAN messages that mimic the behavior of real CAN messages and noise in order to generate samples. The GAN consists of two neural networks, a generator network and a discriminator network. The generator network generates synthetic CAN messages, while the discriminator network evaluates whether the messages are real or synthetic. The two networks are trained together in a process called adversarial training, where the generator tries to create synthetic messages that fool the discriminator, while the discriminator tries to distinguish between real and synthetic messages. This procedure was carried on until the loss value of the discriminator is reduced. After the generator was used to create attack samples, those samples were added to the training set.

Step 5: Testing

The trained model was used to predict the classification of the test set using unsupervised learning techniques, where the model learns to identify abnormal patterns without being explicitly trained on labeled data. Using the softmax function, the probability of the predicted classification was computed for comparison with the original labels.

Step 6: Deployment

Once the IDS model is trained, it can be deployed to monitor CAN networks and detect abnormal traffic patterns in real-time.

4.6 Evaluation Matrices

For evaluation purposes, Precision (P), Recall (R), F-measure (F) and Accuracy (ACC) metrics are used. These metrics are calculated by using four different measures, true positive

(TP), true negative (TN), false positive (FP) and false negative (FN).

- TP: the number of anomaly records correctly classified.
- TN: the number of normal records correctly classified.
- FP: the number of normal records incorrectly classified.
- FN: the number of anomaly records incorrectly classified.

Accuracy (AC): The percentage of true detection over total traffic records,

$$AC = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

Precision (P): The percentage of predicted anomalous instances predicted are actual anomalous instances,

$$P = \frac{TP}{TP + FP} \tag{4.2}$$

Recall (R): The percentage of predicted anomalous instances versus all the anomalous instances presented

$$R = \frac{TP}{TP + FN} \tag{4.3}$$

In this chapter, provided an overview of various deep learning algorithms and presented a comprehensive approach for detecting attacks. Also discussed different vehicular communication technologies including internal communication and external communication. Then discussed different communication protocols used in internal communication, including CAN, LIN, MOST, and FlexRay. Further explained CAN in detail and discussed its data frame structure. Then the proposed framework for a vehicle-based intrusion detection system (IDS) is also described in this section. The IDS is aimed at identifying a wide range of potential threats in internal vehicular networks. The proposed IDS framework involves collecting data from vehicular networks, preprocessing and transforming the data into images, and labeling the images according to attack patterns. This IDS can be installed on top of the CAN-bus in IVNs to monitor report on any suspicious CAN communications. Overall, this chapter provides a foundation for understanding the vehicular communication system and methodolgy of intrusion detection system.

Chapter 5

IMPLEMENTATION

5.1 Dataset used

5.1.1 Real ORNL Automoative Dynamometer CAN Intrusion Dataset

It consists of 5 attack captures lasting approximately 30 minutes and 10 ambient captures lasting approximately 3 hours. These takeaways are listed below. We collected CAN data on a Linux computer with SocketCAN software and a Kvaser Leaf Light V2 connected to the OBD-II port [12] . All of the data comes from a single vehicle, the make/model of which we will not reveal, and was manufactured in the mid-2010s. The published data has been obfuscated in such a way that the vehicle's anonymity is preserved while all important aspects of the data for an IDS are preserved on a dynamometer and actively being driven during all of the attacks. Ambient data was collected on a dynamometer and on the road while performing a variety of normal and sometimes unusual driving activities (e.g., driving with an unbuckled seatbelt or an open door).

- Accelerator Attack (In Drive)
- Accelerator Attack (In Reverse)
- Correlated Signal Masquerade Attack
- Reverse Light Off Masquerade Attack
- Reverse Light On Masquerade Attack

5.1.2 Attacks

The following attack captures are listed in order of expected detection difficulty.

1.Targeted ID Fabrication Masquerade Attacks:
Used flam delivery to perform targeted ID fabrication attacks, which means a message is injected immediately after a legitimate message with the target ID is seen. The flam technique supports dynamic injection, which means that the legitimate ID message is read, only the bits corresponding to the target signal are modified with malicious values, and then the

spoofed message is injected. When only a portion of the message is changed.
The following are examples of targeted ID fabrication and masquerade attacks:

- Correlated Signal — The single ID transmitting the four wheels' speeds (each is a two-byte signal) is injected with four false wheel speed values that are vastly distinct from one another. Accelerator has no effect on the vehicle, assuming the injected frames are at least as rapid as the ambient frames with that ID (which is the case in this dataset). This loss of control begins immediately and continues for the duration of the injection. In certain instances, the car required a reset to resume normal operation.
- Reverse Light — Targeted is a binary (one bit) signal that indicates whether the reverse lights are on or off. We carry out two minor modifications of the attack, manipulating the value to off (on), respectively, when the vehicle is in reverse (drive). As a result, the reverse lights do not accurately indicate the gear the car is in.

(2) Accelerator Attacks: The result of this is that the car is in a situation over which the driver has less control, as the following occurs:

- After being placed into drive gear, the vehicle quickens its pace to a predetermined level, which it then maintains (regardless of accelerator pedal position or cruise control setting).
- While in reverse, the vehicle begins to move at a (different) constant speed and maintains this speed throughout the manoeuvre (regardless of accelerator pedal position or cruise control setting).
- The cruise control has been turned off.
- When the brake pedal is pressed, the vehicle's acceleration stops, and the brakes start working as they normally would.
- Once the parking brake is removed, the car will begin to accelerate in the same manner as was previously explained.

While the car is in this mode, the accelerator attack captures do not include any injected messages; instead, they just record the CAN data. There are inconsistencies between the actions taken by the vehicle and the inputs given by the driver. For example, acceleration happens regardless of where the driver's foot is on the accelerator pedal.

5.2 Proposed Model

Generative Adversarial Networks (GANs) are a type of deep learning algorithm the Generator and Discriminator models are adversaries because they compete against each other in a zero-sum game. GANs are used to generate synthetic data that is similar to real data by learning the underlying distribution of the data. The Generator generates fake images in an attempt to trick the Discriminator into thinking they are real. Meanwhile, the Discriminator learns the key characteristics of the images in order to distinguish between actual and

fraudulent examples.

Generative Adversarial Networks attack detection is a technique that is used to identify and detect attacks on GANs, which are models of deep learning that are able to create realistic samples of data. This approach is called "attack detection". The process involves training a classifier to differentiate between genuine and produced samples of data [13] , and then using this classifier to identify and classify attacks on the GAN model that is displayed in Figure 5.1. The categorization of GAN attacks is essential for maintaining the safety and dependability of GANs, which are used in a wide range of applications including the production of images and texts, the augmentation of data, and the identification of anomalies. This increase the resilience and efficiency of these models by identifying and defending against assaults made against GANs.

GANs have gained significant attention in recent years due to their ability to generate realistic synthetic data for a variety of applications. However, GANs can be challenging to train and require careful selection of hyperparameters and network architectures. Additionally, the generated synthetic data may not always be representative of the true underlying distribution of the real data, and caution should be exercised when using synthetic data for critical applications such as security.

