ABSTRACT

Artificial Intelligence (AI) has significantly impacted the development of Network Intrusion Detection Systems (NIDS). AI algorithms, such as machine learning, have enabled NID systems to analyze vast amounts of data, identify patterns, and detect anomalies in real-time. This helps in detecting and preventing cyber-attacks, which are becoming increasingly sophisticated and challenging to detect. NID systems that use AI algorithms are capable of learning from past experiences and can adapt to new threats, making them more effective in detecting intrusions. Furthermore, AI algorithms also help in reducing false positive alarms, which can be a significant hindrance in the effective functioning of NID systems. In conclusion, the integration of AI into NID systems has significantly improved their performance and effectiveness in detecting network intrusions, making them an essential tool in the fight against cybercrime.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

NID	Network Intrusion Detection System
AI	Artificial Intelligence
IDS	Intrusion Detection System
APT	Advanced Persistent Threats
CNN	Convolutional Neural Network
R&D	Research and Development
API	Application Programming Interface
NIDSA	Network Intrusion Detection System Evaluation Dataset A
VoIP	Voice-Over Internet Protocol
VPN	Virtual Private Network
NIDSB	Network Intrusion Detection System Evaluation Benchmark
NIDSC	Network Intrusion Detection System Corpus
GSA	Gravitational Search Algorithm
SDN	Software-Defined Network
NETAD	Network Traffic Anomaly Detection
NOX	Network Operating System for Openflow
SMAD	Statistical Modelling-based Anomaly
AIAD	Adaptive Immune-based Anomaly Detection
SCADA	Supervisory Control and Data Acquisition System
ICS	Internet Connection Sharing
HML	Hybrid multi-level
BDHDLS	Big Data based Hierarchical Deep Learning System
PCA	Principal Component Analysis
LDA	Linear Discriminant Analysis
RFE	Recursive Feature Elimination
ANN	Artificial Neural Networks

ROC	Receiver Operating Characteristics
TPR	True-Positive Rate
FPR	False-Positive Rate
AUC	Area Under the Curve

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

This chapter introduces about network intrusion detection systems (NIDS) and their various types.

1.1 Overview

Cybersecurity risks are a constant worry for people, organizations, and governments alike in the modern world. The risk of a cyber-attacks has significantly grown with the development of technology and the growing dependence on digital systems to store and communicate sensitive information. APT malware, such as trojan horses and backdoors, is specifically designed to bypass firewalls and anti-virus programs on the target network. It is also used to collect private data from infected hosts over a lengthy period, in addition to being utilized for remotely controlling the compromised workstations in an APT assault. APT malware may circumvent firewalls by utilizing protocol on authorized ports and avoid anti-virus software by exploiting polymorphic code.

A successful cyber-attack may have drastic effects. These attacks can result in stealing of sensitive information, such as personal or financial data, intellectual property, or government secrets, which can be then used to for their own personal gain. This data may be utilized unlawfully for espionage, fraud, or identity theft. The work of an NIDs is to scour t the network and keeps track of the packets and data that flow through it. If it finds any suspicious or malicious networks it sends an alert and logs the details within the system.

A NIDS's main job is to find instances of abuse, illegal access, and other network threats. To find possible security issues, it looks at network traffic, including packet data, flow data, and other network metadata. NIDS can be installed in the network at several locations, such as the perimeter, the internal network, and on specific hosts. Depending on where they are deployed, NIDS can offer various degrees of visibility and defense against network intrusions.

1.2 Purpose

The primary aim of this study aims to handle the problematic concerns regarding securing digital information, protection, and communication arising from the internet's staggering expansion and usage. Hackers utilize a variety of attacks in today's world to obtain vital information. This encourages us to carry on this initiative.

1.3 Scope

The scope of AI technology integration into the development of Network Intrusion Detection Systems (NID) includes the following main areas:

- **Massive data analysis:** AI methods, such as machine learning, allow NID systems to analyze massive volumes of data and spot trends, making them more successful at detecting network breaches.
- **Real-time anomaly detection:** NID systems integrated with AI algorithms can identify abnormalities in real-time, providing enterprises with the essential cyber security protection.
- **Learning from previous encounters:** AI algorithms enable NID systems to learn from previous events and adapt to new threats, making them more successful at detecting intrusions.
- **Autonomous response:** AI-powered NID systems can automatically respond to detected threats, such as blocking network traffic or isolating affected systems. This reduces the need for manual intervention and enables faster response times to cyber-attacks.

1.4 Network Intrusion Detection Systems (NIDS)

An essential part of network security is a Network Intrusion Detection System (NIDS), which tracks and examines network traffic to find possible security risks. NIDS is crucial in defending enterprises from cyberattacks as the quantity and sophistication of cyber threats rise.

A NIDS's main job is to find instances of abuse, illegal access, and other network threats. To find possible security issues, it looks at network traffic, including packet data, flow data, and other network metadata. NIDS can be installed in the network at several locations, such as the perimeter, the internal network, and on specific hosts. Depending on where they are deployed, NIDS can offer various degrees of visibility and defense against network intrusions. The ability of NIDS to identify network assaults in real-time and alert security personnel to possible risks is one of its key advantages. Insights on the many kinds of assaults that are directed at the network may also be provided, enabling enterprises to strengthen their security posture. Organizations are able to better analyze their network utilization and find possible areas for improvement with the help of NIDS, which can also give insight into network traffic.

NIDS does, however, have certain restrictions. It can produce a lot of warnings which security personnel may find difficult to handle. Furthermore, it could be prone to false positives and false negatives, which could lessen its efficacy. To reduce false alarms and increase the accuracy of NIDS, it is crucial to correctly install and optimize it. Organizations should deploy additional security measures in addition to a NIDS to safeguard their network. To avoid unwanted access and data breaches, this includes putting in place firewalls, access restrictions, and encryption. It also entails putting security best practices into action, such as updating software often and educating employees about security.

NIDS is a crucial tool for network security since it offers real-time detection and alerting capabilities to support companies in their defense against network threats.

Additionally, it may offer insights on network usage and traffic, helping businesses understand their network and pinpoint areas for improvement. However, NIDS must be properly set and tuned, and businesses must put additional security measures in place to safeguard their network to optimize its efficacy. By using these systems, we can ensure and protect private data, assets through a security system.

1.5 Types of Network Intrusion Detection Systems (NIDS)

1.5.1 Knowledge-based Intrusion Detection

Knowledge-based intrusion detection systems using a database of recognized attack signatures or patterns. To identify a match, they compare the network traffic to a database of signatures. If a match is discovered, security personnel are informed and instructed to investigate the suspected infiltration. With the discovery of new attack signatures, this strategy may be updated and is effective against known attacks. However, the capacity of knowledge-based intrusion detection systems to identify fresh or undiscovered threats is constrained.

The generalization problem also affects knowledge-based techniques. The platform and application, as well as the operating system version, have a significant impact on knowledge of threats. Thus, the resulting intrusion-detection system is highly dependent on a certain environment. Furthermore, it is thought that it is more difficult to identify insider assaults involving privilege misuse because the attacker does not really exploit any vulnerabilities.

Since the contextual analysis provided thanks to the intrusion-detection system is thorough and may have a very low false alarm rate, it makes it simpler for the security experts using it to comprehend the issue and take defensive or reactive action. A downside is the challenge of compiling the crucial data by keeping it current with new vulnerabilities. It takes a lot of work to keep the knowledge base of the intrusion detection system up to date since each vulnerability needs to be carefully examined.

This method enables a very effective implementation, and commercial intrusion detection devices use it. The fundamental disadvantage of this strategy, which is present in all knowledge-based approaches, is that it needs regular updates in order to address the constant stream of new vulnerabilities identified. The obligation to capture all potential aspects of the attacks via signatures exacerbates the current problem.

1.5.2 Behavior-based Intrusion Detection

Machine learning algorithms and statistical models are used in this type of system to learn the network's typical behavior and spot deviations from it. They establish a baseline of typical network activity and notify security personnel when any unusual behavior is discovered. This strategy works well against fresh or unidentified assaults since it is independent on prior attack signature knowledge. However, if the machine learning models are not correctly trained or if the network behavior changes over time, behavior-based intrusion detection systems may be vulnerable to false positives.

As behavior might change over time, it is necessary to often retrain the behavior profile online. The intrusion detection system may become unavailable as a result, or there may be an increase in false alerts. The information system could be assaulted while the intrusion detection system learns new behaviors. Invasive behavior that is not considered abnormal will thus be included in the behavior profile.

Statistics is the most popular technique for creating behavior-based intrusion-detection systems. A variety of variables that are sampled throughout time affect how the user or system behaves. These characteristics include things like how much time a resource is used for and how percent of the session's total CPU, memory, and disk resources were used. The duration of the time sample might range from a few seconds to at least one month.

1.6 Problem Statement

Network Intrusion Detection (NID) is a vital responsibility for computer networks. Traditional rule-based and signature-based intrusion detection techniques fall short when it comes to identifying and categorizing novel and unidentified network assault types. Convolutional neural networks (CNNs) have demonstrated promising results in the detection of many sorts of network intrusions. The use of rule-based systems or signature-based systems, which depend on predetermined rules or signatures to identify known threats, is a classic method for NID. These methods, however, are ineffective against unidentified or unique threats, which call for more advanced ways to identify.

Lack of labeled training data, which is required to train supervised learning models like CNNs, is one of the biggest challenges. There is a need to create new methods for data gathering and labeling since obtaining labeled training data requires a lot of time and money. The requirement to create reliable models that can manage diverse forms of network traffic and threats is another difficulty. CNNs are often competent at seeing patterns in network traffic, but they might not be able to pick up on more sophisticated assaults or attacks that employ sophisticated evasion methods. Additionally, methods for real-time intrusion detection must be developed. In this case, the IDS must be capable of processing large amounts of network traffic in real-time and making prompt decisions regarding whether to block or allow traffic.

1.7 Objective

The goal of employing CNN for Network Intrusion Detection (NID) is to create a reliable system that can instantly identify and categorize network intrusions. The following are the aims of employing CNN in NID in more detail:

- **Data Preparation:** Preprocess the unprocessed network traffic data and prepare it for CNN input.

- **Model development:** To create an architecture for a CNN model that can precisely identify and categorize various kinds of network intrusions.
- **Model training:** To increase the exactness and generalizability of the model, it is necessary to train it using a sizable and varied dataset of network traffic.
- **Model Deployment:** To assess the effectiveness of the trained CNN model utilizing measures including accuracy, precision, recall, and F1-score, among others.

CHAPTER 2

LITERATURE SURVEY

This section discusses the previous studies conducted for the development of intrusion detection systems. The information present in this section tells us about the advancement in the research for cyber security related topics. This section helps to understand how these systems were developed and how it can be improved in the further models.

2.1 Introduction

A literature review is a crucial part of every research project. A literature review in the context of network intrusion detection (NID) entails the methodical examination, analysis, and assessment of already published papers on the subject.

Network intrusion detection is a key element of network security, according to several research on the issue. A literature review on NID includes an analysis of the present research as well as the identification of knowledge gaps and possible research topics. As a foundation for the research study, the literature review finds pertinent topics, research questions, and procedures that have been used in earlier studies.

There are several subtopics that the literature review on NID may be divided into. The examination of intrusion detection methods is the first subtopic. This subtopic examines the many methods that have been employed to identify intrusions, including hybrid, anomaly-based, and signature-based methods. The review of the literature looks at these approaches' advantages and disadvantages as well as how well-suited they are to various network settings.

The assessment of intrusion detection systems is the second subtopic. This subtopic focuses on the various intrusion detection systems that are currently in use and the metrics that are employed to assess them. The review of the literature examines several assessment approaches, including analytical, simulation-based, and empirical evaluation. It also outlines

opportunities for development and addresses the limits of these evaluation approaches.

The use of big data in intrusion detection is the third subtopic. This subtopic looks at the problems with intrusion detection in the big data age and the many methods that have been employed to solve them. The literature review examines several big data analytics methods, including association, classification, and clustering analysis. It also examines these strategies' advantages and disadvantages as well as their applicability in various network contexts.

2.2 Existing System

2.2.1. NIDs and its Related Studies

In this section, the broad concept of NIDS is introduced, and associated research on freely accessible datasets for NIDS R&D is described. IDS, or intrusion detection system is security tool used to keep an eye on system and network activity to identify malicious or unauthorized activity. IDS come in a variety of forms and can be either software or hardware based. There are three types of intrusion detection systems: signature-based, anomaly-based, and hybrid. Signature-based intrusion detection systems (IDS): Also known as knowledge-based intrusion detection systems (IDS), this form of IDS analyses network traffic or system actions to a database of known attack signatures, patterns, or profiles. The IDS generate an alert if the traffic or behavior matches a signature in the database. As new attacks are identified, the database is regularly updated with new signatures. Nevertheless, Anomaly-based IDS are more accurate at finding unfamiliar types of attacks on the network which are not on the database. Each variation that surpasses a certain threshold is considered a possible attack. Hybrid intrusion detection systems use the advantages of both technologies to improve overall security coverage.

It is essential to adopt cutting-edge network intrusion detection systems (NIDs) driven by artificial intelligence (AI) to safeguard corporate networks against cyberattacks, which have recently gotten more varied and complex. To train AI-powered NIDs, high-quality

labelled training datasets are necessary, but creating new training datasets is time-consuming and difficult to find internationally. The Existing system extracts information from the network such as IP Address, Port Number and Time Stamp.

The Data used in the NID is mostly stored as packet data, which is the most common type. It contains network packet headers and payloads, which might contain information about the traffic's source and destination, the protocol employed, and the contents of the packets. NIDS can also analyze metadata, which is data about data. For example, metadata can include information about the size, timestamp, and other characteristics of network packets or other data sources. Flow data is a summary of network traffic that includes information about the source and destination of the traffic. This is used to keep track of where the data is coming from and where it is going to. This is a very useful tool to keep track of normal networks and detect suspicious activity in the network.

The fact that there is a many-to-many link between alerts and the data from API responses is a problem. Therefore, we might think of this connection as one in which numerous alarms from an API answer are tied to one API response. In other words, a communication ow has several alarms connected with it. This indicates that, to implement the suggested technique for the alert of NIDSA. In this case, a single caution can subsequently affect a lot of training labels. However, the system already includes this feature. Some publicly available datasets were created before the worrying number of assaults that have been employed in recent years and do not exist on these databases. Furthermore, the NID's performance will suffer if the dataset is too small.

