Spring 2023

Machine Learning-Based Anomaly Detection in Cloud Virtual Machine Resource Usage

Tarun Mourya Satveli
San Jose State University

Machine Learning-Based Anomaly Detection in Cloud Virtual Machine Resource Usage

A Project Report

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

by

Tarun Mourya Satveli

May 2023

MACHINE LEARNING-BASED ANOMALY DETECTION IN CLOUD VIRTUAL MACHINE RESOURCE USAGE

by

Tarun Mourya Satveli

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE´ STATE UNIVERSITY

May 2023

Dr. Robert Chun	Department of Computer Science
Dr. Faranak Abri	Department of Computer Science
Mr. Revanth K Maddula	Software Engineer, TikTok

ACKNOWLEDGEMENTS

I would like to express our gratitude and appreciation to Dr. Robert Chun, my advisor, for his invaluable support and guidance throughout this research endeavor. The completion of this undertaking has been greatly aided by his insightful comments, constructive criticism, and constant encouragement.

I would also like to thank Dr. Faranak Abri and Mr. Revanth K Maddula, both members of my committee, for their insightful comments and suggestions, which helped us refine and enhance our work. I would like to acknowledge the assistance and resources provided by our university, which allowed us to conduct this research.

Finally, I would like to thank everyone who has contributed to this endeavor in any way, including our families and friends, who have always been there to offer support and encouragement.

Abstract

Anomaly detection is an important activity in cloud computing systems because it aids in the identification of odd behaviours or actions that may result in software glitch, security breaches, and performance difficulties. Detecting aberrant resource utilization trends in virtual machines is a typical application of anomaly detection in cloud computing (VMs). Currently, the most serious cyber threat is distributed denial-of-service attacks. The afflicted server's resources and internet traffic resources, such as bandwidth and buffer size, are slowed down by restricting the server's capacity to give resources to legitimate customers.

To recognize attacks and common occurrences, machine learning techniques such as Quadratic Support Vector Machines (QSVM), Random Forest, and neural network models such as MLP and Autoencoders are employed. Various machine learning algorithms are used on the optimised NSL-KDD dataset to provide an efficient and accurate predictor of network intrusions. In this research, we propose a neural network based model and experiment on various central and spiral rearrangements of the features for distinguishing between different types of attacks and support our approach of better preservation of feature structure with image representations. The results are analysed and compared to existing models and prior research. The outcomes of this study have practical implications for improving the security and performance of cloud computing systems, specifically in the area of identifying and mitigating network intrusions.

Keywords: Machine Learning, Anomaly detection, NSS-KDD Dataset, Cloud resources, Neural networks, DDoS attack.

Chapter 1 Contents

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

Cloud computing (CC) has widely been known for daily tasks, including web-based email, social networking platforms like Facebook, Twitter, and on-demand subscription services like Netflix, which are growing increasingly popular due to their simplicity of use and low prices. The term refers to a method of utilizing the Internet for the purpose of distributing computing resources, such as memory, CPU, disk capacity, and software applications, across many different kinds of users. Given the advantages the cloud provides, the risks and vulnerabilities associated with moving to the cloud are always changing. There have been recorded examples of data breaches, private data leaking, and cryptocurrency mining using cloud resources [1]. In order to promptly identify virtual machines that are at risk, there is a need to enhance the security architecture of the cloud. This work improves the security measures in the public cloud Infrastructure-as-a-Service (IaaS) architecture, where multi-tenancy makes resources vulnerable. The Figure 1.1 depicts the architecture of the environment in which cloud computing is executed.