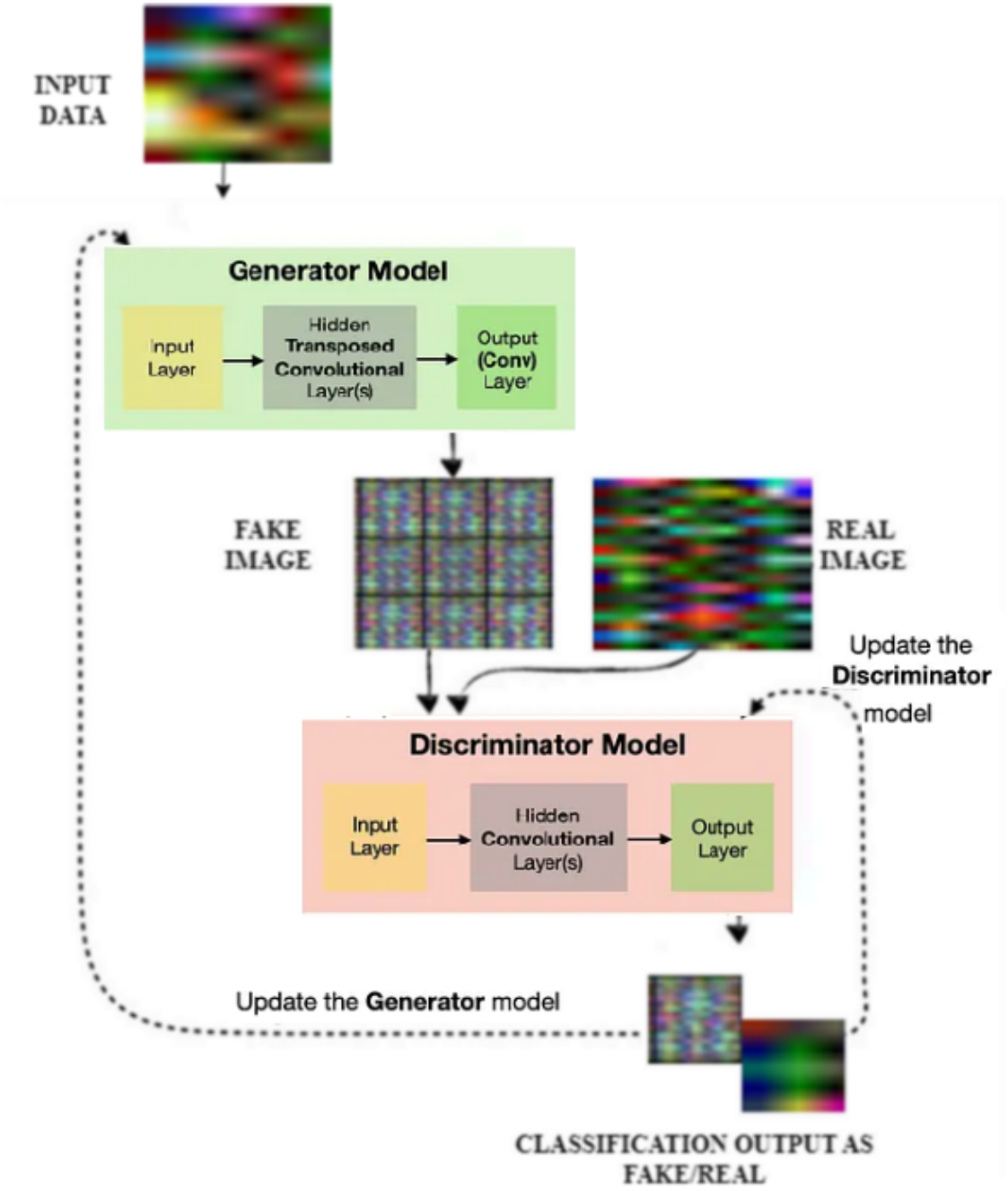

Figure 5.1: GAN Block Diagram

However, GANs are susceptible to a variety of attacks, including mode collapse, in which the generator produces only a limited set of samples, and adversarial attacks, in which an adversary manipulates the input to the generator to produce fake samples that are indistinguishable from real ones. Both of these types of attacks are examples of GAN vulnerabilities. GAN attack categorization is a new study topic that has arisen as a means of detecting and mitigating these kinds of assaults. This is accomplished by first training a classifier to dif-

ferentiate between genuine and produced samples of data, and then utilising this trained classifier to identify and categorise attacks made against the GAN model.

Overall, the classification of GAN attacks is an important research area for ensuring the safety and dependability of GANs, which have numerous applications in fields such as computer vision, natural language processing, and data generation. This is because GANs have become increasingly widespread in recent years.

5.2.1 GAN Model Architecture

Given a set of N images showing normal attacks of image size 64*64. For training the GAN, giving only normal images and train in an unsupervised generative adversarial network to learn the normal images that represent the variability of the training images. Giving samples of unseen images that are distinct attacks of size 64*64 for testing. Labels are also assigned during testing to evaluate the performance of anomaly detection.

A GAN is made up of two adversarial modules, a generator G and a discriminator D. The generator G learns a distribution over normal data. In this case, the network design of the generator G is identical to a convolutional decoder that employs a stack of strided convolutions. D is a standard CNN that converts a 2D picture to a single scalar value.

The generator network accepts input in the form of random noise and produces synthetic samples of data, such as pictures or text, based on that input. Fake Images generated by Generator is shown in Figure 5.2 The generator typically comprises of many layers of linear and non-linear transformations, such as convolutional or recurrent layers as in Figure 5.4. Other types of layers may also be used. The output of the generator is then delivered into the discriminator once it has been processed.

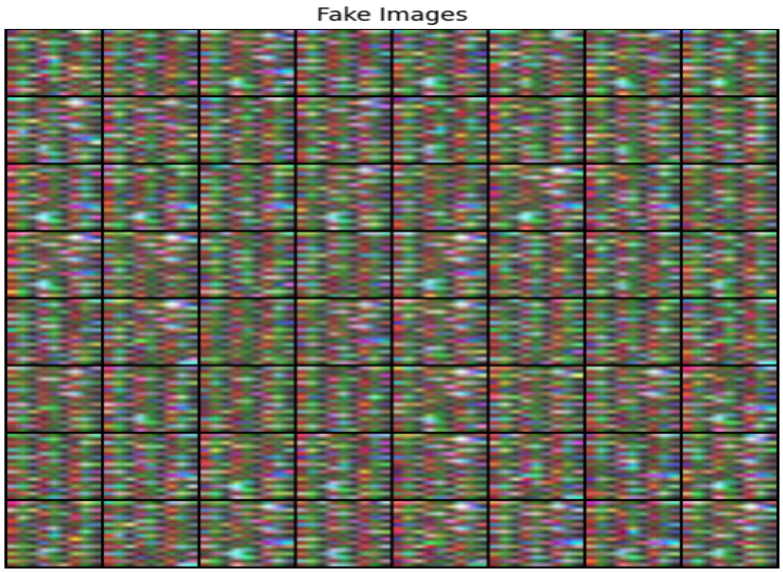

Figure 5.2: Fake Images generated by Generator

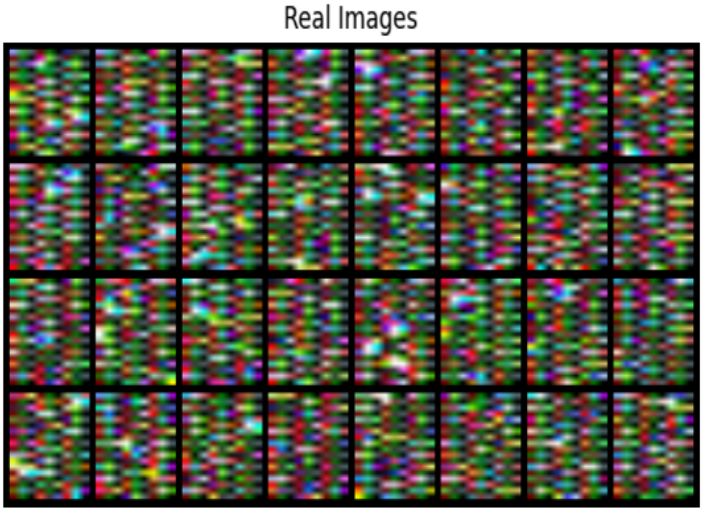

Figure 5.3: Real Images generated by Discriminator

The discriminator network accepts both actual and fabricated data samples as input and attempts to differentiate between them. Real Images generated by Discriminator is shown in Figure 5.3. The discriminator also consists of multiple layers of linear and nonlinear transformations, and is typically trained with binary classification loss to predict whether the input is authentic or fabricated as in Figure 5.5. Discriminators produce outputs based

on the probability that the inputs supplied to them were real images sampled from training data X or generated by the generators. As a result, the generator learns to generate samples that are indistinguishable from real time data, whereas the discriminator learns to effectively differentiate between true and fake samples.