ADVANTAGES:

1. Ability to Deliver High Quality Results
2. It’s not difficult to see what is Impacted
3. High Effective with Complex Problems

DISADVANTAGES:

1. Poor Application Performance
2. Cannot be implemented real time
3. Computation burden may limit its further application for real scenarios.

2.2.2. Evaluation of Communication Packets and Trigger Packets in NIDS

Employing cutting-edge network intrusion detection systems(NIDS) driven by artificial intelligence (AI) is essential for safeguarding corporate networks against recent increases in the variety and sophistication of assaults AI-powered NIDS must be trained using high-quality labelled training datasets however these datasets are hard to come by worldwide and creating fresh training datasets is thought to be laborious the authors of the current study look at the viability of a method that combines the advantages of current security appliances to provide labeled training datasets which might be utilized to produce new AI-powered cybersecurity solutions. The authors of the current work begin by detecting communication flows that the installed NIDs view as suspicious, investigating their causes, and uniformly labeling them with the appropriate labels. The packet data in the identified communication flows is then produced by the studies' authors as labelled data along with the necessary alert-type labels. The authors demonstrate the effectiveness of the labeling scheme by contrasting classification models that were trained using the labeled dataset produced by the authors of the prior study. Additionally, authors of earlier studies provide case studies to examine the effectiveness of a number of regularly used NIDs and to discuss practical solutions to automate the security triage procedure. Labeled datasets for this study were made using open-source NIDs and public datasets to ensure that the results could be repeated. The public is given access to the datasets and software tools for use in research.

The authors used traffic data which T-pot connected with clients in our network and alerts served by NIDSA (Network Intrusion Detection System Evaluation Dataset A) NIDSB (Network Intrusion Detection System Benchmark) NIDSC (Network Intrusion Detection System Corpus).

ADVANTAGES:

1. Reduce resource consumption while meeting reliability demands.
2. Simple to understand and interpret the model.
3. Minimizes the workload on infrastructures.

DISADVANTAGES:

1. Narrowly specialized knowledge
2. Difficult to be used in large-scale parallel computing.
3. This system is Opportunistic and uncontrollable.

2.2.3. Flow based Intrusion Detection System

Flow-based intrusion detection is now the subject of substantial study. The authors have used a multi-layer perceptron and gravitational search algorithm-based flow-based anomaly detection system. The system has a very high accuracy rate for classifying benign and harmful traffic. The authors suggested a NIDS and obtained a low false alarm rate utilizing a one-class support vector machine for their study. To previous research, the system is trained on a hostile network dataset. It is possible to use intrusion detection algorithms from conventional networks in SDN. Numerous anomaly detection algorithms have also been implemented in the SDN environment to secure the OpenFlow network.

The author's demonstrated how a programmable home network router may offer the best platform for identifying security issues in network by utilizing the programmability of SDN. Rate limitation, maximum entropy detector, and NETAD are four well-known traffic anomaly detection methods that are implemented in an SDN environment utilizing OpenFlow compliant switches and a NOX controller. Experiments show that these algorithms are significantly more effective than the ISP (Internet Service Provider) at identifying malicious activities in the SOHO network, and the anomaly detector can operate at line rates without adding any new performance overhead for the traffic on the home network.

ADVANTAGES:

1. High Effective with Complex Problems
2. Quick Calculation Time
3. Lowering the Complexity Threshold

DISADVANTAGES:

1. Complexity of its Real Time Implementation
2. Big payloads
3. Heavyweight

2.3 Summary of Literature Survey

2.3.1 Immune System Based Intrusion Detection System (IS-IDS)

The author's in this paper came up with a method for harnessing the immune system, which helps our bodies fight off illnesses, for protecting computer networks. The researchers wanted to find a way to watch over the network, keep a record of what happens, and use special tools to find any problems or intrusions in the network. The authors of this paper have developed an intrusion detection system based on the human body's immune system to protect computer networks. It also keeps a watchful eye over network to find any vulnerabilities and can detect nefarious assaults on the system. This system was tested and used in real-time situations.

In this system the authors developed a dual based system. The first layer is referred to as Statistical Modelling based Anomaly Detection (SMAD). This is used to keep a watchful eye over the computer network and the second layer as Adaptive Immune-based Anomaly Detection (AIAD) It looks at certain features of the suspicious network packets, which are like small pieces of information, to find any abnormal activity. It pays attention to things like where the packets are from and what information they contain.

ADVANTAGES:

1. Fast and efficient, but also as accurate as the state-of-the-art algorithms
2. Achieve sub-optimal performance.
3. Improve the operational efficiency.

DISADVANTAGES:

1. Complexity of its Real Time Implementation
2. Cannot be implemented real time
3. Cannot meet current network business demands

2.3.2 A Hybrid-Multilevel Anomaly Prediction Approach for Intrusion Detection in SCADA

This paper talks about safeguarding utility systems and creating an intrusion detection model to solve these problems. This paper discusses the drawbacks wherein there is little to no dataset for these types of attacks they mainly occur in nuclear power plants electricity generators and gas stations. However, many researchers have come up with distinct methods and processes to solve this issue as harmful assaults have increased in frequency over time the authors of this work have created a system that employs an imbalanced dataset with a small number of attacks.

To solve this problem the researchers combined various different techniques. Firstly, the dataset is categorized for finding patterns in the data easier. This is done by using a dimensionality algorithm for improving anomaly detection. Using the above two techniques a database is created for the relevant data to be used in the system. Finally, the authors have combined various detection methods to make a hybrid system that can find new attacks

ADVANTAGES:

1. Proving High Robustness and imperceptibility.
2. Provides the integrity and non-transferability.
3. Relatively simple and computationally inexpensive method.

DISADVANTAGES:

1. Difficult to be used in large-scale parallel computing.
2. High complexity of installing and maintaining
3. Additional configuration is required

2.3.3 Applying big data based deep learning system to intrusion detection

A Large amount of data is generated everyday as the world's internet is getting more sophisticated and more devices are being interconnected. Due to this there is a rise of security risks also increases with interconnectivity. Thus, special systems are designed to detect and protect such data. During the boom of the internet many researchers have designed and deployed various systems to stop security risks in the network. As these models use machine learning, it is important to keep it updated as many attacks are more complex than ever before. Since these take place in a very small amount, the datasets for training the systems are very marginal.

To make this system more efficient and accurate in finding and protecting from malicious networks, the authors have devised a model which uses an approach called Big Data Hierarchical Deep Learning System (BDHDLS). This system is like having many experts working together. Each Expert focuses on finding suspicious or malicious networks. Thus, this method has the ability to detect faster and become more accurate. The Authors using multiple machines when deployed reduces the time taken and makes it easier to find attacks than conventional methods.

ADVANTAGES:

1. Excellent empirical performance
2. Fast and efficient, but also as accurate as the state-of-the-art algorithms
3. Capable of further reducing the required level of human effort

DISADVANTAGES:

1. Big payloads
2. Solutions have been proved ineffective
3. Approach is time-consuming

2.3.4 A Linear Systems Perspective on Intrusion Detection for Routing in Reconfigurable Wireless Networks

Some wireless networks have the potential to change and adapt without actually using any framework. These wireless networks contain nodes through which they transmit data from one place to another. This type of networks usually carries a risk as they are continually changing between, there is a chance of security threat in the network during the change. To protect these types of systems Intrusion detection systems are deployed to protect them from malicious attacks. Here the Authors use a different approach for these types of networks.

The Authors use a linear system approach wherein the network is rerouted on the linear systems theory. The authors use a concept called z-plane. This is a special two-dimensional map like structure independent form the various detection metrics. This helps the authors to detect how the packets behave in the network and help in easily detecting the malicious attacks in the network. These two methods were tested and found to be working well in the real-world environment for detecting using locally available information.

ADVANTAGES:

1. Eliminating the huge workload of traditional methods
2. Simple, fast and less complex.
3. It is a fast and easy procedure to perform

DISADVANTAGES:

1. Unsuitable for large scale scenarios.
2. Computation burden may limit its further application for real scenarios.
3. Big payloads

2.3.5 Decentralized Intrusion Prevention (DIP) Against Co-Ordinated Cyberattacks on Distribution Automation Systems

The Power Grid is an Electricity system that provides electricity to our homes, offices, buildings. As technology is getting advanced and smarter so this the power generation systems. These systems can now be controlled from vast distances using computer networks. As such, there is a high risk of malicious attacks to disrupt the working of such system due to the convenience of the networking. Thus, researchers are developing systems in order to protect from these security breaches. But since most of these security systems are used to protect main control areas and individual systems and not much attention is placed on coordinated attacks at multiple points of the system.

The Authors of the system created a novel approach to protect the power grid. This system is called multi-agent system which is a system-wide cybersecurity system for distributed systems. This helps in detecting breaches at multiple nodes of the grid system. The authors also tested this approach with IEEE 13 Node test feeder with high accuracy of the proposed distributed system which is an important part of the power grid.

ADVANTAGES:

1. Improve the quality and consistency of data
2. Improve the operational efficiency.
3. Streamlined and decoupled services

DISADVANTAGES:

1. Proposed system has high polynomial running times.
2. Heavyweight.
3. High complexity, inaccuracy, and inadequacy.

2.4 Issues in Existing System

- Improve the sustainability of the system.
- Poor Application Performance.
- High complexity, inaccuracy, and inadequacy.
- Uses more computational resources to achieve the performance gains.
- Their distribution is not uniform, which makes classification difficult.

CHAPTER 3

SYSTEM DESIGN

The design of the proposed system is discussed in this section. This architecture design shows all the details and specifics involved in the system and how they coordinate with each other.

3.1 System Architecture

The design and execution of a system that quickly recognizes and reacts to network-based threats is referred to as Network Intrusion Detection (NID) System. When it comes to identifying and reducing network security risks, an NID's system design is essential.

An NID system's architecture is made to watch network traffic and examine it for any shady or criminal activity.

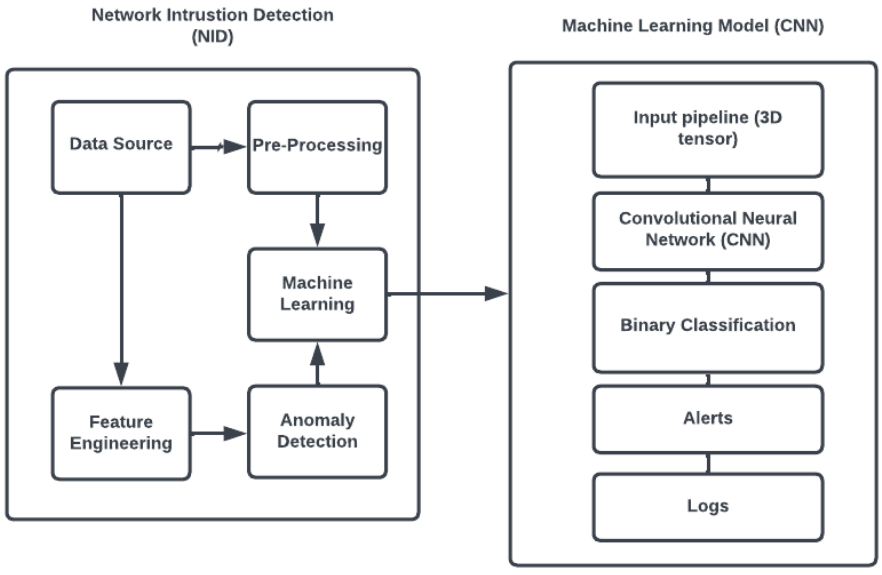


Fig 3.1 Machine Learning Model for Network Intrusion Detection System

Getting the data ready for training is the initial part of the data preparation procedure. Data transformation, normalization, and cleansing are all part of this process. Outliers, noise, and duplicates are removed from the dataset during the data cleaning process. To normalize the data and eliminate any biases, data normalization is utilized. Data transformation entails putting the data in a format that will allow for further processing.

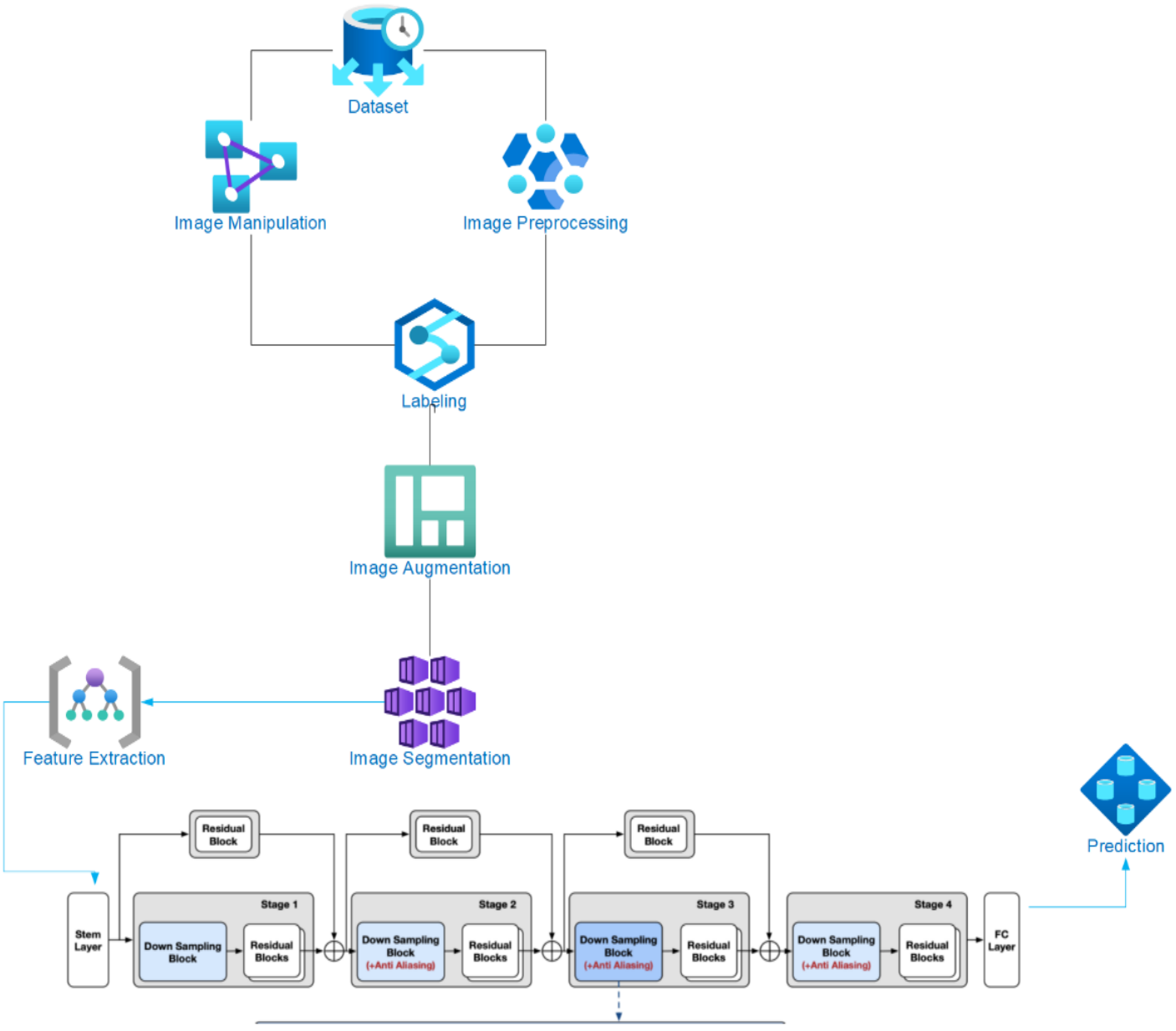


Fig 3.2 System design

Finding the most pertinent characteristics from the preprocessed data is the second component, feature extraction. This is a crucial step since it lowers the dataset's complexity and increases the precision of the model, Recursive Feature Elimination (RFE) are a few techniques for feature extraction.