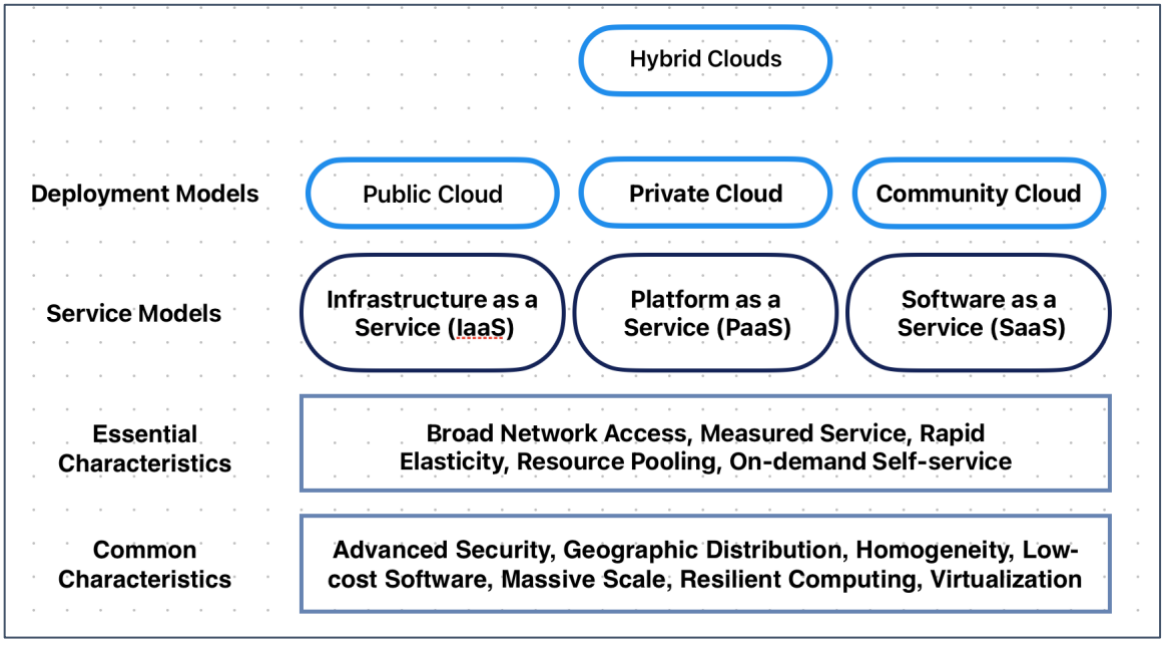


Figure 1.1 Architecture of cloud computing environment

As the Internet continues to grow, a wider variety of computer threats are becoming more common and expected. Ransomware attacks are expanding unprecedentedly, and zero-day exploits are becoming so crucial. Because a strong network today requires help from numerous security layers, rendering antivirus software and setting up firewalls are inadequate in providing appropriate protection. One of the essential layers is an intrusion detection system (IDS). This type of technology constantly monitors the system to safeguard its objective. IDSs can also be categorized based on the detection mechanism they employ. They are categorized into three types, as indicated in Figure 1.2.

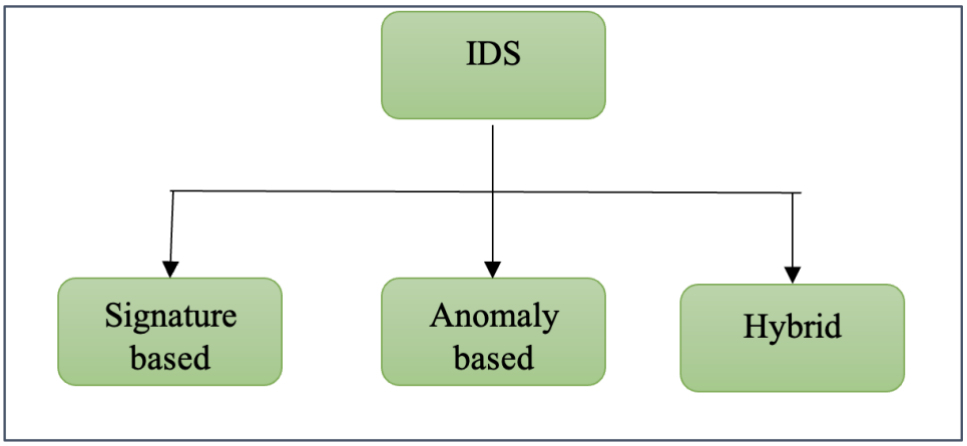


Figure 1.2 Classification of IDSs by detection method.

Anomaly and signature-based detection are the leading intrusion detection technologies (or "detection"). Signature-based detection matches IDS data to attack patterns. This method, utilized by Snort and Suricata, can only detect known threats in a database. Anomaly detection involves modelling the typical behaviour of a system and subsequently identifying anomalies within observed data. [2].

The emergence of cloud computing can be attributed to the increasing need for internet utilization, communication, and related functionalities. Its fundamental concept involves the provision of an elastic and expansive internet service through the utilization of virtualized resources. The term "cloud" or "internet" serves as an analogy for addressing interconnected systems. Historically, images of clouds have been used to symbolize communication networks, and they are still employed to denote the Internet and its underlying infrastructure when they are abstracted. The limited definition of cloud computing refers to the IT rental model infrastructure.