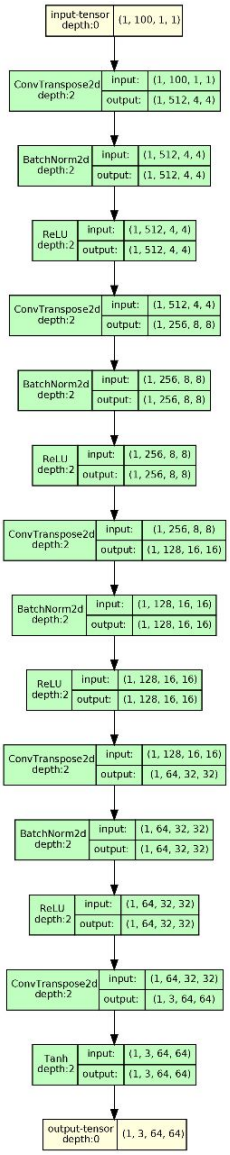

Figure 5.4: Generator architecture

Here is a two-player minimax game with value function V in which value functions G and D are simultaneously optimized.

$$\min_G \max_D V(D,G) = E_{\mathbf{x}\sim p_{\text{data}}\,(\mathbf{x})}[\log D(\mathbf{x})] + E_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \tag{5.1}$$

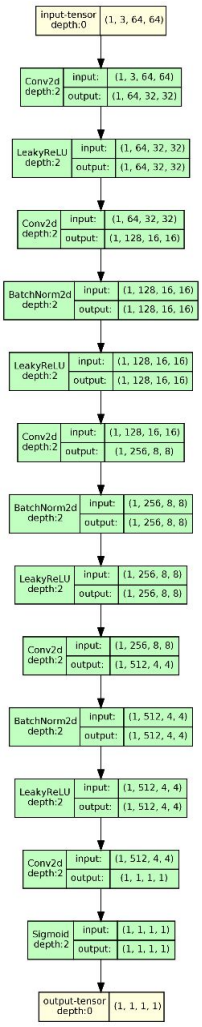

Figure 5.5: Discriminator architecture

The discriminator is trained to maximise the likelihood of labelling actual training examples as "real" and samples from distribution as "fake". Simultaneously, the generator G is being trained to deceive D by minimising

$$V(G) = log(1 - D(G(z))) \tag{5.2}$$

which is equivalent to maximizing

$$V(G) = D(G(z)). \tag{5.3}$$

During adversarial training, the generator improves at generating realistic images, while the discriminator improves at correctly distinguishing between genuine and created images. As a result of generated image and discriminated image, the loss function is defined. Loss function is defined as the mapping of new images which comprises of two components, Discriminator Loss and Residual Loss.

Residual Loss is defined as the difference between the input image and the reconstructed image from generator network G($z\gamma$).

$$LR(z\gamma) = \sum |x - G(z\gamma)| \tag{5.4}$$

Discriminator loss is defined as the difference between the similarity scores of the input images and the generated images.

$$LD(z\gamma) = \sum |f(x) - f(G(z\gamma))| \tag{5.5}$$

5.2.2 Detection of Anomalies

During anomaly detection with test data, determine whether the new query image x is normal or abnormal. So inorder to detect anomaly or attack, here an Anomaly score is defined. Based on that anomaly score, the GAN decides whether the image is normal or attack.

Anomaly score is defined as the difference between the Discriminator Loss and threshold value.

$$A(x) = T - D(x) \tag{5.6}$$

where, A(x) = Anomaly score, T = Threshold value, D(x) = Discriminator loss.

Threshold value is the mean discriminator loss of a set of training images . If anomaly score is greater than the threshold value, then it is a attack. If anomaly score is less than the threshold value, then it is normal.

Therefore, if the model produces a high anomaly score then it will be an attack. If the model produces a low anomaly score then it will be an normal image. So based on the proposed Anomaly score, it will classify whether it is attack or not.

Therefore, to enhance performance and stability, various modifications to the fundamental GAN architecture have been proposed, including the addition of regularisation terms, the use of alternative loss functions, and the incorporation of feedback from the discriminator to the generator. These modifications have resulted in the creation of a vast array of GAN models with diverse applications in fields such as image synthesis, data augmentation, and anomaly detection.

Chapter 6

RESULTS AND DISCUSSION

In this chapter, it presents an overview of the performance of the developed intrusion detection system. Also it then compares with five different classification models such as Inceptionet, VGG16, Resnet50, Efficienet, CNN-BiLSTM. The comparison results shows that all the five models achieve high accuracy and correctly classifies in the case of all the known attacks. But in the case of unknown attack, these algorithms will not detect any unknown attacks. So, there comes the advantage of using GAN. GAN correctly classifies all the known and unknown attacks that occurs in the network.

6.1 Dataset Used

The performance of the proposed model is validated on the dataset by Real ORNL Automoative Dynamometer CAN Intrusion Dataset. The dataset was composed of 165000 samples which is randomly divided into training and testing set. The dataset consists of 100000 rows and 19 features in training dataset and 65000 rows and 19 features in testing dataset. Training dataset consists of normal samples only. Test dataset consists of both normal and attack samples. Figure 6.1 shows some examples of sample images from the dataset.

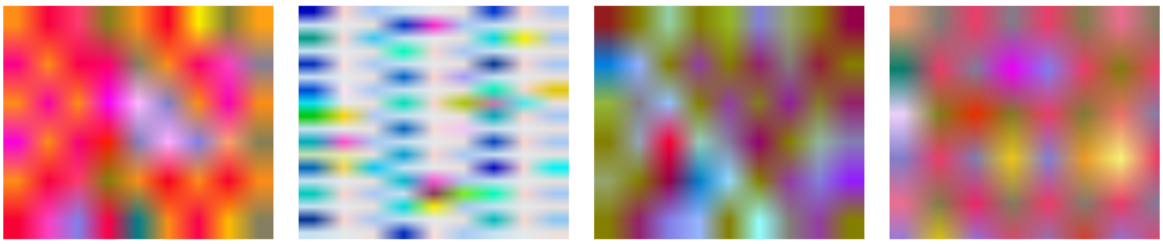

Figure 6.1: Sample images of each class in ROAD dataset

6.2 Experimental Setup

From this experiments it is confirmed that the proposed model has good classification performance. The suggested model is implemented in PYTHON 3.7 using the required libraries,

such as Scikit-learn, pandas, NumPy, Keras, Tensorflow, and others. The experiments are coded and it is implemented on Intel Core i7 6200CPU, 2.40 GHz computer with an 16 GB memory. Then, GAN is established by TensorFlow of Python.

6.3 Results

Five models' performances are compared with the proposed GAN model. The five models are Inception, VGG16, Resnet50, Efficienet, CNN-BiLSTM.

From the comparison results of accuracy, as shown in Table 6.1, it can be seen that the GAN model performs best when compared to other models. GAN model achieved a accuracy of 99% and correctly classifies unknown attacks.

Table 6.1: Accuracy evaluation of models on ROAD dataset

Method	Train-Data	Test-Data
GAN	**0.998**	**0.987**
Inception	0.990	0.751
Resnet50	0.982	0.633
Efficientnet	0.997	0.603
CNN-BiLSTM	0.958	0.761
VGG16	0.991	0.702

6.4 Results of Proposed Model

Figure 6.2 shows Real Image generated by Discriminator and Fake Image generated by Generator obtained for test data.

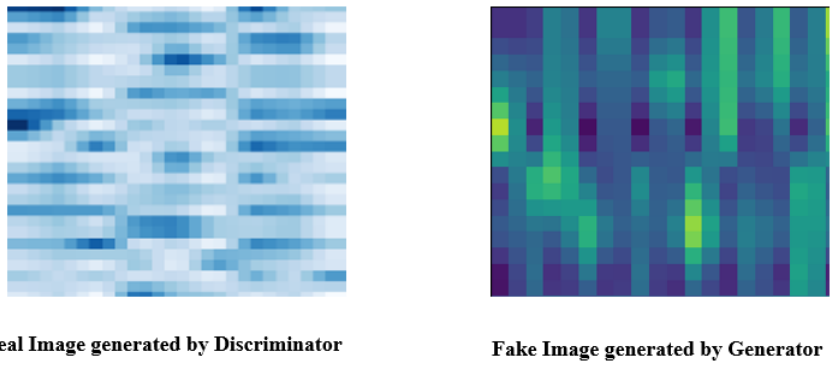

Figure 6.2: Images generated by Discriminator and Generator

From the results of accuracy, as shown in Table 6.2, it can be seen that the GAN has achieved a Discriminator accuracy of 99.99%.

Table 6.2: Accuracy scores of GAN model

Model	Discriminator Loss	Discriminator Accuracy	Generator Loss	Generator Accuracy
GAN	0.26	0.99	7.1	3.5

Figure 6.3 shows the Discriminator loss and Generator loss plots for GAN. The blue line indicates Generator loss, and the orange line indicates Discriminator loss.