The CNN model is trained using the preprocessed and feature-extracted data in the third step, which is model training. The model is trained to recognize patterns and characteristics in the data that indicate the presence of malicious behavior. The model's parameters, such as the learning rate, batch size, and number of epochs, are adjusted during the training phase to maximize the model's accuracy.

3.2 Feature Engineering

In the machine learning stage of data preprocessing, feature engineering is a critical step. It entails picking the raw data that is best for model training and translating it into that format. Finding the most crucial data characteristics that are pertinent to the issue at hand and extracting them such that they are helpful to the model are the two main objectives of feature engineering.

The selection of features is one of the most crucial components of feature engineering. This entails deciding which elements to include and remove from the model. It is recommended to only incorporate features that are pertinent to the issue at hand since including too many features might lead to overfitting, which causes the model to become too dependent on the training set and perform badly on fresh data.

Feature transformation is another facet of feature engineering. This entails altering the characteristics so that they are more suited for the model. For instance, one-hot encoding, which generates a binary variable for each category in the data if it contains categorical variables, is a widely used method. As opposed to treating each category as a single variable, this enables the model to consider each as a unique feature.

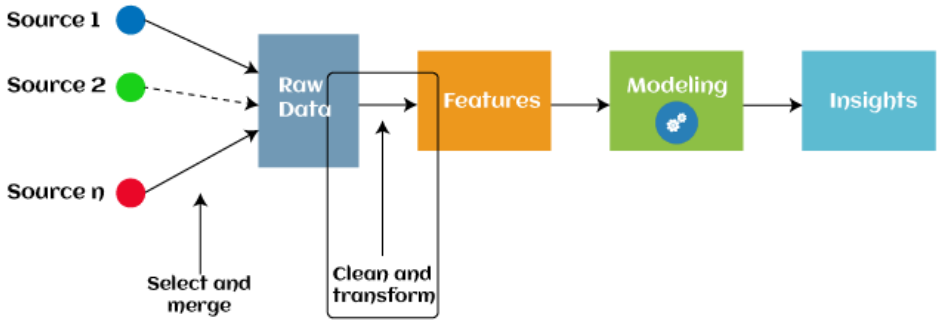


Fig 3.3 Feature Engineering

3.3 Convolutional Neural Network (CNN)

CNNs are a type of deep neural network that is commonly used for image processing tasks. CNNs are intended to identify patterns in visual data by processing the input through a sequence of convolutional and pooling layers that extract data characteristics. In text classification challenges, we may consider each word in a sentence to be a pixel in an image and apply a similar method to learn features at various levels of abstraction.

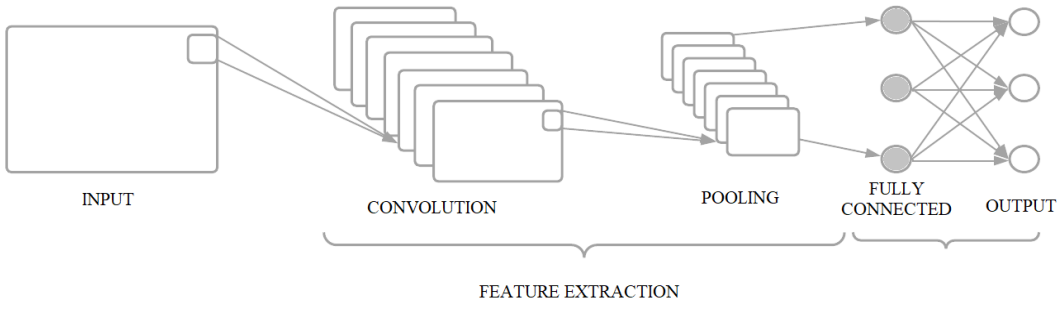
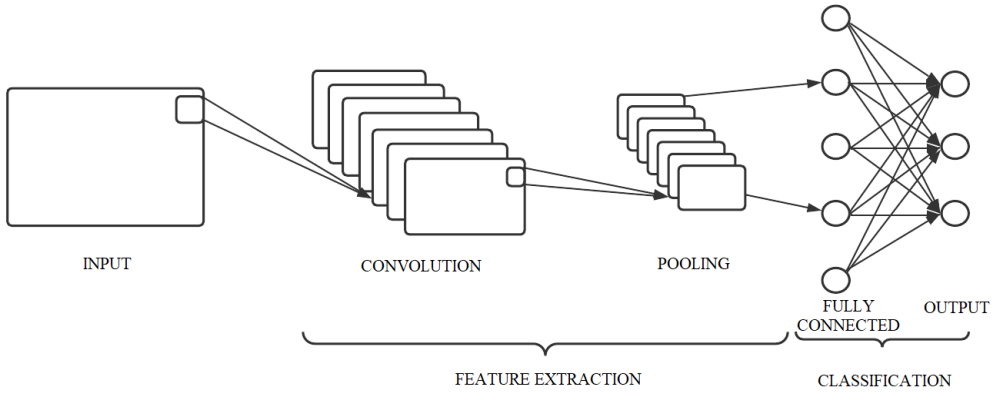


Fig 3.4 Convolutional Neural Network (CNN)

3.3.1 Convolutional Layer

The convolutional layer is the foundation of a CNN. It processes the input picture via a collection of learnable filters and generates a set of feature maps that capture various features of the input data. Increasing the number of filters helps the model to learn more complicated patterns in the input, but it also increases the model's parameter count. Each filter is a tiny, weighted matrix that is convolved with the input picture to get a single output value. The weights in the filter matrix are learnt during training to optimize the model's accuracy.



.

Fig 3.5 Layers of CNN

3.3.2 Pooling Layer

The pooling layer is used to minimize the spatial size of the convolutional layer's feature maps. This is accomplished by dividing the feature maps into tiny sub-regions and producing a summary statistic, such as the maximum or average value, for each sub-region. This decreases the number of parameters in the model and aids in the prevention of overfitting. The feature maps created by the convolutional layers are down sampled by the pooling layer. A typical pooling approach is max pooling, which takes the maximum value from each non-overlapping subregion of the feature map. This decreases the dimensionality of the feature maps and aids in the prevention of overfitting.

3.3.3 Fully Connected Layer

The CNN's last layer is the fully connected layer, which is used to categorize the input data. It takes the preceding layer's output and applies a set of weights and biases to provide a set of class scores. This layer's weights and biases are learnt during training. The fully connected layer conducts classification on the output of the convolutional and pooling layers. This layer is generally comprised of a single dense layer followed by a SoftMax activation function that provides the probability distribution over the classes. Large quantities of data are sent into the network during training, and the weights of the connections between neurons are changed to reduce the difference between the expected and actual output. Backpropagation is a technique used to improve the prediction accuracy. This method is repeated a few times based on new information and improve the accuracy rate.

3.4 Summary

This section has discussed about the overall structure and the elements of the architectural structure in detail with their roles and responsibilities. The data and the blockchain will be working efficiently without the necessity to check for any of the power consumption problems. The proposed structure is more effective and efficient than the double blockchain layer.

CHAPTER 4

SYSTEM MODULES

4.1 Introduction

In the field of network security, intrusion detection and prevention are essential for protecting computer networks from unwanted access and harmful activity. The first line of defense consists of Network Intrusion Detection (NID) systems, which continually monitor network traffic in order to spot and address any threats. The integration of several system components that cooperate to monitor network traffic, find abnormalities, and categorize suspected intrusions is necessary to build an efficient NID system. The NID system's core components, these modules offer the essential capability to guarantee precise and prompt threat detection. Here, a brief explanation of the essential NID system components and how they contribute to a reliable and effective cybersecurity system. Each module has a specific function that adds to the NID system's overall efficacy and performance.

4.2 Data Acquisition

Data acquisition is the procedure of gathering and recording data from numerous sources. Physical sensors and instruments, as well as digital platforms and systems, can be used as these sources. Retrieving accurate and trustworthy data that can be studied and used for insights, optimizations, and improvements across a wide range of fields is the aim of data collection. Data acquisition for NIDS refers to the process of capturing and analyzing network traffic data to detect potential intrusions or security breaches. It entails gathering data on the packets that are moving over the network, analyzing their properties, and spotting any shady or malicious activity.

NIDS may detect data exfiltration, malicious software, unauthorized access attempts, and other security-related activities by monitoring network traffic. Network traffic data capturing is the first stage in this process. This helps in gathering malicious data packets

directly from the network. These sensors which are placed across the network can be both hardware as well as virtual computers. These are used to keep an eye on the network for any malicious activity. Once this has been achieved, the next step is to analyze the data. This contains source, destination of the packets. Important information included such as packet headers, like as source and destination IP addresses, ports, protocols, and timestamps, can be used to spot suspicious activity.

In order to process and analyze the acquired network traffic, many approaches and technologies are used in data gathering for NIDS. Signature-based detection is a popular technique that involves the NIDS comparing the observed network traffic to a database of known attack patterns or signatures. If a match is discovered, an alert is produced warning of a possible intrusion. A different strategy is anomaly-based detection, in which the NIDS maintains a baseline of typical network activity and identifies any departures as potential threats. Data collecting for NIDS includes ongoing monitoring and analysis to guarantee accurate and efficient detection. It might be difficult to detect threats in real time because to the frequent vastness and complexity of network communication. Therefore, to automate the analysis process and more effectively identify possible dangers, new algorithms and machine learning approaches are used. These algorithms can adapt to new and changing attack patterns by learning from previous network data.

Data management and storage must be carefully taken into account while acquiring data for NIDS. Data about network traffic is continually being collected, thus it needs to be kept effectively and securely. To organize and handle the enormous amount of collected data, large-scale storage systems are used, such as databases or data lakes. The ability to acquire and analyze network traffic data makes data acquisition essential to network intrusion detection systems. Data capture for NIDS promotes network security and improves the overall resilience of digital infrastructure in the face of growing cyber threats by utilizing cutting-edge methodologies and technologies.

4.3 Data Description

Network traffic data and data related to intrusions are the two primary categories into which data for NIDS may be roughly divided. The packets that go over the network are included in network traffic data, whereas intrusion-related data contains details on recognized attack signatures, trends, or abnormalities. To comprehend the important categories in NIDS:

4.3.1 Network Traffic Data

The basis for NIDS operations is network traffic data. It consists of the packets, which hold important knowledge about network communication, that go across the network architecture. Insights into communication patterns, employed protocols, source and destination IP addresses, ports, and payload contents are all provided by this data. NIDS can spot probable abnormalities, suspicious conduct, or malicious actions by examining network traffic data. Important elements of network traffic information include:

- **Packet Headers:** Packet headers are essential pieces of data that include source and destination IP addresses, ports, protocol types (such TCP and UDP), and timestamps. NIDS can follow the source and destination of network traffic and spot any unusual behavior or suspicious activities by analyzing packet headers.

- **Packet Payloads:** Application data or command messages are examples of packet payloads, which relate to the actual substance of the packets. NIDS can study the particulars of network communication and spot any malicious or suspicious activity in the data by analyzing packet payloads.

- **Traffic Flows:** The series of packets transferred between two endpoints is referred to as a traffic flow. In order to detect possible threats, NIDS analyze traffic flows to find patterns and abnormalities in the communication between network devices.

4.3.2 Intrusion Data

NIDS uses intrusion-related data as a benchmark for comparing and identifying probable intrusions or security breaches. It contains details on well-known attack signatures, patterns, or abnormal behavioral patterns. NIDS are able to distinguish between legitimate network activity and malicious activity thanks to intrusion-related data. Important elements of intrusion-related information include:

- **Attack signatures:** These are pre-defined patterns or signatures that stand in for particular known attacks. To find possible intrusions, NIDS checks network traffic against a database of attack signatures. New threats and vulnerabilities are regularly added to attack signatures.

- **Anomaly Models:** To discover anomalies, a baseline of typical network activity must be established, and any departures from that baseline must be noted. In order to identify any unusual patterns or behaviors that would point to a possible intrusion, NIDS uses anomaly models that learn from past network data.

- **Threat intelligence:** External data on new dangers, weaknesses, and attack strategies is referred to as threat intelligence. To keep up to date with the most recent security information and improve its detection skills, NIDS utilizes threat intelligence feeds.

4.4 Library Description

4.4.1 TensorFlow

This Machine learning framework developed by Google for use in designing, constructing, and training deep learning models is called TensorFlow. The TensorFlow library may be used to do numerical computations, which does not seem very remarkable in and of itself, however these computations are carried out via data flow graphs. These networks have nodes that represent mathematical processes and edges that represent the data

that is exchanged between these edges, often multidimensional data arrays or tensors. Tensors, which are multidimensional data arrays, are used by neural networks to conduct the operations which give TensorFlow its name. The fact that TensorFlow is so scalable is another crucial factor. For the goal of training, you may create your code and then have it executed on a CPU, GPU, or across a cluster of these devices. Typically, a significant portion of the computation is spent on model training. To address any potential problems, the training procedure is also performed several times. Due to the increased power consumption caused by this procedure, distributed computing is required. TensorFlow makes it simple to process massive quantities of data by running the code in a distributed fashion.