The required resources are acquired via the network based on principles such as flexible scalability and on-demand provisioning. In the field of cloud computing, the pay-and-use framework is acknowledged and this service could be related to information technology (IT), software, or the Internet. The following are some of the primary benefits of cloud computing:

1) Resource allocation is dynamic. Considering consumer demand in real-time, various virtual resources can be dynamically distributed or accessed in the cloud. Resource elasticity is achieved in the cloud by expanding the pool of available resources, allowing it to fulfil a request rapidly. The user can make these resources available to others if they are no longer necessary. It is common to think of cloud computing as an infinite resource pool that allows scaling up IT resources. This idea is based on how cloud computing enables users to store data on distant computers.

2) On-demand self-service. Self-service cloud computing lets clients acquire resources without contacting suppliers. The cloud system also provides a directory of application services from which customers can select the service that best suits their needs.

3) Network accessibility. Users can access the network using various terminal devices, making it available globally. Service that can be measured. According to several service categories in cloud computing, the allocation of resources is optimal and can be managed automatically. It is virtualization, a pay-as-you-go service. These days, cloud computing is quickly developing and becoming more widely understood. Due to features like great scalability, flexibility, and low operational costs, most cloud service users can learn how cloud-based software works without knowing how cloud-based software works. Instead, they submit queries to the cloud provider, and the platform operates. Waiting for computational resources' outcomes is much simpler and faster, but the cloud has issues.

Integrated intrusion detection and real-time monitoring system for networks when it finds suspicious actions, it will either alert network administrators or take preventative measures itself if an attack is detected. An IDS is typically employed to monitor system performance with the purpose of preventing behaviours that were not planned. A network's integrated intrusion detection and real-time monitoring system will look for suspicious activity and, depending on the situation, either raise an alarm or take action to defend the network. An IDS, is distinguished from other

types of network security systems by the fact that it provides a workable strategy for maintaining the safety of the network as a whole. In addition, there are other approaches that can be taken while developing an IDS. One such technique is designed to identify unusual occurrences. Searching for unexpected behaviours or data that do not conform to an existing model is an example of what is meant by the term "anomaly detection."

Lethal anomalies are obvious the moment a node stops responding, although non-fatal anomalies could go unnoticed for a while. In any event, it is preferable that the abnormality be discovered as soon as possible. If this is the case, the system has the ability to immediately respond by resuming the affected components. This helps to ensure that users are not adversely affected by the issue. Practitioners employ monitoring techniques to continuously assess the system's "health" in order to accomplish this goal. Traditional monitoring techniques frequently combine the following elements:

a) Rule sets and thresholds are established for the values of key indicators, such as memory use and CPU utilisation.

b) In-depth inquiries with stringent deadlines for finding successful solutions.

c) Health checks that are specifically customised to each individual system. These methods, which are required to be based on predetermined criteria, might not be able to identify complicated diseases that do not inevitably end in fatality.

Since multi-cloud systems are new, machine learning techniques have yet to be researched from a security standpoint in the literature. Two problems that need to be solved are the classification of different kinds of attacks and the transferability of research findings to environments with many clouds. The majority of researchers working in this area have been focusing on problems related to anomaly detection rather than giving the classification of attacks much effort. So that, this classification is necessary for determining the state of a system's vulnerabilities and the appropriate secure and preventative measures against various types of attacks.

In CC, the popularity, data security problems, and the proliferation of cloud service offers emerged as the most challenging obstacle. Most commonly used publicly accessible datasets only contain one type of attack. Additionally, research studies that employ datasets with numerous attacks tend to emphasize anomaly identification more than attack classification or categorization. In this study, rather than focusing solely on identifying abnormal traffic, we explore the viability of using machine-learning approaches to differentiate between attacks. In other words, we take them a step further by giving anomaly detection systems the ability to recognize particular sorts of attacks [3].

Cloud computing presents numerous challenges when it comes to detecting and exploiting security flaws. Furthermore, several types of research indicated that all cloud computing stakeholders believe that cloud service capabilities significantly impact cloud computing adoption. The distributed denial-of-service attack, also known as a DDoS attack, is one of the most critical hazards in today's cloud computing environment [4]. The goal of a distributed denial of service (DDoS) attack is to reduce network resources shortage are even does not allow the user access to the system from using it.