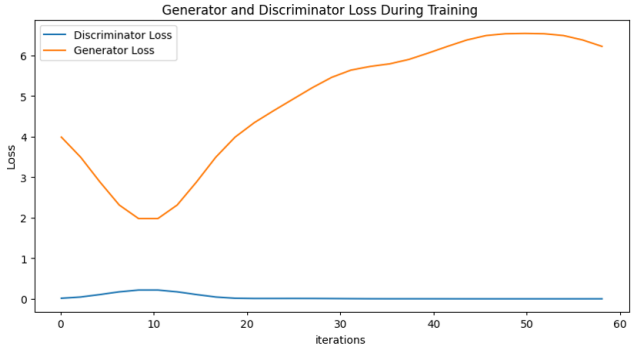

Figure 6.3: Generator and Discriminator Loss Graph

Figure 6.4 shows the Discriminator loss and Generator loss plots for GAN along with the **threshold value = 0.3.**

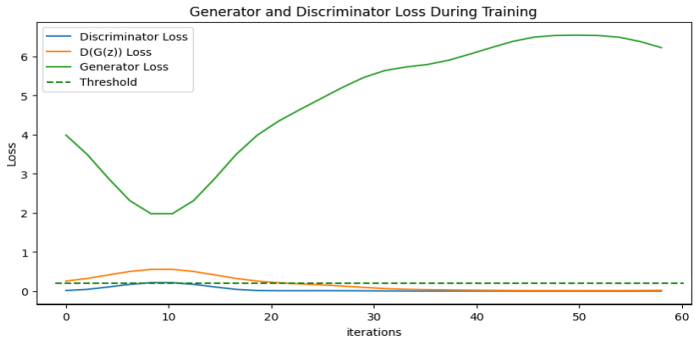

Figure 6.4: Generator and Discriminator Loss Graph with threshold value

Based on this threshold value, GAN decides whether it is a attack or not. After setting the threshold value, the anomaly score is calculated. If the anomaly score is greater than the threshold value then it indicates it is a attack. And if the anomaly score is less than the threshold value it indicates it is normal.

Figure 6.5 shows the Anomaly score obtained after training. The obtained score is **0.039**.

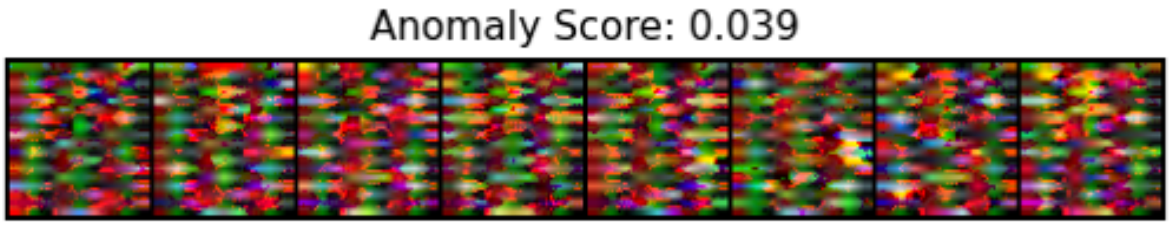

Figure 6.5: Anomaly score of training data

Figure 6.6 shows the Anomaly score obtained after testing. The obtained score is **0.823**.

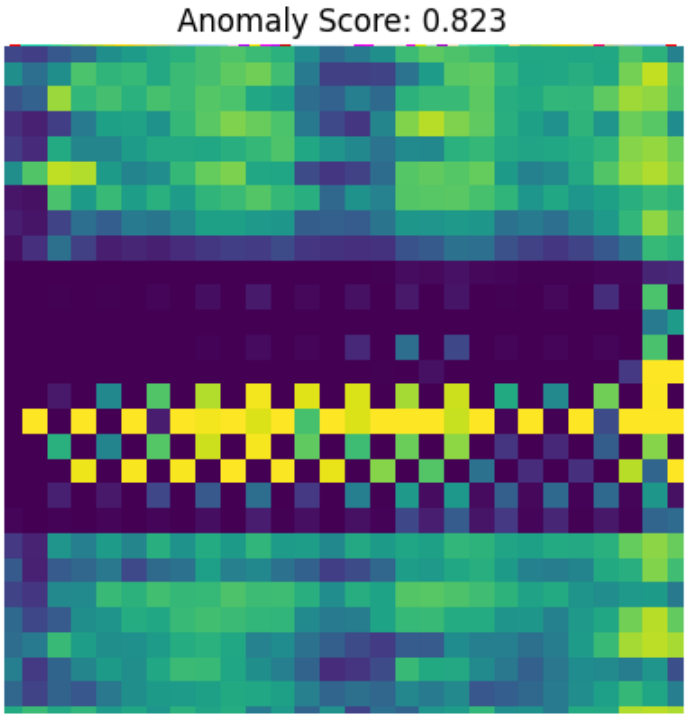

Figure 6.6: Anomaly score of test data

So, the anomaly score obtained for test data is greater than the given threshold value. Therefore, it is clear that the given test data classifies the image into the catergory of an attack.

Figure 6.7 shows the predicted results of Normal and Attack.

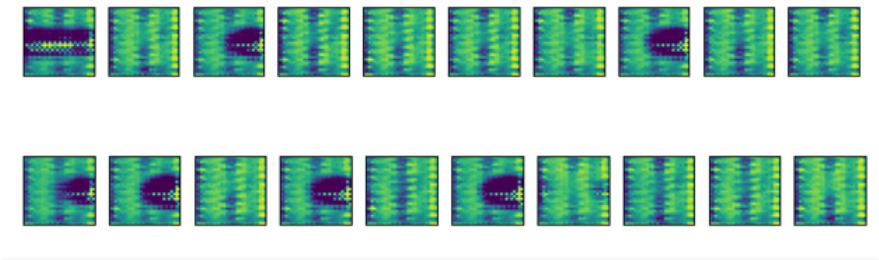

Figure 6.7: Predicted result of Normal and Attack

6.5 Comparison results of other model

6.5.1 Results of Inception

From the results of accuracy, as shown in Table 6.3, it can be seen that the **Inception** model achieves an accuracy of 0.99%.

Table 6.3: Performance measures of INCEPTION model

Performance measures	Train Data	Test Data
Accuracy	0.990	0.751
Precision	0.985	0.780
Recall	0.982	0.752
F1-Score	0.991	0.730

The accuracy graphs of the inception models are shown in Figure 6.8 respectively. The blue line indicates validation accuracy, and the orange line indicates training accuracy.

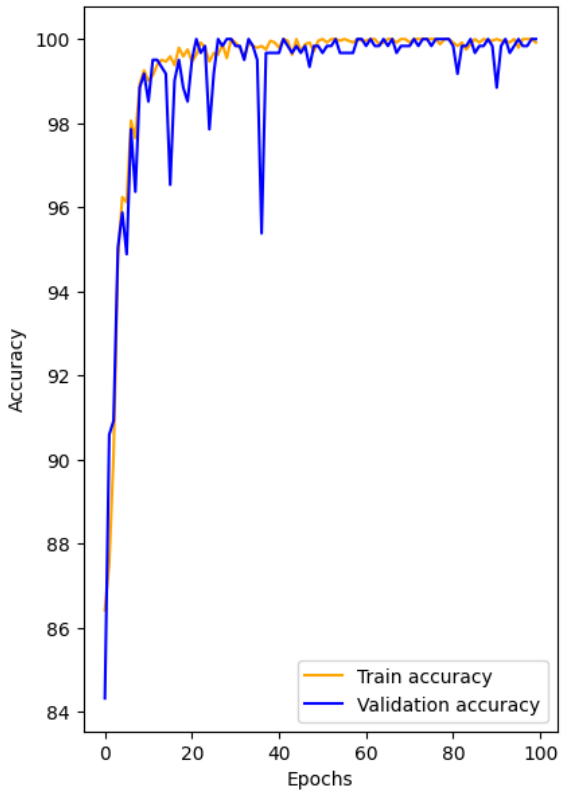

Figure 6.8: Accuracy graph plot of Inception

The loss graphs of the inception models are shown in Figure 6.9 respectively. The blue line indicates train loss, and the red line indicates validation loss.