4.4.2 Deep Learning

Machines are now able to learn and make judgments based on data thanks to a subset of machine learning called "deep learning," which is motivated by the structure and operation of the human brain. It is a potent technology that has transformed a variety of industries, including speech recognition, computer vision, natural language processing, and even gaming. A branch of machine learning known as "deep learning" uses artificial neural networks with several layers. Due to the depth of the neural network, which may include hundreds or thousands of layers, it is known as "deep" learning. Deep learning algorithms are created to learn automatically from vast volumes of frequently difficult and unstructured data. It is possible to train deep learning models on a variety of data formats, including pictures, text, and audio. Deep learning algorithms are based on artificial neural networks (ANNs), which are inspired by the biological neurons in the human brain. ANNs are made up of multiple layers of interconnected artificial neurons, each of which processes information before transmitting it to the next layer to generate the final output. To maximize the output of the network, these connections between neurons are weighted, and the weights are modified throughout training.

4.4.3 Pandas

Pandas is a well-known Python library for data science, and with good reason: it provides strong, expressive, and flexible data structures that, among other things, make data manipulation and analysis simple. These structures include the DataFrame.

Wes McKinney created the sophisticated data manipulation program known as Pandas. It is based on the Numpy package and uses the DataFrame as its primary data structure. You can store and modify tabular data in rows of observations and columns of variables using data frames.

Since Pandas is built on top of the NumPy package, it makes use of or replicates a lot of NumPy's structure. Data from pandas is often used to feed machine learning, SciPy's statistical analysis, and Matplotlib's charting capabilities.Pandas provides a range of features that make it a powerful tool for data analysis, including:

• **Data Structures:** Pandas provides two primary data structures: Series and Data Frame. A Series is a one-dimensional array-like object that can hold any data type. A DataFrame is a two-dimensional tabular data structure consisting of rows and columns. Both structures are flexible and efficient for working with structured data.

• **Data Manipulation**: Pandas provides a range of methods for manipulating data, including filtering, grouping, joining, pivoting, and reshaping. These methods enable users to transform and clean data easily.

4.4.5 Keras

Developers can easily design and train deep learning models with Keras, an open-source API built on Python. François Chollet created it with the intention of offering a user-friendly and extendable interface for creating neural networks. Keras provides a streamlined API that enables developers to concentrate on the network's design and architecture rather than the implementation specifics by employing lower-level deep learning libraries like TensorFlow and Theano. The modularity of Keras is one of its key strengths. Layers of neural networks

can be built and stacked by developers to create complex architectures. As a result of this modularity, developers can experiment with different setups and architectures by adding, removing, or changing layers. Additionally, Keras offers a variety of activation functions, loss functions, and optimization algorithms in addition to supporting a number of neural network topologies, such as feedforward neural networks, convolutional neural networks (CNNs), and recurrent neural networks (RNNs). Keras is made to be simple to use and beginner-friendly. Building deep learning models is made simple by the API's clarity, usability, and explicit documentation. Keras offers numerous tutorials and examples to help developers build deep learning models.

Keras is not only easy to use, but also quite effective. It is constructed on top of TensorFlow, which enables effective CPU and GPU processing. Additionally, Keras enables distributed training, enabling programmers to train models across a number of computers or GPUs, which can help shorten training times. The mobility of Keras is another important benefit. Developers may reuse and distribute their models across various platforms and environments thanks to Keras models' simple saving and loading capabilities. Additionally, Keras offers interface with well-known data science libraries like Scikit-learn, NumPy, and Pandas. Keras offers a strong and straightforward API for creating deep learning models. Both novices and specialists in the field of deep learning favor it because of its modularity, effectiveness, and versatility. With its thorough documentation and expanding community, Keras is a vital tool for anybody working with neural networks as it continues to develop and improve.

4.4.6 Matplotlib

A well-liked data visualization toolkit called Matplotlib offers a number of tools and methods for making excellent plots and charts in Python. Developers may make a variety of visualizations with Matplotlib, such as line plots, scatter plots, bar charts, histograms, and more. Additionally, it gives you the freedom to change the colors, labels, axes, and legends of plots to suit your preferences. To produce intelligent data visualizations, Matplotlib may also be used in concert with other data science tools like NumPy and Pandas. Matplotlib is a

potent tool for data visualization in scientific research and business applications due to its adaptability and simplicity.

4.5 Hardware Requirements

- Processor: i7 12th Gen.
- 1 TB HDD.
- Memory: Minimum 4GB RAM; Recommended 8GB RAM.
- GPU: Minimum Nvidia MX250; Recommended NVIDIA Geforce RTX 3060ti.

4.6 Software Requirements

- Operating System: Windows 10
- Anaconda
- Jupyter Notebook
- Language: Python
- Tensorflow
- Pandas
- Keras

CHAPTER 5

IMPLEMENTATION

5.1 Introduction

In this Chapter, the details of the implementation of the project is discussed. Firstly, the Dataset that contains a combination of normal and malicious networks that is used for training the machine learning model for the intrusion detection system. Secondly using machine learning models such as neural networks to train the dataset to the required level in order to detect the malicious networks and provide details on it. This is done be using the feature elimination method to get the appropriate data and to avoid the null and empty values. Lastly, the model is then tested for its accuracy and evaluated to be used in real-time applications for detecting malicious networks.

5.2 Overview of the Platform

The implementation was carried in Jupyter Notebook to take advantage of its simplicity and ease to use platform A web-based interactive tool called Jupyter Notebook enables users to develop and exchange documents with narrative text, live code, equations, and visualizations. It was developed in 2011 by Fernando Pérez as a side project from IPython and was first known as IPython Notebook.

Jupyter Notebook is a popular data science and research tool because it enables people to undertake data analysis, machine learning, and statistical modeling in an interactive environment. It also enables students to document their work, discuss their discoveries, and communicate in real time with others.

One of the primary benefits of Jupyter Notebook is that it allows users to execute code cell by cell. This means users can execute a single line or block of code and immediately see the outcome, making it easier to debug and test their programs. Users may also add

markdown cells to their notebooks, allowing them to document their work using narrative prose, photos, and equations.

Jupyter Notebook also has the potential to connect with other prominent data science tools such as NumPy, Pandas, and Matplotlib. Users may now do complicated data analysis, generate interactive visualizations, and communicate their findings in a more appealing manner.

The implementation process is divided into 3 main categories namely:

- Data Pre-processing
- Recursive Feature Selection
- Neural Network Learning and Evaluation

5.3 Data Pre-processing

Pre-processing data is a critical level in any data analysis or machine learning project. It entails converting raw data into a format that may be conveniently analyzed later. The pre-processing stage is critical since it can alter the accuracy of the analytical results. The following will give an overview of data pre-processing and its significance in data analysis.

Data cleansing is the initial stage in data pre-processing. This entails finding and repairing data flaws and inconsistencies. Human mistakes, sensor faults, and system flaws are all possible causes of these problems. Data cleaning procedures often employed include deleting duplicates, dealing with missing data, and correcting out-of-range data values.

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_srv_count	dst_host_same_srv_rate	dst_host_diff_srv_rate	dst_host_same_src_port_rate	dst_host_srv_diff_host_rate
0	0	tcp	http	SF	181	5450	0	0	0	0	...	9	1.0	0.0	0.11	0.00
1	0	tcp	http	SF	239	486	0	0	0	0	...	19	1.0	0.0	0.05	0.00
2	0	tcp	http	SF	235	1337	0	0	0	0	...	29	1.0	0.0	0.03	0.00
3	0	tcp	http	SF	219	1337	0	0	0	0	...	39	1.0	0.0	0.03	0.00
4	0	tcp	http	SF	217	2032	0	0	0	0	...	49	1.0	0.0	0.02	0.00
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
494016	0	tcp	http	SF	310	1881	0	0	0	0	...	255	1.0	0.0	0.01	0.05
494017	0	tcp	http	SF	282	2286	0	0	0	0	...	255	1.0	0.0	0.17	0.05
494018	0	tcp	http	SF	203	1200	0	0	0	0	...	255	1.0	0.0	0.06	0.05
494019	0	tcp	http	SF	291	1200	0	0	0	0	...	255	1.0	0.0	0.04	0.05
494020	0	tcp	http	SF	219	1234	0	0	0	0	...	255	1.0	0.0	0.17	0.05

Table 5.1 Dataset

To make the data compatible with the analytic tools being utilized, data transformation is necessary. Normalization, scaling, and attribute building are examples of data transformation techniques. Data reduction is the fourth stage in data pre-processing. This entails shrinking the dataset without losing critical information. When dealing with huge datasets that might be computationally expensive to examine, this is required. Data reduction techniques include feature selection, principal component analysis and clustering.

Data discretization is the final stage in data pre-processing. This entails transforming continuous data into discrete data. When working with datasets with a high number of potential values, data discretization comes in handy. Since the dataset contains large quantities of data an elimination process must be used to get the required feature of the dataset for training. The bigger the dataset, the more the computational power is required for training the machine learning model. Hence, it is required to find the foremost features in the data for training.

5.4 Recursive Feature Elimination

Recursive Feature Elimination (RFE) is a machine learning approach for selecting essential features from a dataset. The primary goal of feature selection is to get the relevant and informative features that can increase the accuracy of machine learning models while lowering the computational cost of training these models.

RFE is a feature selection method that picks features recursively by considering smaller and smaller groups of features. In other words, it removes the least significant elements from the dataset while keeping just the most relevant ones. RFE works by training a model on the whole collection of features and then prioritizing the relevance of each feature depending on the performance of the model. RFE may be used in conjunction with a combination of different machine learning models, including classification, regression, and clustering techniques. It is especially beneficial for models that are prone to overfitting or when dealing with huge datasets with many characteristics. The removal of irrelevant characteristics can assist to minimize the model's complexity, lowering the danger of overfitting.

RFE has the benefit of being applicable to both supervised and unsupervised learning tasks. RFE is used in supervised learning to choose the most essential features for a certain target variable. RFE may be used in unsupervised learning to find the most significant features for clustering or dimensionality reduction tasks. RFE does, however, have some limitations. One of the key drawbacks is that it is computationally costly, particularly for huge datasets. Furthermore, the performance of RFE is affected by the scoring function and estimator used. Because different scoring functions and estimators can produce different results, it is critical to experiment with various combinations to obtain the best results.

5.5 Neural Network Training and Evaluation

The use of neural networks as algorithmic tools allows for the rational collection of additional input-output pairs after first understanding the relationship between two sets of data. Theoretically, knowledge-based intrusion detection systems may employ neural networks to recognize assaults and locate them in the audit stream.

Neural networks is like a smart system that is capable of finding patterns in large number of different formats unlike statistics. Neural networks learn the regular patterns of that a specific user follows. In this case a UNIX Root user has specific task to be done thus it is easier to predict the regular schedule of this user. Any deviation form this regular tasks can be detected. This helps in recognizing suspicious or malicious activities and to send alerts for

these outcomes. Since neural network finds patterns in a large dataset it is much more useful compared to traditional statistical analysis.

5.6 Metrics of Interest

For evaluation it is important to gather a variety of metrics of interest in our trials, including accuracy, precision, recall, and false positive rate. This may derive additional key measures from these, such as the F1 score, the area under the ROC curve, and the location in the precision-recall (PR) space.

A low precision indicates that the detector is misclassifying a substantial proportion of regular traffic as a threat, resulting in too many alarms being sent to network managers or, worse, steps being implemented to deal with these false threats that damage traffic. A low recall, on the other hand, indicates that the detector is unable to distinguish attacks, and the network is not adequately secured. The PR curve is especially useful in imbalanced settings when accuracy alone may provide outcomes that are skewed towards the dominant class. Jupyter Notebook also has the potential to connect with other prominent data science tools such as NumPy, Pandas, and Matplotlib. Users may now do complicated data analysis, generate interactive visualizations, and communicate their findings in a more appealing manner.

CHAPTER 6

RESULTS AND DISCUSSIONS

6.1 Introduction

This chapter will be discussing about the performance analysis of the Network Intrusion Detection system (NID) and test the model for accuracy

6.2 Analysis of Result and Output

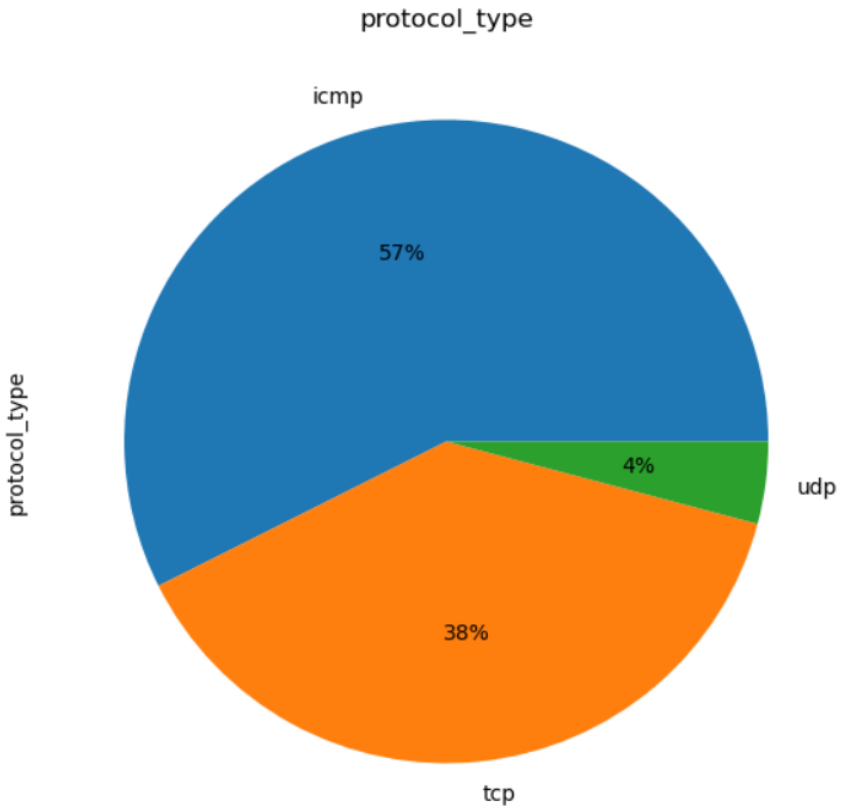


Fig 6.1 Vulnerabilities where attacks have taken place

The above graph shows the vulnerabilities in the network where most of the malicious attacks have happened and tells us where to improve more security in the network. The above graph also shows the most common types of attacks and which place on the network it has taken place.