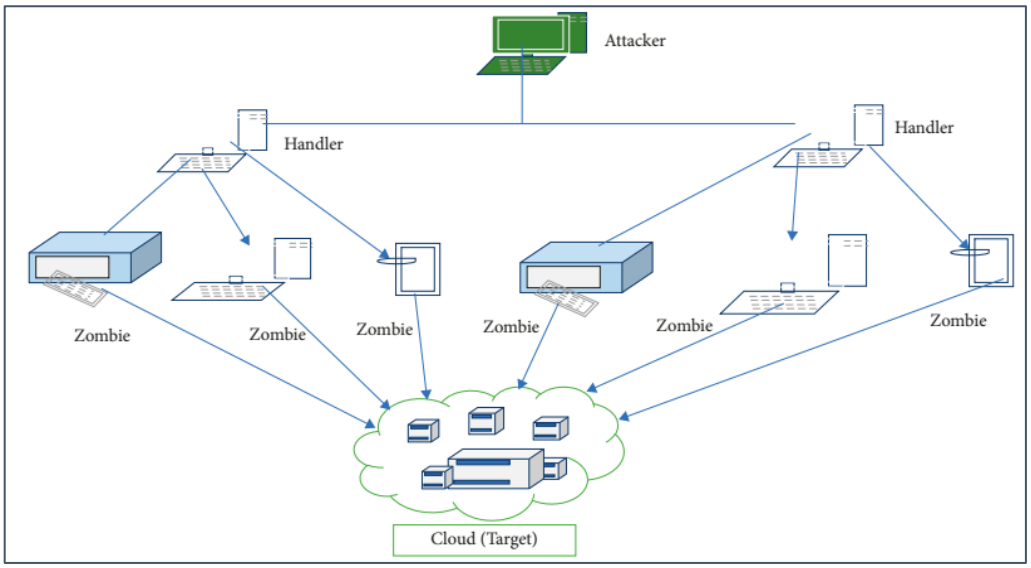


Figure 1.3 DDoS attack on Cloud [6]

The diagram depicted in Figure 1.3 illustrates a Distributed Denial of Service (DDoS) attack that has been executed on cloud computing. In this scenario, numerous systems, also known as zombies, have been employed to launch a DDoS attack on the Cloud. DDoS attacks can come

from the inside or outside and can harm cloud computing infrastructure. DDoS attacks, which flood the target with massive amounts of traffic, have the potential to grind the system to a standstill [5]. In this scenario, many systems, sometimes known as zombies, launch a DDoS attack against the cloud [7].

1.2 Research questions

In the context of the issues that were discussed in the previous part, this thesis provides answers to several research topics and raises new concerns in the field of anomaly detection:

RQ1: What network attributes are significant and relevant in anomaly detection systems research?

RQ2: What is the best way to define normal network behavior? How are system resource attacks mitigated?

RQ3: What improvements are required to improve the anomaly detection model for accessing all the network or system resources in the cloud?

RQ4: What are the options for developing an anomaly detection system that decreases overhead while improving the performance of the models?

The answers to these research questions have been uncovered after conducting a comprehensive review of recent scientific and practical activities. An exhaustive study of the relevant literature found that although numerous methods achieve a high level of precision, they each have several important drawbacks that must be addressed. Anomaly-based detection systems are a developing area of research that aims to achieve optimal detection rates while minimizing the occurrence of false positives. However, in-depth analysis and careful planning are required to design a superior detection model. As a consequence of this, this thesis has provided a comprehensive analysis of the primary challenges that are present in the field of anomaly detection.

1.3 Research Aim:

This study aims to use supervised machine learning and neural network-based techniques to find anomalies in a NSL-KDD dataset to make adversarial samples that can be used against supervised learning models.

1.4 Motivation

Cloud computing is a technology that is being developed to address the growing need for use of the internet, online interactions, and associated functionalities. In the past, clouds were commonly employed to represent communication networks. However, the picture can now relate to both the conceptualization of the internet and its fundamental framework. The term "generalized cloud computing" pertains to a computing model that involves the rental and utilization of resources through a network, depending on conditions such as on-demand availability and convenient scalability. This service may be related to information technology, software, or the internet or completely unrelated. Most cloud customers are worried about security risks such data leakage, incorrect information, and unauthorized access. Distributed IDS can be used to monitor and protect cloud-based VMs and virtual networks. This intrusion detection system also monitors system performance to avoid unintended behaviours.

In this thesis, we are building an intrusion detection system for the cloud platform. The challenge is to make sure that the IDS doesn't overload busy nodes and slows down the detection process. To solve this, the IDS needs to be flexible and adjust its architecture based on real-time resource usage information. This is because this information can aid in the prevention of security breaches. Moreover, it should also be able to detect new attacks in the cloud, which can be more demanding for resources.