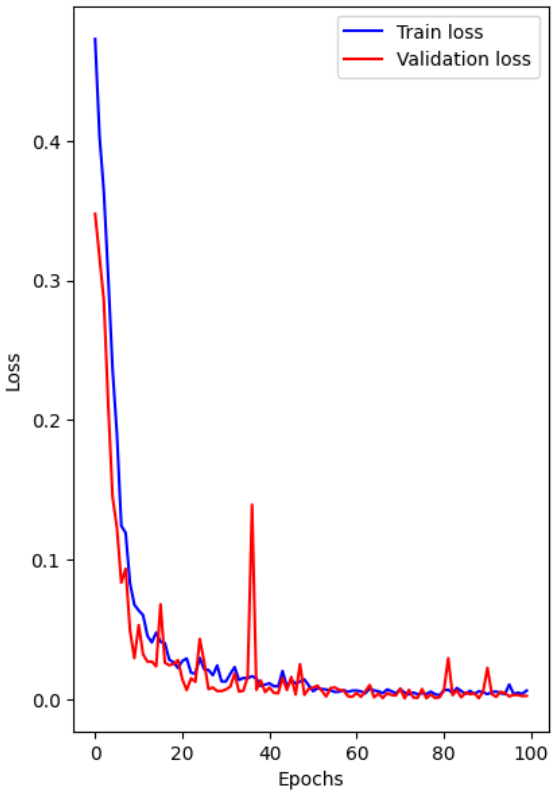

Figure 6.9: Loss graph plot of Inception

A confusion matrix is a graphical representation and summary of the results of a classification process. A confusion matrix is used to improve the classification algorithm's performance.

Here a multi-class confusion matrix is used to evaluate the performance of a classification model that has more than two classes. It provides a visual representation of how well the model predicted the correct class for each observation, and how often it misclassified it. Each row represents the actual class, while each column represents the predicted class. The diagonal cells indicate the number of correct predictions, while the off-diagonal cells indicate the number of misclassifications. In the case of an intrusion detection system, the confusion matrix can help identify which types of attacks were misclassified and which ones were correctly identified, providing valuable information for improving the system's performance.

The confusion matrix of both known attack and unknown attack is given in Figure 6.10 and Figure 6.11 respectively.

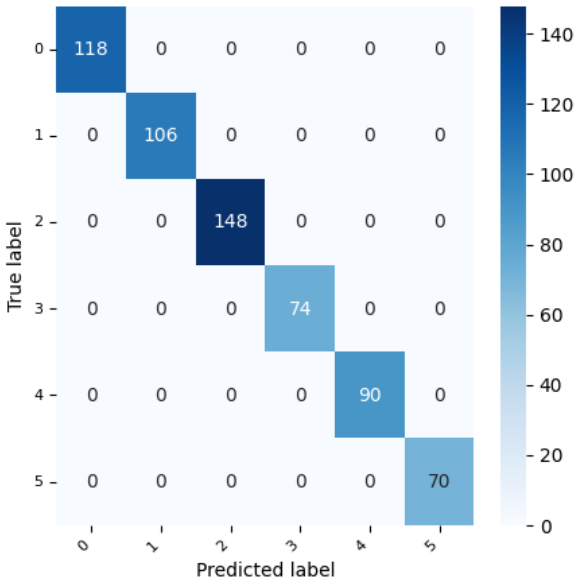

Figure 6.10: Confusion matrix of known attacks in INCEPTION

Figure 6.10 shows the correct classification of all the known attacks. From this, it is evident that it will correctly classifies all the known attacks that occurs in a network.

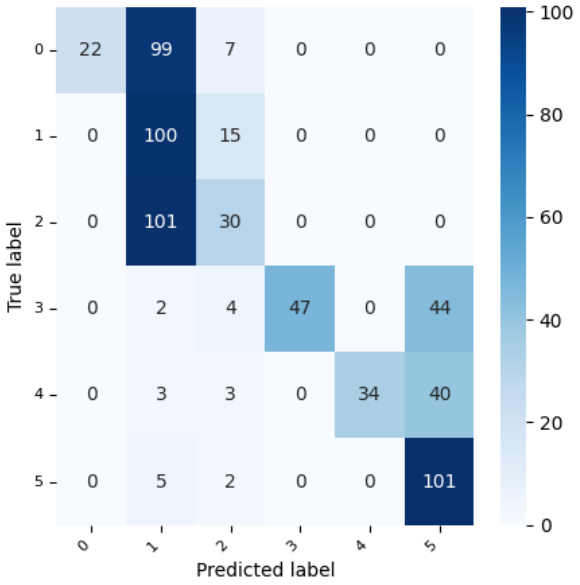

Figure 6.11: Confusion matrix of unknown attacks in INCEPTION

Figure 6.11 shows the missclassification of unknown attacks. From this, it proves that it will not work for classifying unknown attacks. In case, if any unknown attacks occurs it will

missclassify it. It will not classify as known attack.

6.5.2 Results of VGG16

From the results of accuracy, as shown in Table 6.4, it can be seen that the **VGG16** model achieves an accuracy of 0.99%.

Table 6.4: Performance measures of VGG16 model

Performance measures	Train Data	Test Data
Accuracy	0.99	0.70
Precision	0.99	0.79
Recall	0.99	0.69
F1-Score	0.99	0.68

The accuracy graphs of the VGG16 models are shown in Figure 6.12 respectively. The blue line indicates validation accuracy, and the orange line indicates training accuracy.

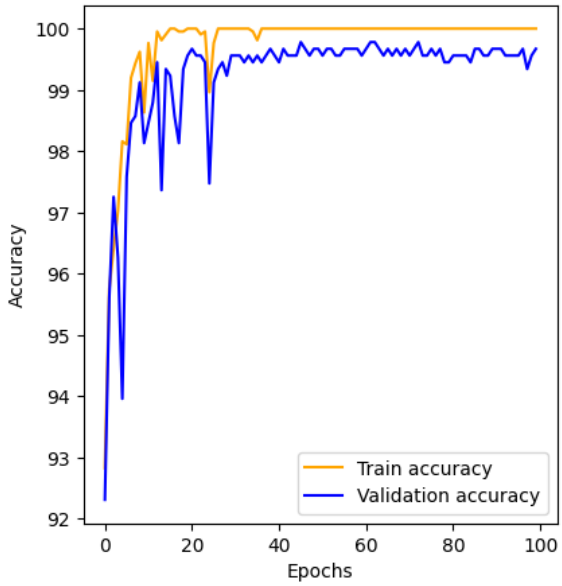

Figure 6.12: Accuracy graph plot of VGG16

The loss graphs of the VGG16 models are shown in Figure 6.13 respectively. The blue line indicates train loss, and the red line indicates validation loss.

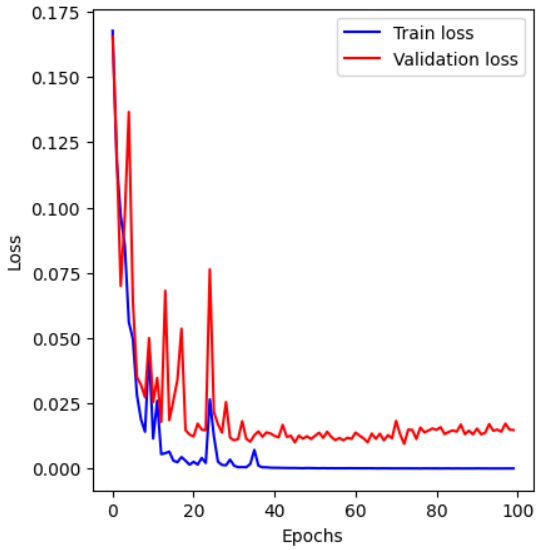

Figure 6.13: Loss graph plot of VGG16

A confusion matrix is a graphical representation and summary of the results of a classification process. A confusion matrix is used to improve the classification algorithm's performance. In the confusion matrix, values in the diagonal represent the perfect classifier.

The confusion matrix of both known attack and unknown attack is given in Figure 6.14 and Figure 6.15 respectively.