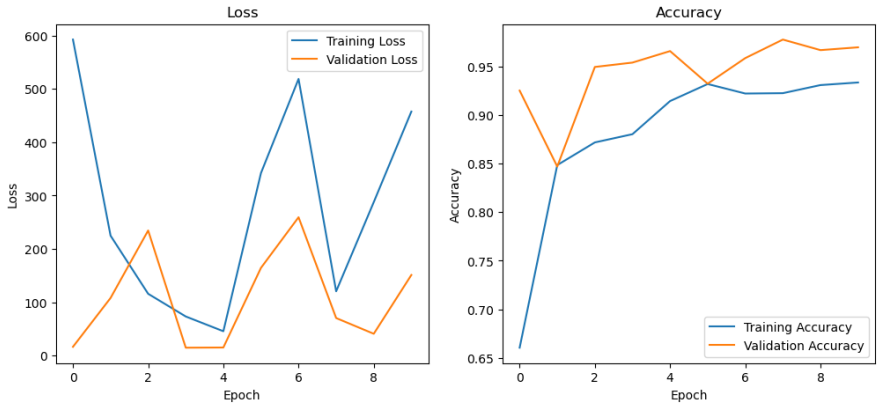


Fig 6.2 Loss and Accuracy of the Training model

The above graph shows the loss values and the accuracy of the training model The loss function, also known as the cost function, is used to assess the model's fit to the training data. The training process's purpose is to reduce the loss function. Loss functions vary depending on the task at hand, such as mean squared error for regression problems and categorical cross-entropy for classification problems.

The accuracy metric, on the other hand, calculates the model's proportion of right predictions based on the training data. It is established by dividing the number of right guesses by the total number of forecasts. The model is supplied with batches of training data during the training phase, and the model's weights are updated depending on the gradients of the loss function with respect to the weights. The loss should reduce, and the accuracy should improve as the training goes. However, this is not always the case, and the loss can sometimes become stuck in a local minimum, resulting in a suboptimal solution.

The accuracy versus epoch value graph is a useful tool for tracking a machine learning model's development during training. Researchers can acquire insight into how well the model is learning and how much additional training may be required by tracking the accuracy over time. Furthermore, the graph can be used to identify model problems, such as overfitting, and adjust to improve its performance.

Moreover, the loss versus epoch graph provides insight into how the model's performance is improving over time. A decreasing loss function indicates that the model is learning and making progress in reducing errors. However, it is important to note that the loss function alone does not provide a complete picture of the model's performance. A low loss function does not necessarily imply a high accuracy, and it is possible to have a low loss function and poor accuracy due to overfitting.

Additionally, the accuracy and loss functions are essential in determining whether a model is overfitting or underfitting. Overfitting occurs when a model performs well on the training data but fails to generalize to new data. In contrast, underfitting occurs when a model is too simple and does not capture the complexity of the data, resulting in poor performance on both training and new data. The accuracy and loss graphs can help researchers identify overfitting or underfitting and adjust the model's hyperparameters or architecture accordingly.

In conclusion, the accuracy and loss graphs are powerful tools for monitoring the training process of machine learning models. They provide insight into how well the model is learning and help researchers identify any issues with the model's performance. By understanding the accuracy and loss graphs, researchers can optimize the model's architecture and hyperparameters to achieve the best possible performance.

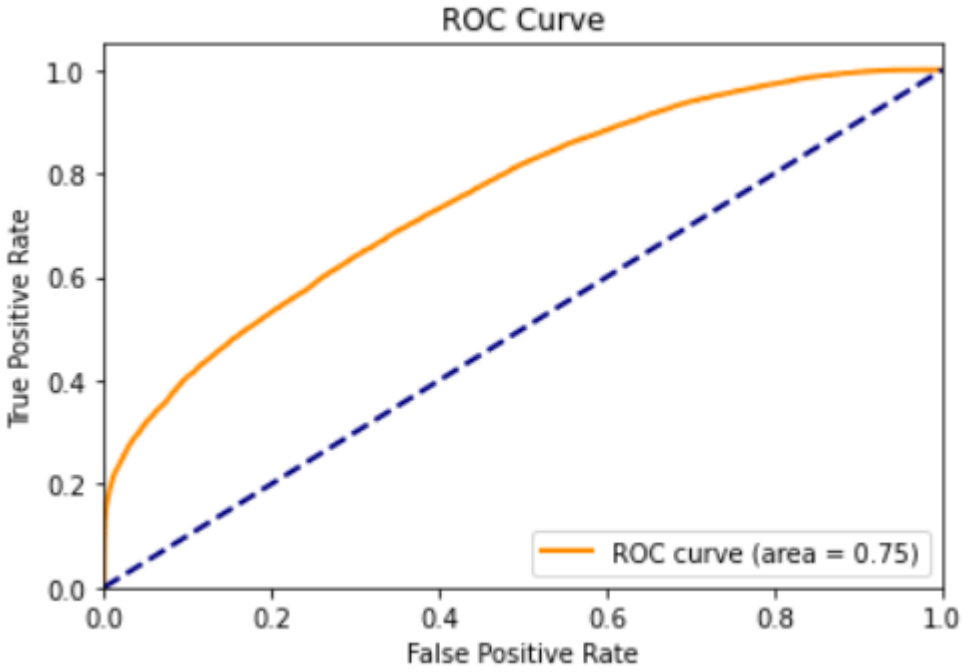


Fig 6.3 ROC Curve of the Model

The ROC curve is a graph that shows how the performance of a binary classifier changes as the discriminating threshold is changed. At various threshold settings, it plots the TPR versus FPR. The AUC is a popular statistic for assessing a classifier's performance.

The ROC curve and AUC are useful in model assessment for numerous reasons It gives a thorough assessment of the model's performance across all categorization thresholds, allowing the user to select the threshold that best meets their requirements. It is unaffected by class imbalance, making it an excellent tool for testing models with skewed datasets. It is a valuable tool for assessing the performance of multiple models, particularly when the models have comparable accuracy but differing misclassification costs.

6.3 Summary

The performance of the proposed system is discussed above which gives a high accuracy compared to the previous models which were used in the intrusion detection systems. The loss - accuracy graph and the ROC Curve demonstrate performance and how it corresponds to testing and analysis.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

The application of artificial intelligence, namely the convolutional neural network (CNN) algorithm, has substantially increased the efficiency and accuracy of network intrusion detection systems. These systems can discover patterns and detect anomalies in network traffic in real time by harnessing the power of machine learning. There are various advantages to applying AI in network intrusion detection systems, including faster threat detection and reaction times, less false positives, and the capacity to identify previously undisclosed or zero-day assaults. Furthermore, AI-based intrusion detection systems can adapt to new and developing threats, making them a significant tool for enterprises and organizations that rely on secure networks to function. Overall, the use of AI in NIDs has changed the way we think about network security, delivering more advanced and dependable protection against cyber-attacks.

7.2 Future Work

The advent of novel attack strategies and the growing complexity of network infrastructures are two factors contributing to the ongoing evolution of the area of network intrusion detection (NID). Future work will focus on enhancing the precision and effectiveness of NID systems by utilizing cutting-edge machine learning methods By allowing NID systems to learn from more data, including unstructured and semi-structured data, these strategies can increase their accuracy. Additionally, Future research will also focus on improving methods for identifying and reducing advanced persistent threats (APTs). APTs, which are sophisticated assaults meant to avoid detection by typical security measures, are becoming more prevalent in the threat environment of today. The interpretability of NID systems, particularly those that include machine learning approaches, must be improved. As a result, security analysts will be better equipped to respond to security risks by understanding the rationale behind the NID systems' decisions.

REFERENCES

[1] Ryosuke Ishibashi, Kohei Miyamoto, Chansu Han , Tao Ban Takeshi Takahashi and Jun'ichi Takeuchi "Generating Labeled Training Datasets Towards Unified Network Intrusion Detection Systems" May 2022

[2] System Integration and Security of Information Systems Andrii Boikoa, Vira Shendryka,P 2016.

[3] Iqbal, Irshad & Calix, Ricardo "Analysis of a Payload-based Network Intrusion Detection System Using Pattern Recognition Processors." October 2016 398-403. 10.1109/CTS.2016.0077.

[4] Ali, Mohammed & Zolkipli, Mohamad. "Intrusion-Detection System Based on Fast Learning Network in Cloud Computing. Advanced Science Letters" October 2018 24. 7360-7363. 10.1166/asl.2018.12942

[5] Mohamed, Walid & Omar, Mohd & Al-Majmar, Nashwan & Fazea, Yousef. "Normal Profile Updating Method for Enhanced Packet Header Anomaly Detection" January 2020 10.1007/978-3-030-33582-3_69.

[6] Hosseinpour, Farhoud & Ramadass, Sureswaran & Meulenberg, Andrew & Vahdani Amoli, Payam & Moghaddasi, Zahra. "Distributed Agent Based Model for Intrusion Detection System Based on Artificial Immune System. International Journal of Digital Content Technology and its Applications" May 2013

[7] ICS-CERT Annual Vulnerability Coordination Report, Department of Homeland Security, Washington DC , USA 2016

[8] Efstathopoulos, Georgios & Radoglou Grammatikis, Panagiotis & Sarigiannidis, Panagiotis & Argyriou, Vasilis & Sarigiannidis, Antonios & Stamatakis, Konstantinos & Angelopoulos, Michail & Athanasopoulos, Solon. "Operational Data Based Intrusion Detection System for Smart Grid" September 2019 10.1109/CAMAD.2019.8858503.

[9] C. Feng, T. Li and D. Chana, "Multi-level Anomaly Detection in Industrial Control Systems via Package Signatures and LSTM Networks," August 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Denver, CO, USA,

2017, pp. 261-272, doi: 10.1109/DSN.2017.34.

[10] Shitharth, & Winston D, Prince. “An enhanced optimization based algorithm for intrusion detection in SCADA network. Computers & Security” May 2017 70. 10.1016/j.cose.2017.04.012.

[11] Mohammadi, Bahram & Sabokrou, Mohammad. “End-to-End Adversarial Learning for Intrusion Detection in Computer Networks.” April 2019

[12] Indirani, G. & Selvakumar, K. “A swarm-based efficient distributed intrusion detection system for mobile ad hoc networks MANET” January 2014 International Journal of Parallel, Emergent and Distributed Systems. 29. 90-103. 10.1080/17445760.2013.773001.

[13] Ganapathy, Satish & Palanichamy, Yogesh & Arputharaj, Kannan. “Intelligent Agent-Based Intrusion Detection System Using Enhanced Multiclass SVM” September 2012 Computational intelligence and neuroscience. 2012. 850259. 10.1155/2012/850259.

[14] Sun, Chih-Che & Hong, Junho & Liu, Chen-Ching. “A coordinated cyber attack detection system (CCADS) for multiple substations” June 2016 1-7. 10.1109/PSCC.2016.7540902.

[15] Mahjabin, Tasnuva & Xiao, Yang & Sun, Guang & Jiang, Wangdong.” A survey of distributed denial-of-service attack, prevention, and mitigation techniques. International Journal of Distributed Sensor Networks” December 2017 13. 155014771774146. 10.1177/1550147717741463.

[16] Amullen, Esther & Keel, Lee. “Consensus-Based Intrusion Detection for the Electric Power Grid Control System” June 2018 1-5. 10.23919/WAC.2018.8430423.

[17] Clarke, Nathan & Furnell, Steven & Tjhai, Gina & Papadaki, Maria.” Investigating the problem of IDS false alarms: An experimental study using Snort” June 2008 International Federation for Information Processing Digital Library; Proceedings of The Ifip Tc 11 23rd International Information Security Conference ;. 278. 10.1007/978-0-387-09699-5_17.

[18] Bela, Genge & Siaterlis, Christos & Karopoulos, Georgios. “ Data fusion-base anomay detection in networked critical infrastructures” June 2013 Dependable Systems and Networks Workshop (DSN-W), 2013 43rd Annual IEEE/IFIP Conference on. 1-8. 10.1109/DSNW.2013.6615505.

[19] Esmalifalak, Mohammad & Nguyen, Huy & Zheng, Rong & Han, Zhu. “ Stealth False Data Injection using Independent Component Analysis in Smart Grid” October 2011 10.1109/SmartGridComm.2011.6102326.

[20] B. Zhu, A. Joseph and S. Sastry, "A Taxonomy of Cyber Attacks on SCADA Systems," 2011 International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing, Dalian, China, 2011, pp. 380-388, doi: 10.1109/iThings/CPSCom.2011.34.

APPENDIX - I

The project on CNN intrusion detection system aims to design and develop a robust intrusion detection system using convolutional neural networks (CNNs). The project involves an in-depth study of the existing intrusion detection systems, including their advantages and limitations. The proposed CNN-based intrusion detection system utilizes a deep learning approach, which allows it to learn from data patterns and make accurate predictions on unseen data. The project involves data collection, pre-processing, model training, and evaluation. The CNN architecture used for the intrusion detection system is carefully designed to improve the performance of the system. The project also includes the implementation of hashing techniques to ensure the integrity of the data. The CNN intrusion detection system's performance is evaluated using various evaluation metrics such as accuracy, precision, recall, and F1 score. The results of the project provide insights into the potential of using deep learning techniques for intrusion detection systems and show the effectiveness of the proposed CNN-based intrusion detection system in detecting network intrusions with high accuracy. Overall, the project contributes to the development of reliable and efficient intrusion detection systems, which are crucial for enhancing the security of computer networks. Thus, the code and concept of the project has been executed and processed to get output successfully.

Sample Coding

```
import pandas as pd
import numpy as np
import tensorflow as tf
import seaborn as sns
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import Label Encoder
from sklearn.model_selection import train_test_split
```

```
from tensorflow.keras import layers, models, callbacks
from sklearn.preprocessing import LabelBinarizer
from keras.layers import Dense, Conv1D, MaxPool1D, Flatten, Dropout
from keras.models import Sequential
from keras.layers import Input
from keras.models import Model
from keras.utils import to_categorical
 from keras.utils.vis_utils import plot_model
from keras.optimizers import Adam
        url1='https://archive.ics.uci.edu/ml/machine-learning-databases/kddcup99-

        mld/kddcup.data.gz'

        url2='https://archive.ics.uci.edu/ml/machine-learning-databases/kddcup99-
        mld/kddcup.data_10_percent.gz'

        file_name1 = 'kddcup.data.gz'

        file_name2 = 'kddcup.data_10_percent.gz'

        data_path = tf.keras.utils.get_file(file_name2, origin=url2, extract=True)

        column_name = ['duration', 'protocol_type', 'service', 'flag', 'src_bytes',

        'dst_bytes', 'land', 'wrong_fragment', 'urgent', 'hot', 'num_failed_logins',

        'logged_in', 'num_compromised', 'root_shell', 'su_attempted', 'num_root',
```

```
 'num_file_creations', 'num_shells', 'num_access_files', 'num_outbound_cmds',

'is_host_login', 'is_guest_login', 'count', 'srv_count', 'serror_rate',

 'srv_serror_rate', 'rerror_rate', 'srv_rerror_rate', 'same_srv_rate',

 'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count', 'dst_host_srv_count',

 dst_host_same_srv_rate', 'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate',

 'dst_host_srv_diff_host_rate', 'dst_host_serror_rate', 'dst_host_srv_serror_rate',

  'dst_host_rerror_rate', 'dst_host_srv_rerror_rate', 'attack_type']
verify_record[msg.sender].time=now;

df = pd.read_csv(data_path, header=None,names=column_name)

df.columns = column_name

Df.columns

Df.shape

print(df.describe())

df.attack_type.unique()

df_corr = df.drop(['num_outbound_cmds', 'is_host_login'], axis=1)
```

```
plt.figure(figsize=(55,55))

sns.heatmap(df_corr.corr(), cmap='coolwarm', annot=True)

plt.show()

print('number of classes:', df['attack_type'].nunique())

print('')

label_counts = df['attack_type'].value_counts()

plt.figure(figsize=(18,6));

sns.barplot(y=label_counts.index, x=label_counts.values, color='Grey');

plt.title('values per class');

display(label_counts)

def pie_plot(df, cols_list, rows, cols):

 fig, axes = plt.subplots(rows, cols)

    for ax, col in zip(axes.ravel(), cols_list):

        df[col].value_counts().plot(ax=ax, kind='pie', figsize=(15, 15),
```

```
fontsize=10, autopct='%1.0f%%')

        ax.set_title(str(col), fontsize = 12)

    plt.show()

pie_plot(df, ['protocol_type', 'attack_type'], 2, 1)

categorical_columns = ['protocol_type', 'service', 'flag', 'attack_type']

encoders = {}

for column in categorical_columns:

    encoder = LabelEncoder()

    df[column] = encoder.fit_transform(df[column])

    encoders[column] = encoder

df.attack_type.value_counts
```

APPENDIX - II

```
In [1]:
import pandas as pd
import numpy as np
import tensorflow as tf
import seaborn as sns
import matplotlib.pyplot as plt
```

```
In [2]:
url1 = 'https://archive.ics.uci.edu/ml/machine-learning-databases/kddcup99-mld/kddcup.data.gz'
url2 = 'https://archive.ics.uci.edu/ml/machine-learning-databases/kddcup99-mld/kddcup.data_10_percent.gz'
file_name1 = 'kddcup.data.gz'
file_name2 = 'kddcup.data_10_percent.gz'
data_path = tf.keras.utils.get_file(file_name2, origin=url2, extract=True)
```

```
In [3]:
column_name = ['duration', 'protocol_type', 'service', 'flag', 'src_bytes',
               'dst_bytes', 'land', 'wrong_fragment', 'urgent', 'hot', 'num_failed_logins',
               'logged_in', 'num_compromised', 'root_shell', 'su_attempted', 'num_root',
               'num_file_creations', 'num_shells', 'num_access_files', 'num_outbound_cmds',
               'is_host_login', 'is_guest_login', 'count', 'srv_count', 'serror_rate',
               'srv_serror_rate', 'rerror_rate', 'srv_rerror_rate', 'same_srv_rate',
               'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count', 'dst_host_srv_count',
               'dst_host_same_srv_rate', 'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate',
               'dst_host_srv_diff_host_rate', 'dst_host_serror_rate', 'dst_host_srv_serror_rate',
               'dst_host_rerror_rate', 'dst_host_srv_rerror_rate', 'attack_type']
```

```
In [4]:
df = pd.read_csv(data_path, header=None,names=column_name)
```


```
In [1]:
import pandas as pd
import numpy as np
import tensorflow as tf
import seaborn as sns
import matplotlib.pyplot as plt
```

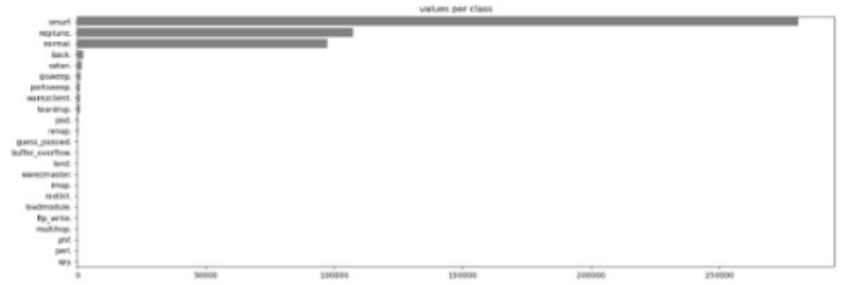
```
In [2]:
url1 = 'https://archive.ics.uci.edu/ml/machine-learning-databases/kddcup99-mld/kddcup.data.gz'
url2 = 'https://archive.ics.uci.edu/ml/machine-learning-databases/kddcup99-mld/kddcup.data_10_percent.gz'
file_name1 = 'kddcup.data.gz'
file_name2 = 'kddcup.data_10_percent.gz'
data_path = tf.keras.utils.get_file(file_name2, origin=url2, extract=True)
```

```
In [3]:
column_name = ['duration', 'protocol_type', 'service', 'flag', 'src_bytes',
               'dst_bytes', 'land', 'wrong_fragment', 'urgent', 'hot', 'num_failed_logins',
               'logged_in', 'num_compromised', 'root_shell', 'su_attempted', 'num_root',
               'num_file_creations', 'num_shells', 'num_access_files', 'num_outbound_cmds',
               'is_host_login', 'is_guest_login', 'count', 'srv_count', 'serror_rate',
               'srv_serror_rate', 'rerror_rate', 'srv_rerror_rate', 'same_srv_rate',
               'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count', 'dst_host_srv_count',
               'dst_host_same_srv_rate', 'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate',
               'dst_host_srv_diff_host_rate', 'dst_host_serror_rate', 'dst_host_srv_serror_rate',
               'dst_host_rerror_rate', 'dst_host_srv_rerror_rate', 'attack_type']
```

```
In [4]:
df = pd.read_csv(data_path, header=None,names=column_name)
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d (Conv1D)              (None, 41, 32)            128

max_pooling1d (MaxPooling1D  (None, 10, 32)            0
)

dropout (Dropout)            (None, 10, 32)            0

conv1d_1 (Conv1D)            (None, 10, 32)            3104

max_pooling1d_1 (MaxPooling  (None, 2, 32)             0
1D)

dropout_1 (Dropout)          (None, 2, 32)             0

flatten (Flatten)            (None, 64)                0

dense (Dense)                (None, 50)                3250

dense_1 (Dense)              (None, 23)                1173

=================================================================
Total params: 7,655
Trainable params: 7,655
Non-trainable params: 0
_________________________________________________________________
```

```
number of classes: 23

smurf.              280790
neptune.            107201
normal.              97278
back.                 2203
satan.                1589
ipsweep.              1247
portsweep.            1040
warezclient.          1020
teardrop.              979
pod.                   264
nmap.                  231
guess_passwd.           53
buffer_overflow.        30
land.                   21
warezmaster.            20
imap.                   12
rootkit.                10
loadmodule.              9
ftp_write.               8
multihop.                7
phf.                     4
perl.                    3
spy.                     2
Name: attack_type, dtype: int64
```



```
history = model.fit(X_train, y_train_onehot, epochs=10, batch_size=5000, validation_split=0.2)
```

```
Epoch 1/10
64/64 [==============================] - 7s 89ms/step - loss: 571.3087 - accuracy: 0.5894 - val_loss: 32.5714 - val_accurac
y: 0.8944
Epoch 2/10
64/64 [==============================] - 6s 92ms/step - loss: 185.8041 - accuracy: 0.8192 - val_loss: 203.4849 - val_accura
cy: 0.9206
Epoch 3/10
64/64 [==============================] - 6s 91ms/step - loss: 77.3911 - accuracy: 0.8888 - val_loss: 37.5794 - val_accurac
y: 0.9300
Epoch 4/10
64/64 [==============================] - 5s 85ms/step - loss: 347.9767 - accuracy: 0.8853 - val_loss: 46.5974 - val_accurac
y: 0.9652
Epoch 5/10
64/64 [==============================] - 5s 84ms/step - loss: 134.4339 - accuracy: 0.8935 - val_loss: 51.9897 - val_accurac
y: 0.9651
Epoch 6/10
64/64 [==============================] - 5s 85ms/step - loss: 63.6361 - accuracy: 0.9131 - val_loss: 30.6374 - val_accurac
y: 0.9620
Epoch 7/10
64/64 [==============================] - 6s 87ms/step - loss: 294.5980 - accuracy: 0.8844 - val_loss: 54.4435 - val_accurac
y: 0.9499
Epoch 8/10
64/64 [==============================] - 6s 97ms/step - loss: 629.3636 - accuracy: 0.8893 - val_loss: 63.7228 - val_accurac
y: 0.9676
Epoch 9/10
64/64 [==============================] - 6s 99ms/step - loss: 746.1953 - accuracy: 0.9017 - val_loss: 53.4017 - val_accurac
y: 0.9773
Epoch 10/10
64/64 [==============================] - 6s 96ms/step - loss: 375.6761 - accuracy: 0.8888 - val_loss: 63.4281 - val_accurac
y: 0.9262
```

```
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score

y_pred = model.predict(X_test)
y_pred_classes = np.argmax(y_pred, axis=1)
accuracy = accuracy_score(y_test, y_pred_classes)
```

```
3088/3088 [==============================] - 5s 2ms/step
```

```
duration = 60
protocol_type_enc = 1
service_enc = 5
flag_enc = 2
src_bytes = 1000
dst_bytes = 500
land = 0
wrong_fragment = 0
urgent = 0
hot = 5
num_failed_logins = 0
logged_in = 1
num_compromised = 0
root_shell = 0
su_attempted = 0
num_root = 0
num_file_creations = 0
num_shells = 0
num_access_files = 0
num_outbound_cmds = 0
is_host_login = 0
is_guest_login = 0
count = 50
srv_count = 20
serror_rate = 0.1
srv_serror_rate = 0.2
rerror_rate = 0.05
srv_rerror_rate = 0.1
same_srv_rate = 0.8
diff_srv_rate = 0.2
srv_diff_host_rate = 0.1
dst_host_count = 150
dst_host_srv_count = 50
dst_host_same_srv_rate = 0.5
dst_host_diff_srv_rate = 0.2
dst_host_same_src_port_rate = 0.1
dst_host_srv_diff_host_rate = 0.01
dst_host_serror_rate = 0.05
dst_host_srv_serror_rate = 0.1
dst_host_rerror_rate = 0.1
dst_host_srv_rerror_rate = 0.05

custom_input2 = np.array([[duration, protocol_type_enc, service_enc, flag_enc, src_bytes, dst_bytes, land, wrong_fragment, u


prediction = model.predict(custom_input2)

if prediction[0][0] == 0:
    print('The predicted attack type is NORMAL.')

else:
    print('An malicious attack type,it can be known or unknown')
```

```
1/1 [==============================] - 0s 23ms/step
The predicted attack type is NORMAL.
```

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Deemed to be University u / s 3 of UGC Act, 1956)

Office of Controller of Examinations

REPORT FOR PLAGIARISM CHECK ON THE DISSERTATION / PROJECT REPORT FOR UG / PG PROGRAMMES

(To be attached in the dissertation / project report)

1	Name of the Candidate **(IN BLOCK LETTERS)**	VISWANATH VS SURENDHARAN TG NAVEEN PK
2	Address of Candidate	Bharathi Salai , Ramapuram , Chennai - 89. **Mobile Number:** 9884066711, 8220224496, 7904482942
3	Registration Number	RA1911003020012 RA1911003020021 RA1911003020054
4	Date of Birth	11/06/2002, 4/07/2002, 30/05/2001
5	Department	Computer Science and Engineering
6	Faculty	Engineering &Technology
7	Title of the Dissertation / Project	A Flexible Hybrid Behavior Dynamic Methodology for Network Intrusion Detection using Convolutional Neural Network
8	Whether the above project / dissertation is done by	Individual or group : Group (Strike whichever is not applicable) a) If the project / dissertation is done in group, then how many students together completed the project : 03 b) Mention the Name & Register number of the candidates: VISWANATH V S [RA1911003020012] SURENDHARAN T G [RA1911003020021] NAVEEN PK [RA1911003020054]
9	Name and address of the Supervisor / Guide	Ms. P. Jayalakshmi, Assistant Professor, Department of ComputerScience and Engineering, SRM Institute of Science and Technology, Ramapuram Campus, Chennai 89 **Mail ID:** jayalakp@srmist.edu.in **Mobile Number:** 9894795761

10	Name and address of the Co-Supervisor / Guide	NA		
11	Software Used	Turnitin		
12	Date of Verification	09/05/2023		
13	**Plagiarism Details: (to attach the final report from the software)**			
Chapter	**Title of the Report**	**Percentage of similarity index (including self citation)**	**Percentage of similarity index (Excluding self citation)**	**Percentage of plagiarismafter excluding Quotes, Bibliography, etc.,**
1	A Flexible Hybrid Behavior Dynamic Methodology for Network Intrusion Detection using Convolutional Neural Network	NA	NA	8%
Appendices		NA	NA	NA

I / We declare that the above information has been verified and found true to the best of my / our knowledge.