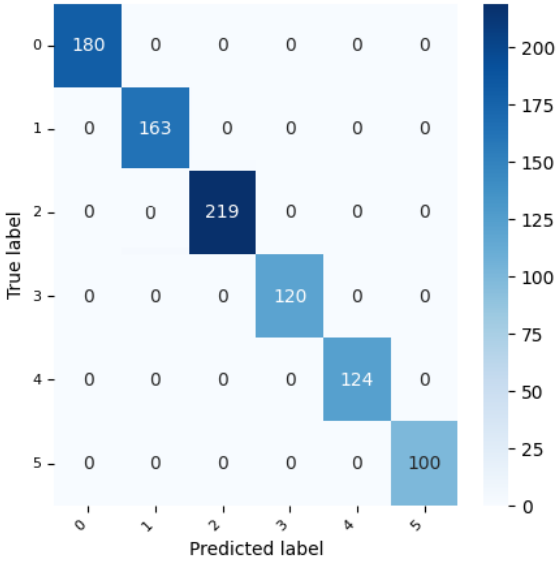

Figure 6.14: Confusion matrix of known attacks in VGG16

Figure 6.14 shows the correct classification of all the known attacks. From this, it is evident that it will correctly classifies all the known attacks that occurs in a network.

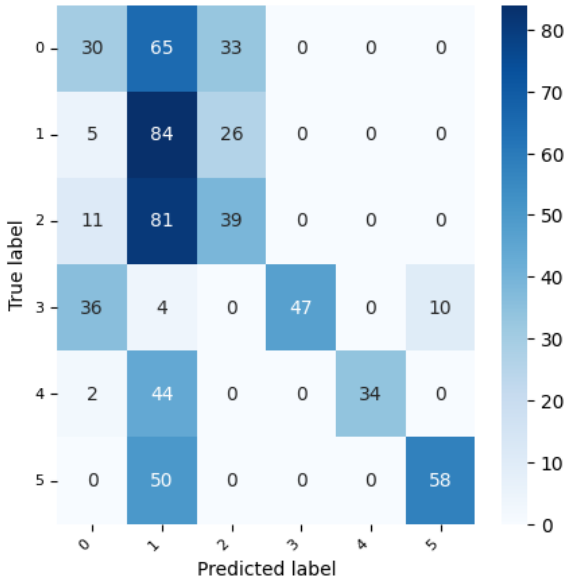

Figure 6.15: Confusion matrix of unknown attacks in VGG16

Figure 6.15 shows the missclassification of unknown attacks. From this, it proves that it will not work for classifying unknown attacks. In case, if any unknown attacks occurs it will missclassify it. It will not classify as known attack.

6.5.3 Results of Resnet50

From the results of accuracy, as shown in Table 6.5, it can be seen that the **RESNET50** model achieves an accuracy of 0.99%.

Table 6.5: Performance measures of RESNET50 model

Performance measures	Train Data	Test Data
Accuracy	0.98	0.63
Precision	0.99	0.75
Recall	0.99	0.63
F1-Score	0.98	0.68

The accuracy graphs of the RESNET50 models are shown in Figure 6.16 respectively. The blue line indicates validation accuracy, and the orange line indicates training accuracy.

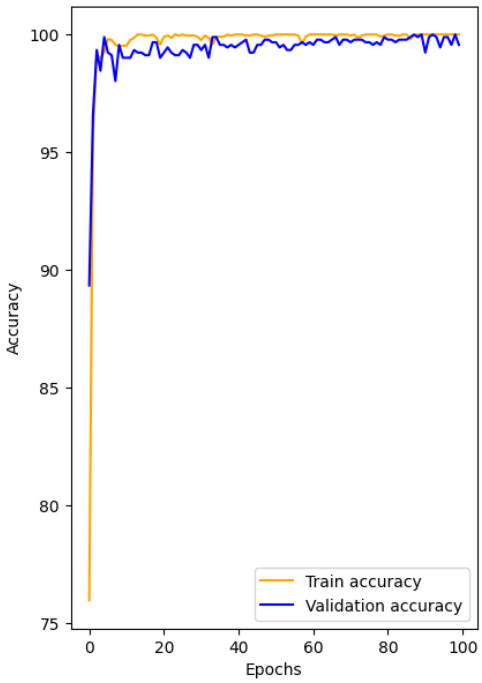

Figure 6.16: Accuracy graph plot of RESNET50

The loss graphs of the RESNET50 models are shown in Figure 6.17 respectively. The blue line indicates train loss, and the red line indicates validation loss.

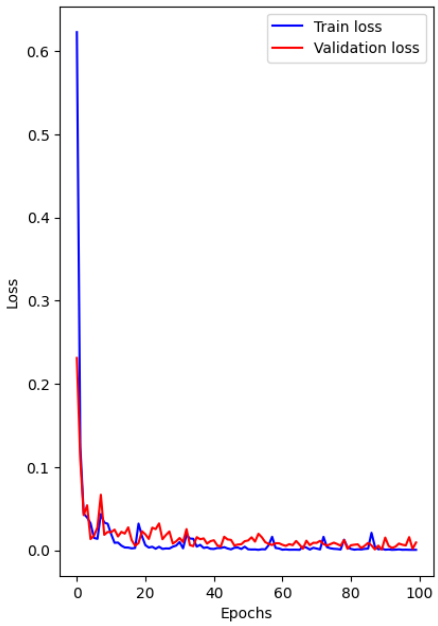

Figure 6.17: Loss graph plot of RESNET50

The confusion matrix of both known attack and unknown attack is given in Figure 6.18 and Figure 6.19 respectively.

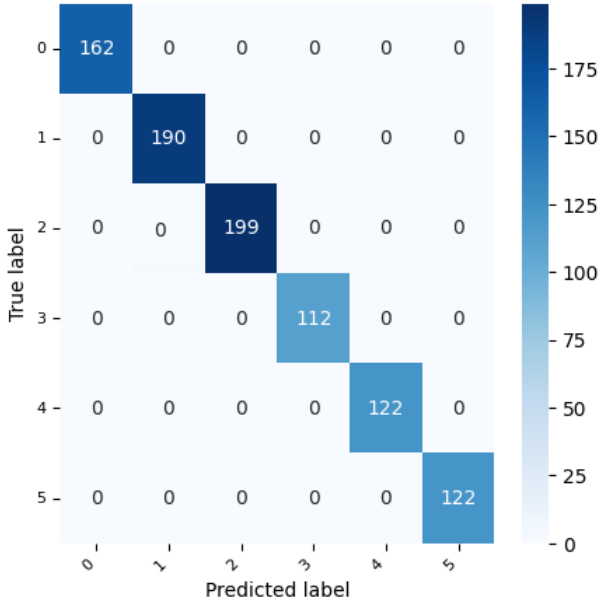

Figure 6.18: Confusion matrix of known attacks in RESNET50

Figure 6.18 shows the correct classification of all the known attacks. From this, it is evident that it will correctly classifies all the known attacks that occurs in a network.

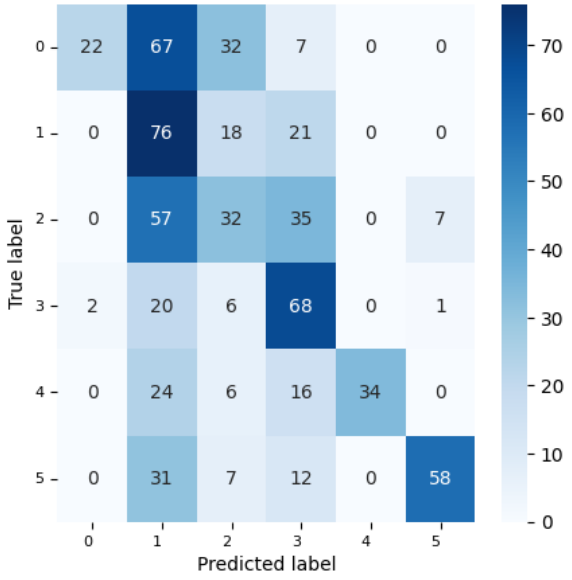

Figure 6.19: Confusion matrix of unknown attacks in RESNET50

Figure 6.19 shows the missclassification of unknown attacks. From this, it proves that it will not work for classifying unknown attacks. In case, if any unknown attacks occurs it will missclassify it. It will not classify as known attack.

6.5.4 Results of Efficientnet

From the results of accuracy, as shown in Table 6.6, it can be seen that the **EFFICIENTNET** model achieves an accuracy of 0.99%.

Table 6.6: Performance measures of EFFICIENTNET model

Performance measures	Train Data	Test Data
Accuracy	0.987	0.60
Precision	0.978	0.75
Recall	0.997	0.60
F1-Score	0.988	0.68

The accuracy graphs of the EFFICIENTNET models are shown in Figure 6.20 respectively. The blue line indicates validation accuracy, and the orange line indicates training accuracy.

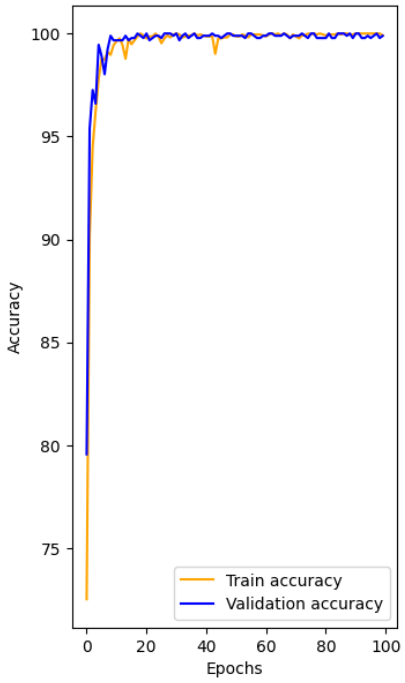

Figure 6.20: Accuracy graph plot of EFFICIENTNET

The loss graphs of the EFFICIENTNET models are shown in Figure 6.21 respectively. The green line indicates train loss, and the red line indicates validation loss.

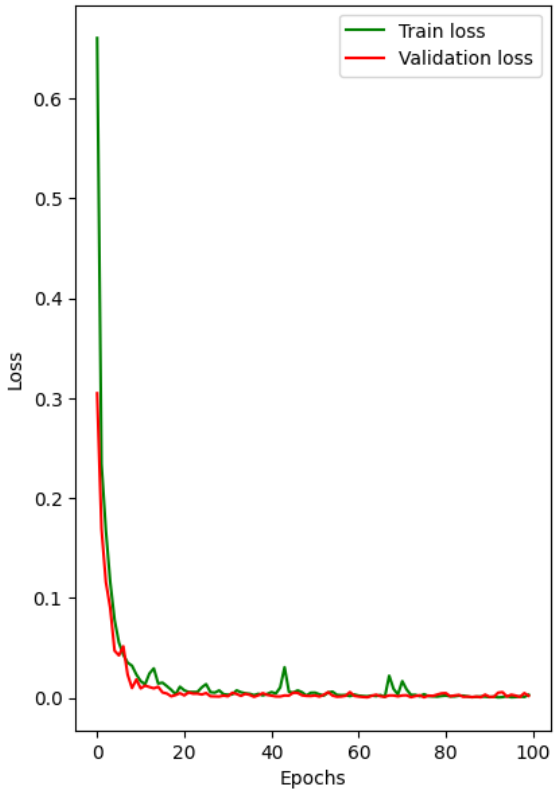

Figure 6.21: Loss graph plot of EFFICIENTNET

The confusion matrix of both known attack and unknown attack is given in Figure 6.22 and Figure 6.23 respectively.

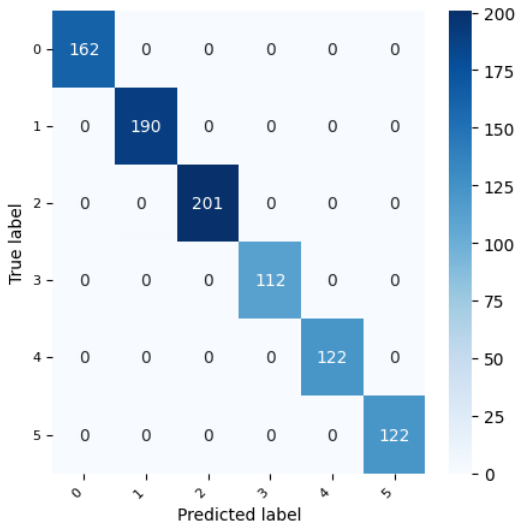

Figure 6.22: Confusion matrix of known attacks in EFFICIENTNET

Figure 6.22 shows the correct classification of all the known attacks. From this, it is evident that it will correctly classifies all the known attacks that occurs in a network.

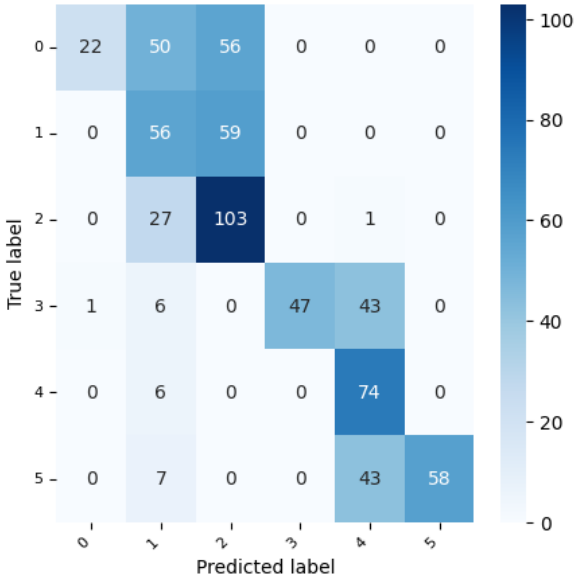

Figure 6.23: Confusion matrix of unknown attacks in EFFICIENTNET

Figure 6.23 shows the missclassification of unknown attacks. From this, it proves that it will not work for classifying unknown attacks. In case, if any unknown attacks occurs it will missclassify it. It will not classify as known attack.

6.5.5 Results of CNN-BiLSTM

From the results of accuracy, as shown in Table 6.7, it can be seen that the **CNN-BiLSTM** model achieves an accuracy of 0.99%.

Table 6.7: Performance measures of CNN-BiLSTM model

Performance measures	Train Data	Test Data
Accuracy	0.958	0.761
Precision	0.940	0.742
Recall	0.958	0.757
F1-Score	0.962	0.743

The accuracy graphs of the CNN-BiLSTM models are shown in Figure 6.24 respectively. The blue line indicates validation accuracy, and the orange line indicates training accuracy.

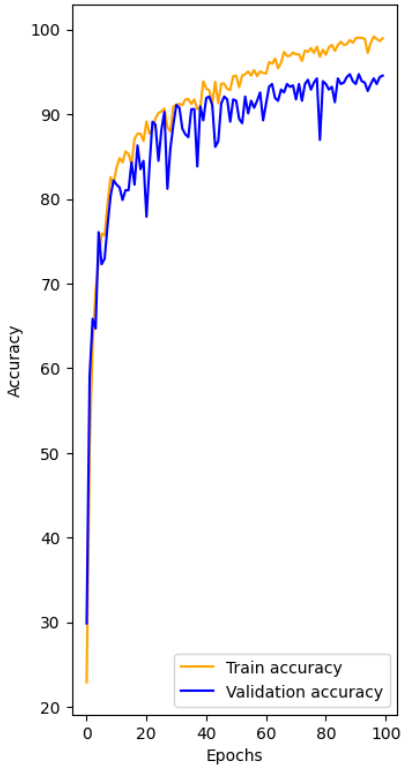

Figure 6.24: Accuracy graph plot of CNN-BiLSTM

The loss graphs of the CNN-BiLSTM models are shown in Figure 6.25 respectively. The blue line indicates train loss, and the red line indicates validation loss.

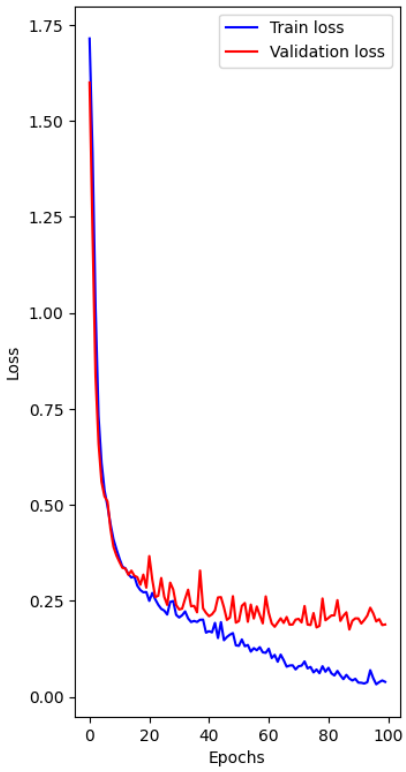

Figure 6.25: Loss graph plot of CNN-BiLSTM

The confusion matrix of both known attack and unknown attack is given in Figure 6.10 and Figure 6.11 respectively.