Signature of the Candidate	**Name & Signature of the Staff (Who uses the plagiarism check software)**
Name & Signature of the Supervisor / Guide	**Name & Signature of the Co-Supervisor / Co-Guide**

Dr. K. Raja

Name & Signature of the HOD

RE-2022-121544-plag-report

ORIGINALITY REPORT

8%	4%	5%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Ryosuke Ishibashi, Kohei Miyamoto, Chansu Han, Tao Ban, Takeshi Takahashi, Jun'ichi Takeuchi. "Generating Labeled Training Datasets Towards Unified Network Intrusion Detection Systems", IEEE Access, 2022 Publication	1%
2	"Deep learning approach for Network Intrusion Detection in Software Defined Networking", 'Institute of Electrical and Electronics Engineers (IEEE)' Internet Source	1%
3	"Proceedings of Third International Conference on Intelligent Computing, Information and Control Systems", Springer Science and Business Media LLC, 2022 Publication	1%
4	Submitted to Higher Education Commission Pakistan Student Paper	1%
5	Submitted to Intercollege Student Paper	<1%

6	"Neural Information Processing", Springer Science and Business Media LLC, 2017 Publication	<1 %
7	Submitted to University of Arizona Global Campus (UAGC) Student Paper	<1 %
8	"Intrusion detection by machine learning = Behatolás detektálás gépi tanulás által", Corvinus University of Budapest, 2020 Publication	<1 %
9	Submitted to Jawaharlal Nehru University (JNU) Student Paper	<1 %
10	par.nsf.gov Internet Source	<1 %
11	Submitted to Loughborough University Student Paper	<1 %
12	www.essaysauce.com Internet Source	<1 %
13	www.ijert.org Internet Source	<1 %
14	Submitted to Aberystwyth University Student Paper	<1 %
15	hl-128-171-57-22.library.manoa.hawaii.edu Internet Source	<1 %

16	repository.up.ac.za Internet Source	<1 %
17	vtechworks.lib.vt.edu Internet Source	<1 %
18	www.nature.com Internet Source	<1 %
19	Submitted to Middle East College of Information Technology Student Paper	<1 %
20	Submitted to University of Western Ontario Student Paper	<1 %
21	dspace.mit.edu Internet Source	<1 %
22	www.slideshare.net Internet Source	<1 %
23	core.ac.uk Internet Source	<1 %
24	Submitted to Bury College Student Paper	<1 %
25	baadalsg.inflibnet.ac.in Internet Source	<1 %
26	C. Papadopoulos, R. Lindell, J. Mehringer, A. Hussain, R. Govindan. "Cossack: coordinated suppression of simultaneous attacks",	<1 %

PAPER PUBLICATION PROOF

A FLEXIBLE HYBRID BEHAVIOR DYNAMIC METHODOLOGY FOR NETWORK INTRUSION DETECTION USING CNN

SURENDHARAN TG, VISWANATH VS, SURENDHARAN TG, NAVEEN PK, P JAYALAKSHMI
DEPARTMENT OF COMPUTER SCIENCE ENGINEERING, SRM INSTITUTE OF SCIENCE AND TECHNOLOGY, RAMAPURAM, TAMIL NADU

Abstract:

Artificial Intelligence (AI) has significantly impacted the development of Network Intrusion Detection Systems (NIDS). AI algorithms, such as machine learning, have enabled NID systems to analyze vast amounts of data, identify patterns, and detect anomalies in real-time. This helps in detecting and preventing cyber-attacks, which are becoming increasingly sophisticated and challenging to detect. NID systems that use AI algorithms are capable of learning from past experiences and can adapt to new threats, making them more effective in detecting intrusions. Furthermore, AI algorithms also help in reducing false positive alarms, which can be a significant hindrance in the effective functioning of NID systems. In this paper, we apply the CNN machine learning model to analyze and enhance NID performance.

International Research Institute of Multidisciplinary Engineering & Technology

CERTIFICATE OF PRESENTATION

This certificate is presented to

Surendharan TG

"Undergraduate Student, Department of **Computer Science Engineering**, **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**, Tamil Nadu"

The Virtual International Conference on Recent Trends in Emerging Technologies and Engineering - ICRTETE – 2023 Date: 09-April-2023

Organized by

International Research Institute of Multidisciplinary Engineering and Technology, Visakhapatnam

Paper Title: **A FLEXIBLE HYBRID BEHAVIOR DYNAMIC METHODOLOGY FOR NETWORK INTRUSION DETECTION USING CNN**

Authors: **SURENDHARAN TG,VISWANATH VS, SURENDHARAN TG, NAVEEN PK ,P JAYALAKSHMI**

General Chair	Session Chair	keynote speaker
Dr. S. Swarnalatha	Dr. S. Nagaraju	Dr. Abhay Bhatia
SVUCE	Srinivasa Ramanujan Institute of Technology	Roorkee Institute of Technology

ISBN-13: 979-8-390-777-114

(Google Meet: meet.google.com/ess-cuvu-ezd)

International Research Institute of Multidisciplinary Engineering & Technology

CERTIFICATE OF PRESENTATION

This certificate is presented to

Naveen PK

"Undergraduate Student, Department of **Computer Science Engineering**, **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**, Tamil Nadu"

The Virtual International Conference on Recent Trends in Emerging Technologies and Engineering - ICRTETE – 2023 Date: 09-April-2023

Organized by

International Research Institute of Multidisciplinary Engineering and Technology, Visakhapatnam

Paper Title: **A FLEXIBLE HYBRID BEHAVIOR DYNAMIC METHODOLOGY FOR NETWORK INTRUSION DETECTION USING CNN**

Authors: **SURENDHARAN TG,VISWANATH VS, SURENDHARAN TG, NAVEEN PK ,P JAYALAKSHMI**

General Chair	Session Chair	keynote speaker
Dr. S. Swarnalatha	Dr. S. Nagaraju	Dr. Abhay Bhatia
SVUCE	Srinivasa Ramanujan Institute of Technology	Roorkee Institute of Technology

ISBN-13: 979-8-390-777-114

(Google Meet: meet.google.com/ess-cuvu-ezd)

International Research Institute of Multidisciplinary Engineering & Technology

CERTIFICATE OF PRESENTATION

This certificate is presented to

Viswanath VS

"Undergraduate Student, Department of **Computer Science Engineering**, **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**, Tamil Nadu"

The Virtual International Conference on Recent Trends in Emerging Technologies and Engineering - ICRTETE – 2023 Date: 09-April-2023

Organized by

International Research Institute of Multidisciplinary Engineering and Technology, Visakhapatnam

Paper Title: **A FLEXIBLE HYBRID BEHAVIOR DYNAMIC METHODOLOGY FOR NETWORK INTRUSION DETECTION USING CNN**

Authors: **SURENDHARAN TG,VISWANATH VS, SURENDHARAN TG, NAVEEN PK ,P JAYALAKSHMI**

General Chair	Session Chair	keynote speaker
Dr. S. Swarnalatha	Dr. S. Nagaraju	Dr. Abhay Bhatia
SVUCE	Srinivasa Ramanujan Institute of Technology	Roorkee Institute of Technology

ISBN-13: 979-8-390-777-114 (Google Meet: meet.google.com/ess-cuvu-ezd)

A Flexible Hybrid Behavior Dynamic Methodology for Network Intrusion Detection using CNN

P. JAYALAKSHMI, VISWANATH VS, SURENDHARAN TG, NAVEEN PK
[1]SRM INSTITUE OF SCIENCE AND TECHNOLOGY
[2]Department of Computer Science Engineering

ABSTRACT Artificial Intelligence (AI) has significantly impacted the development of Network Intrusion Detection Systems (NIDS). AI algorithms, such as machine learning, have enabled NID systems to analyze vast amounts of data, identify patterns, and detect anomalies in real-time. This helps in detecting and preventing cyber-attacks, which are becoming increasingly sophisticated and challenging to detect. NID systems that use AI algorithms are capable of learning from past experiences and can adapt to new threats, making them more effective in detecting intrusions. Furthermore, AI algorithms also help in reducing false positive alarms, which can be a significant hindrance in the effective functioning of NID systems. In this paper, we apply the CNN machine learning model to analyze and enhance NID performance

INDEX TERMS Artificial Intelligence, Network Intrusion Detection Systems, Machine learning

I. INTRODUCTION

Networks have taken on a crucial role in today's linked world and in contemporary commercial operations. Yet, because of their dependency on networks, they have become a popular target for hackers. Businesses are susceptible to serious harm from cyberattacks, including financial loss, harm to their reputation, and loss of critical data. Organizations must thus put strong security measures in place to shield their networks from online threats.[1]

One of the key security measures is the implementation of an intrusion detection system (IDS). Security technologies called IDSs monitor network traffic for any signs of criminal or suspicious behavior. They have the ability to detect a variety of threats, including malware, denial-of-service attacks, and port scans. IDS can produce real-time notifications that assist administrators in responding to security occurrences and taking the necessary steps to reduce the risk of harm. The alerts may also be used to create statistics and reports on the network's security, assisting managers in determining which areas need more attention.[4]

Unified Network Intrusion Detection UNID is necessary since traditional IDSs are constrained. In order to recognize certain types of attacks, conventional IDS frequently rely on a preset set of rules or signatures. As a result, they can be less able to see new or unexpected results in the longer run.[1]

On the other hand, UNID systems are designed to be more versatile and adaptive, enabling them to recognize a wider range of threats. They use a range of instruments and techniques to locate both recognized and unrecognized attacks. By combining several IDSs, UNID can provide a more full and accurate view of network activity, lowering the possibility of false alarms and missing intrusions.
Instead of lowering the object population below the current levels, these requirements attempted to control its pace of expansion. [1,3,6]

In intrusion detection, machine learning approaches have been widely utilized to identify malicious communications. Sometimes these systems will make mistakes and this can result low accuracy and wrong predictions. 8

II. RELATED WORKS

In this part a brief explanation is given on the related works and the datasets published for NIDs research.

A. NID AND ITS RELATED STUDIES

In this section, the broad concept of NIDS is introduced, and associated research on freely accessible datasets for NIDS R&D is described. IDS, or intrusion detection system is a security tool used to keep an eye on system and network activity in order to identify malicious or unauthorized activity as illustrated in [13, Fig. 1]. IDS come in a variety of forms and can be either software or hardware based. There are three types of intrusion detection systems: signature-based, anomaly-based, and hybrid [1-5]. Signature-based intrusion detection systems (IDS): Also known as knowledge-based intrusion detection systems (IDS), this form of IDS analyses network traffic or system actions to a database of known

attack signatures, patterns, or profiles. The IDS generate an alert if the traffic or behavior matches a signature in the database. As new attacks are identified, the database is regularly updated with new signatures. But Anomaly-based IDS is better at detecting the unknown types of attacks on the network which are not on the database. This is done by creating a baseline of usual activity for a network or system and then searches for deviations from that baseline. Each variation that surpasses a certain threshold is considered a possible attack. Hybrid intrusion detection systems use the advantages of both technologies to improve overall security coverage. [1-2]

It is essential to adopt cutting-edge network intrusion detection systems (NIDs) driven by artificial intelligence (AI) to safeguard corporate networks against cyberattacks, which have recently gotten more varied and complex.
To train AI-powered NIDs, high-quality labelled training datasets are necessary, but creating new training datasets is time-consuming and difficult to find internationally. The Existing system extracts information from the network such as IP Address, Port Number and Time Stamp.[11]

The Data used in the NID is mostly stored as packet data, which is the most common type. It contains network packet headers and payloads, which might contain information about the traffic's source and destination, the protocol employed, and the contents of the packets. NIDS can also analyse metadata, which is data about data. For example, metadata can include information about the size, timestamp, and other characteristics of network packets or other data sources. Flow data is a summary of network traffic that includes information about the source and destination of the traffic, the protocols being used, and the amount of data transferred. NIDS can use flow data to detect anomalies and suspicious patterns of network traffic. [11-12]

The fact that there is a many-to-many link between alerts and the data from API responses is a problem. Therefore, we might think of this connection as one in which numerous alarms from an API answer are tied to one API response. In other words, a communication ow has several alarms connected with it. This indicates that, in order to implement the suggested technique for the alert of NIDSA, the appropriate relationship is first collected via the API. Moreover, one alert is connected to several communication flows. In this instance, one warning may subsequently have an impact on many training labels. Yet, this is a built-in aspect of the system. Certain public datasets predate the alarming amount of attacks used in the recent years, which do not exist on these datasets. Moreover, if the dataset is too small the performance of the NID will also reduce.[1]

B. EVALUATION OF COMMUNICATION FLOWS AND TRIGGER PACKES IN NIDS

Employing cutting-edge network intrusion detection systems(NIDS) driven by artificial intelligence (AI) is essential for safeguarding corporate networks against recent increases in the variety and sophistication of assaults AI-powered NIDS must be trained using high-quality labelled training datasets however these datasets are hard to come by worldwide and creating fresh training datasets is thought to be laborious the authors of the current study look at the viability of a method that combines the advantages of current security appliances to provide labeled training datasets which might be utilized to produce new AI-powered cybersecurity solutions. The authors of the current work begin by detecting communication flows that the installed NIDs view as suspicious, investigating their causes, and uniformly labeling them with the appropriate labels. The packet data in the identified communication flows is then produced by the studies' authors as labelled data along with the necessary alert-type labels.

The authors demonstrate the effectiveness of the labeling scheme by contrasting classification models that were trained using the labeled dataset produced by the authors of the prior study. Additionally, authors of earlier studies provide case studies to examine the effectiveness of a number of regularly used NIDs and to discuss practical solutions to automate the security triage procedure. Labeled datasets for this study were made using open-source NIDs and public datasets to ensure that the results could be repeated. The public is given access to the datasets and software tools for use in research.
The authors used traffic data which T-pot connected with clients in our network and alerts served by NIDSA (Network Intrusion Detection System Evaluation Dataset A) NIDSB (Network Intrusion Detection System Benchmark) NIDSC (Network Intrusion Detection System Corpus)..

C. FLOW BASED INTRUSION DETECTION SYSTEM

Flow-based intrusion detection is now the subject of substantial study. The authors have used a multi-layer perceptron and gravitational search algorithm-based flow-based anomaly detection system. The system has a very high accuracy rate for classifying benign and harmful traffic. The authors suggested a NIDS and obtained a low false alarm rate utilizing a one-class support vector machine for their study. To previous research, the system is trained on a hostile network dataset. It is possible to use intrusion detection algorithms from conventional networks in SDN. Numerous anomaly detection algorithms have also been implemented in the SDN environment to secure the OpenFlow network.