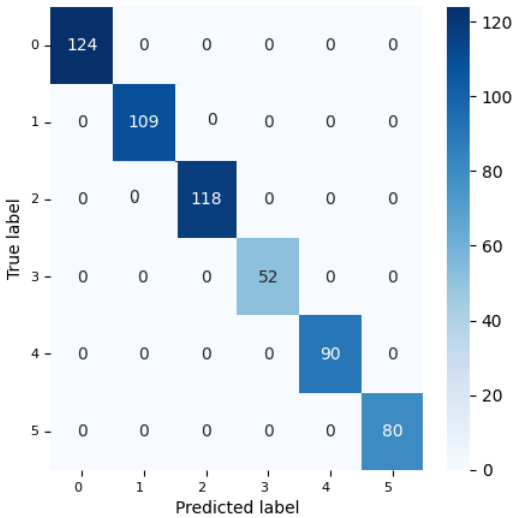

Figure 6.26: Confusion matrix of known attacks in CNN-BiLSTM

Figure 6.26 shows the correct classification of all the known attacks. From this, it is evident that it will correctly classifies all the known attacks that occurs in a network.

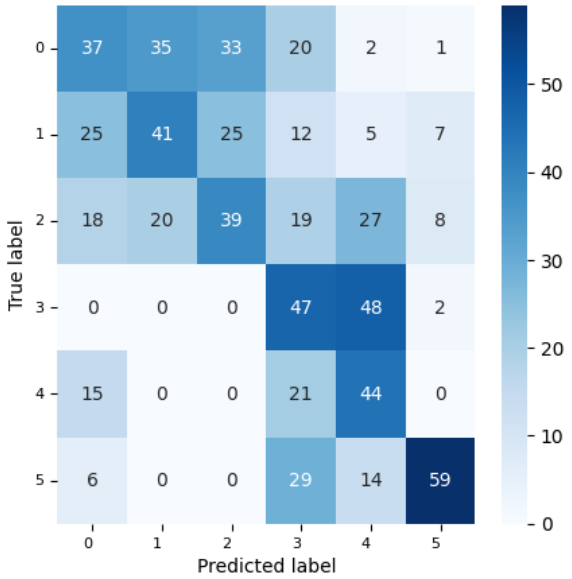

Figure 6.27: Confusion matrix of unknown attacks in CNN-BiLSTM

Figure 6.27 shows the miss-classification of unknown attacks. From this, it proves that it will not work for classifying unknown attacks. In case, if any unknown attacks occurs it will missclassify it. It will not classify as known attack.

From the above all results which is shown in Figure 6.10, 6.14, 6.18, 6.22, 6.26 it states that the model shows correct classification of known attacks. It correctly classifies all the types of known attacks that have occured in any networks.

Chapter 7

CONCLUSION

Deep learning techniques are currently popular in the field of attack detection. The characteristics of data is evident in CAN packet traffic, allowing anomalies to detect CAN network attacks. Future cyberattacks are difficult to anticipate due to the nascent state of automotive cybersecurity. To prevent cyberattacks from penetrating into connected vehicles, this work presents a deep learning framework that employs GAN to identify various of attacks in systems.

From the experimental results it is evident that the proposed GAN model can predict both known and unknown attack when compared with other supervised learning techniques. The other compared models have obtained high accuracy with training data and proved to be succesfull for detecting known attacks. But the compared algorithms are failed in the case of detecting unknown attack. So, inorder to detect the presence of unknown attack the GAN model is proposed.

In addition, quantile transformation method is used for transforming vehicle network traffic data into image data for using GAN as model input. On the ROAD dataset, which represents intra-vehicle data, the proposed IDS is evaluated. The proposed framework has obtained a discriminator accuracy of 0.99,which is robust and effective solution for detecting network intrusions. The experimental results of the proposed IDS framework can identify various categories of intrusions with accuracy of 99.9%. In addition, the results of model testing on a vehicle-level machine demonstrate the viability of the proposed IDS in real-time vehicle networks. Therefore, the GAN framework allows the IDS to generate synthetic data that closely resembles normal network traffic, which enables it to detect novel and previously unseen attack patterns.

The use of GANs for intrusion detection shows promising results, and there is scope for further research and development. One future direction is to explore the application of GANs for detecting more complex attacks, such as multi-stage attacks or stealthy attacks that attempt to evade detection. Another area of research is to investigate the use of GANs in combination with other deep learning techniques, such as deep reinforcement learning, to improve the accuracy and efficiency of intrusion detection. Furthermore, the integration of GANs with real-time intrusion detection systems can lead to the development of more robust and effective security solutions for protecting networks and systems against attacks.

Also, extend to create an online adaptive model capable of online learning and addressing concept drift in time-series vehicle network data. Overall, there is a wide range of possibilities for the use of GANs in intrusion detection, and further research can lead to significant advancements in this field.

References

[1] M. H. Shahriar, Y. Xiao, P. Moriano, W. Lou, and Y. T. Hou, "Canshield: Signal-based intrusion detection for controller area networks," *arXiv preprint arXiv:2205.01306*, 2022.

[2] M. D. Hossain, H. Inoue, H. Ochiai, D. Fall, and Y. Kadobayashi, "Lstm-based intrusion detection system for in-vehicle can bus communications," *IEEE Access*, vol. 8, pp. 185 489–185 502, 2020.

[3] M. Nam, S. Park, and D. S. Kim, "Intrusion detection method using bi-directional gpt for in-vehicle controller area networks," *IEEE Access*, vol. 9, pp. 124 931–124 944, 2021.

[4] M. Hanselmann, T. Strauss, K. Dormann, and H. Ulmer, "Canet: An unsupervised intrusion detection system for high dimensional can bus data," *IEEE Access*, vol. 8, pp. 58 194–58 205, 2020.

[5] L. Nie, Z. Ning, X. Wang, X. Hu, J. Cheng, and Y. Li, "Data-driven intrusion detection for intelligent internet of vehicles: A deep convolutional neural network-based method," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2219–2230, 2020.

[6] F. Laghrissi, S. Douzi, K. Douzi, and B. Hssina, "Ids-attention: an efficient algorithm for intrusion detection systems using attention mechanism," *Journal of Big Data*, vol. 8, pp. 1–21, 2021.

[7] V. Hnamte, H. Nhung-Nguyen, J. Hussain, and Y. Hwa-Kim, "A novel two-stage deep learning model for network intrusion detection: Lstm-ae," *IEEE Access*, 2023.

[8] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Vehicular Communications*, vol. 21, p. 100198, 2020.

[9] H.-C. Lin, P. Wang, K.-M. Chao, W.-H. Lin, and J.-H. Chen, "Using deep learning networks to identify cyber attacks on intrusion detection for in-vehicle networks," *MDPI , Electronics*, vol. 11, no. 14, p. 2180, 2022.

[10] J. Sinha and M. Manollas, "Efficient deep cnn-bilstm model for network intrusion detection," in *Proceedings of the 2020 3rd International Conference on Artificial Intelligence and Pattern Recognition*, 2020, pp. 223–231.

[11] Y. Zhu, T. Brettin, F. Xia, A. Partin, M. Shukla, H. Yoo, Y. A. Evrard, J. H. Doroshow, and R. L. Stevens, "Converting tabular data into images for deep learning with convolutional neural networks," *Scientific reports*, vol. 11, no. 1, p. 11325, 2021.

[12] M. E. Verma, M. D. Iannacone, R. A. Bridges, S. C. Hollifield, P. Moriano, B. Kay, and F. L. Combs, "Addressing the lack of comparability & testing in can intrusion detection research: A comprehensive guide to can ids data & introduction of the road dataset," *arXiv preprint arXiv:2012.14600*, 2020.

[13] X. Xia, X. Pan, N. Li, X. He, L. Ma, X. Zhang, and N. Ding, "Gan-based anomaly detection: a review," *Neurocomputing*, p. 093, 2022.