The author's demonstrated how a programmable home network router may offer the best platform for identifying security issues in network by utilizing the programmability of SDN. Rate limitation, utmost entropy detector, and NETAD are four well-known traffic anomaly detection methods that are executed in the SDN environment utilizing OpenFlow compliant switches and a NOX controller. Experiments show that these algorithms are significantly more effective than the ISP at identifying malicious activities in the network, and the anomaly detector can operate at uniform rate without adding any new performance overhead for the traffic on the home network.

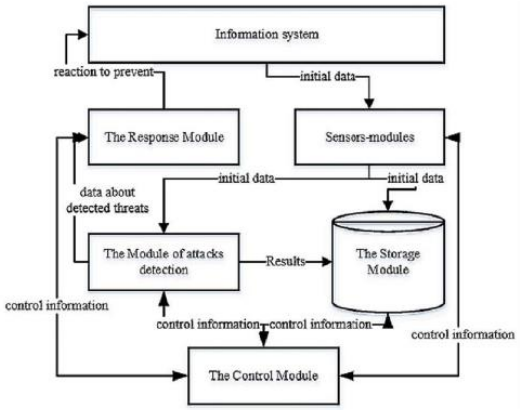


FIGURE 1. Typical Architecture of Network Intrusion Detection System [13]

III. PRELIMNARIES

This section, we will discuss about the preliminaries used in Section IV Proposed Method. We will describe the dataset utilized in this study, followed by the metrics that must be included in the dataset for greater accuracy and reliability.

A. NETWORK TRAFFIC DATA

In Section IV of the Proposed Method we have used KDD Cup Dataset. This dataset was made using DARPA's (Defense Advanced Research Projects Agency) Intrusion Detection and Evaluation systems. Off-line testing was performed on intrusion detection systems utilizing network traffic and audit logs gathered on a simulated network. These data were analyzed in batch mode by the systems, which sought to identify attack sessions in the midst of routine operations.

This dataset contains various attacks namely DoS, DDoS attacks, R2L (Remote-to-User) attack and information gathering attacks such as man-in-the-middle and eavesdropping. These attacks are stores as logs detected by the NIDs with their respective IP Address and port numbers.

B. Measure 1: To determine whether the specified connection sends an alert

It is important to identify from the dataset whether they contain a suspicious or malicious connection within the network. If such a connection is found an alert is produced by the NID and that connection is logged for further analysis. In the case of DoS (Denial-of-Service) and DDoS (Distributed-Denial-of-Service) the NID identifies the source of the attack and blocks traffic from the effected devices. The NIDs then sends an alert that a specified connection is compromised. Traffic filtering and behavioral analysis are used to identify specific attacks by the NIDs. These are the most common approach in detection and prevention of these attacks which are found in the dataset.

IV. PROPOSED METHOD

The predicted decrease in entropy following the split is measured by information gain if the training data is divided based on the values of this feature. Thus, the ability to categories samples more accurately is a property of greater information gain features.

1) DATA PRE-PROCESSING

Pre-processing, the initial step in this stage, is converting data into a format appropriate for deep learning models. Pre-processing decisions must take into account both the type of model being utilized and the data format. When the distances between values indicate some contextual distance, some neural network learning models, such neural networks, often perform at their best. To address these problems, categorical data can be pre-processed using a number of common techniques, such as target encoding or ordinal frequency. The dataset also includes a number of numerical fields in addition to the category ones. Prior to being used in machine learning, numerical data is frequently standardized to guarantee that the inter-feature variation is equal.

Using the above methods, the undesired data from the datasets can be removed as it acts as a critical step in improving the accuracy and reliability of the results obtained from the data. Pre-processing helps to remove noise and irrelevant data, and reduces the impact of outliers, which can improve the accuracy and reliability of analysis results. Data pre-processing can help to organize data in a more structured and consistent way, which makes it easier to analyze and interpret. This can save time and resources in the analysis process. The normalization method is used for a variety of functions, including speeding up the training of classifiers by ensuring that the dataset is consistent and reducing the gap between the data when it exists between huge and small data sets.

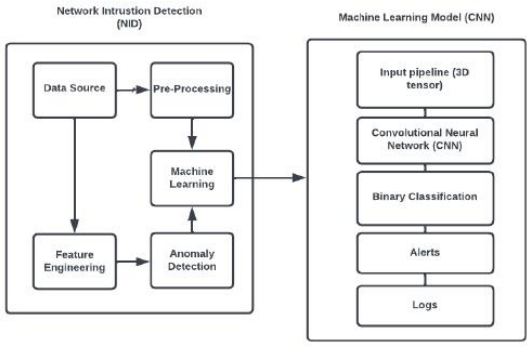


FIGURE 2. CNN model used in Network Intrusion Detection

2) MACHINE LEARNING MODELTRAINING

CNN or Convolutional Neural Network is the main algorithm used for training the dataset to identify if the incoming network traffic is a normal or malicious connection. CNNs are well-suited for analyzing the payload of network packets because they are designed to identify local patterns in the input data. In the context of NID, this involves breaking down the payload of each network packet into smaller segments or "windows" and applying convolutional filters to each segment to identify local patterns. The capacity of CNNs to learn complex feature representations that are challenging to capture using manually constructed rules or signatures is one advantage of their use in NID. This makes it possible for CNNs to identify new or unheard-of attacks that might not be protected by current signatures.

Neural networks are algorithmic approaches that are used to first understand the relationship between two sets of data and then generalize to acquire additional input-output pairs in a sensible manner. In theory, neural networks might be employed in knowledge-based intrusion detection systems to recognize assaults and search them out in the audit stream. Nevertheless, because there is presently no reliable way to interpret what caused the association, the neural network cannot explain the thinking that led to the attack's detection. Neural networks are like a smart system that is capable of finding patterns in large number of different formats unlike statistics. Neural networks learn the regular patterns of that a specific user follows. In this case a UNIX Root user has specific task to be done thus it is easier to predict the regular schedule of this user. Any deviation from these regular tasks can be detected. This helps in recognizing suspicious or malicious activities and to send alerts for these outcomes. Since neural network finds patterns in a large dataset it is much more useful compared to traditional statistical analysis.

V. RESULTS AND EVALUTAION

In this Section we will cover the outcome of the proposed system and evaluate to test the accuracy of the system.

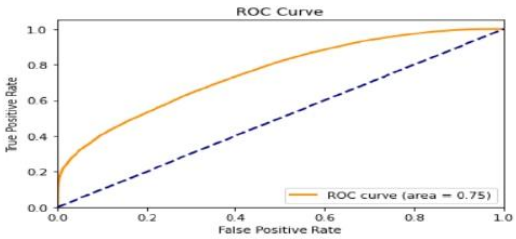


FIGURE 3. The graph shows the ROC Curve

A. TRAINING MODEL ACCURACY

The model is provided with a series of labelled images during the training phase of a CNN, and it learns to extract meaningful features from these images using numerous layers of convolution, pooling, and activation functions. The final layer's output is then sent into a classification layer.

In any machine learning model, there is possibility of false positive in the training model When the model predicts a positive outcome for an input that is actually negative, this is referred to as a false positive. In a binary classification issue, for example, when the aim is to categorize pictures as either containing or not containing a cat, a false positive would arise if the model wrongly predicts that an image without a cat really includes a cat. False positives can arise in a CNN for a variety of reasons, including noisy or unclear data, overfitting, or insufficient training data. These can have serious consequences in particular applications, such as medical diagnosis or fraud detection, where a false positive might result in needless treatments or action Thus, to find the proper accuracy of the training model a separate testing dataset is used. Using this we may acquire a better understanding of the model's behavior and detect possible flaws such as bias, underfitting, or overfitting by assessing it on a different testing dataset. This can assist to increase the model's performance and robustness.

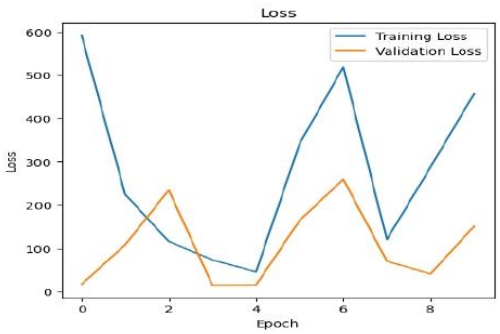


FIGURE 4. The Graph shows the Loss of the Training and validated data

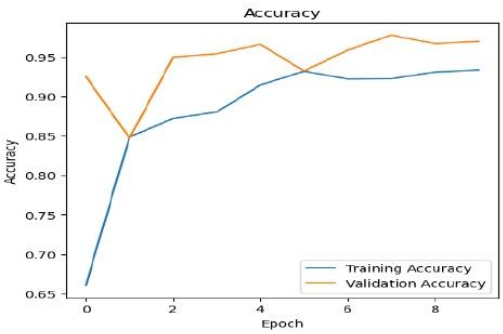


FIGURE 5. The Graph shows the Loss of the Training and validated data

B. EVALUTATION METRIC OF INTEREST

For evaluation it is important to gather a variety of metrics of interest in our trials, including accuracy, precision, recall, and false positive rate. This may derive additional key measures from these, such as the F1 score, the area under the ROC curve, and the location in the precision-recall (PR) space.

A low precision indicates that the detector is misclassifying a substantial proportion of regular traffic as a threat, resulting in too many alarms being sent to network managers or, worse, steps being implemented to deal with these false threats that damage traffic. A low recall, on the other hand, indicates that the detector is unable to distinguish attacks, and the network is not adequately secured. The PR curve is especially useful in imbalanced settings when accuracy alone may provide outcomes that are skewed towards the dominant class. Jupyter Notebook also has the potential to connect with other prominent data science tools such as NumPy, Pandas, and Matplotlib. Users may now do complicated data analysis, generate interactive visualizations, and communicate their findings in a more appealing manner.

With a 0.75 ROC Curve area from Figure 3. The model can determine if a network is malicious or normal for a given piece of data. This demonstrates that the model is functioning and can be used to recognize various networks. These unique types of machine learning approaches can be merged in the future in an IDS to enhance their benefits and overcome their respective shortcomings. Furthermore, developing learning processes with rewards derived from user input or other systems is a viable technique for providing more reliable and resilient intrusion detection systems.

VI. CONCLUSION

The application of artificial intelligence, namely the convolutional neural network (CNN) algorithm, has substantially increased the efficiency and accuracy of network intrusion detection systems. These systems can discover patterns and detect anomalies in network traffic in real time by harnessing the power of machine learning. There are various advantages to applying AI in network intrusion detection systems, including faster threat detection and reaction times, less false positives, and the capacity to identify previously undisclosed or zero-day assaults. Furthermore, AI-based intrusion detection systems can adapt to new and developing threats, making them a significant tool for enterprises and organizations that rely on secure networks to function. Overall, the enlistment of AI in network intrusion detection systems has provided and advantage to network security, delivering more advanced and dependable protection against cyber-attacks.

REFERENCES

[1] Ryosuke Ishibashi, Kohei Miyamoto, Chansu Han , Tao Ban Takeshi Takahashi and Jun'ichi Takeuchi "Generating Labeled Training Datasets Towards Unified Network Intrusion Detection", Systems May 2022

[2] S. Aljawarneh, M. B. Yassein, and M. Aljundi, ``An enhanced J48 classication algorithm for the anomaly intrusion detection systems,"Cluster Comput., vol. 22, no. S5, pp. 1054910565, Sep. 2017

[3] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, ``Toward generating a new intrusion detection dataset and intrusion trafc characterization, '' in Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy, 2018

[4] R.-H. Hwang, M.-C. Peng, V.-L. Nguyen, and Y.-L. Chang, ``An LSTM - based deep learning approach for classifying malicious trafc at thepacket level," Appl. Sci., vol. 9, no. 16, p. 3414, Aug. 2019

[5] H. Hindy, D. Brosset, E. Bayne, A. Seeam, C. Tachtatzis, R. Atkinson,and X. Bellekens, ``A taxonomy of network threats and the

effect of current datasets on intrusion detection systems," IEEE Access, vol. 8, pp. 104650104675, 2020E. E. Reber, R. L. Michell, and C. J. Carter, "Oxygen absorption in the earth's atmosphere," Aerospace Corp., Los Angeles, CA, USA, Tech. Rep. TR-0200 (4230-46)-3, Nov. 1988.

[6] Guide to Intrusion Detection and Prevention Systems (IDPS) - Karen Scarfone, Peter Mell, February 2007

[7] Importance of intrusion detection system (IDS)- A. S. Ashoor and S. Gore, January-2011

[8] Deep learning approach for network intrusion detection in soft_x0002_ware defined networking- T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, M. Ghogho, October 2016

[9] Anomaly detection in ad-hoc networks based on deep learning model: A plug and play device -F. Feng, X. Liu, B. Yong, R. Zhou, Q. Zhou, March 2019

[10] A deep learning method with filter-based feature engineering for wireless intrusion detection system -S. M. Kasongo, Y. Sun, March 2019 the Terahertz Wave eBook. ZOmega Terahertz Corp., 2014.

[11] Y. Zhang, X. Chen, L. Jin, X. Wang, and D. Guo, ``Network intrusion detection: Based on deep hierarchical network and originaldata," IEEE Access, vol. 7,2019

[12] M. Tavallaee, E. Bagheri,W. Lu, and A. A. Ghorbani, ``A detailed analysis of the KDD CUP 99 data set," in Proc. IEEE Symp. Comput. Intell. Security Defense Appl., Jul. 2009

[13] Sperotto, Anna & Pras, Aiko. (2011). Flow-based intrusion detection. 958-963. 10.1109/INM.2011.5990529.

[14] System Integration and Security of Information Systems Andrii Boikoa, Vira Shendryka,P 2016.

[15] M. Hassan, M. E. Haque, M. E. Tozal, V. Raghavan, and R. Agrawal,``Intrusion detection using payload embeddings," IEEE Access, vol. 10,pp. 40154030, 2022,

[16] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall,D. McClung, D.Weber, S. E.Webster, D.Wyschogrod, R. K. Cunningham,and M. A. Zissman, ``Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation," in Proc. DARPA Inf. Survivability Conf. Expo. (DISCEX), Jan. 2000