1.5 Problem statement

Presently, one of the biggest challenges dealing with cloud technology is the restricted number of companies that are completely accepting it. The primary factor contributing to this issue may be attributed to the security concerns associated with cloud computing. DDoS is one such attack where multiple computers attack a target together. This attack aims to overload the system and make it difficult for people to use it. It is very harmful to cloud computing as it can prevent people from accessing the services. Rapid detection of such attacks is essential for the

uninterrupted operation of diverse cloud computing services. Accurately distinguishing these attacks is a complex challenge for ML classifications for detecting and alleviating the attacks.

During empirical studies, several researchers have found that supervised machine learning is widely used to detect and prevent Distributed Denial of Service (DDoS) attacks. However, accuracy remains a top priority for everyone due to the limitations of the models' independent variables. To address this issue, a proposed classifier has been developed that can detect DDoS vulnerabilities in cloud computing. This research aims to use machine learning to identify DDoS threats in order to improve the effectiveness of cloud computing.

1.6 Research Objectives

The stated objectives aim to enhance the accuracy of classification models for detecting and mitigating attacks by achieving reliable anomaly detection.

1) To thoroughly review the literature on anomaly detection and propose a simple detection and classification model.
2) To develop a machine learning-based and neural network-based anomaly detection classifier model on the NSLKDD+ dataset.
3) To develop a two-level framework using machine learning techniques accurately and increase classification accuracy on every attack label.
4) To assess the effectiveness of the proposed model and measure it’s performance using accuracy detection rate and couple of other metrics.

1.7 Thesis Organization

The subsequent sections of the thesis are organized in the following manner:

The first chapter discusses key principles in intrusion detection. It outlines several attacks and an intrusion detection system and gives historical context for the area. The present work outlines the role of machine learning in the discussed topic, as well as the various individual datasets that were utilized. The pros and limitations of several kinds of IDSs are discussed. Lastly, the measures that are most typically utilized are specified.

Chapter 2 discusses existing work on intrusion detection using machine learning methods. The various models are described in-depth, along with a brief explanation of how they function. Methods and outcomes are provided sequentially before being compared in a standard part to find the optimal strategies.

In Chapter 3, an in-depth explanation is presented regarding the dataset and the pre-processing methodologies employed for the purpose of feature extraction. Furthermore, the work provides an explanation of the various machine learning models employed for the purpose of detecting attacks on the NSS-KDD dataset. The chapter additionally discusses upon the proposed framework for intrusion detection.

Chapter 4 presents the results and explanation of intrusion detection of a dataset and its evaluation parameters. It compares the results with various models that are then trained along with results of the previous works to get the optimum detection quality.

Chapter 5, the final chapter of the thesis provides a comprehensive overview of the main ideas presented throughout the rest of the work. Emphasis is placed on future progress as well as the significance of machine learning for intrusion detection.

Chapter 2

LITERATURE WORK

The present section is structured as follows: the initial subsection provides an overview of the current state of the art in anomaly detection; the subsequent subsection examines the current state of the art in methods for utilizing cloud resources; the third subsection explores the utilization of anomaly detection methods in prediction; and the fourth subsection scrutinizes the related literature that has been presented.

2.1 Overview of Anomaly detection

The application of anomaly detection involves the identification of a normal or expected system activity pattern. Any deviation from the established pattern is deemed as an intrusion, thereby triggering an alarm. This solution does not require any updates and does not even require the existence of a database. Although it can detect previously unknown dangers, it also produces many false positives that are difficult to process. Gathering information about the attack is more challenging because it needs a distinct signature. This increases the overall difficulty of the task. The method for detecting anomalous intrusions tackles both the issue of false alarms and the issue of a poor detection rate. A machine learning hybrid that classifies data using support vector machines and K-means clustering. It had a far higher detection rate, a lower rate of false-negative alarms, and a much higher rate of false-positive alarms than comparable systems. Several researchers' primary interest was the investigation of odd occurrences.

This study emphasizes using machine learning (ML) methodologies to find dynamic anomalies in software rather than static ones. The use of machine learning (ML) methods to identify operating anomalies in distributed software systems has received minimal investigation, although ML is frequently used to address a wide range of problems in many domains. This is true even if ML is commonly employed. Therefore, we talk about the following: A lot of research is being done on intrusion detection in addition to the machine learning methods already existing for detecting anomalies, as shown in Figure 2.1.

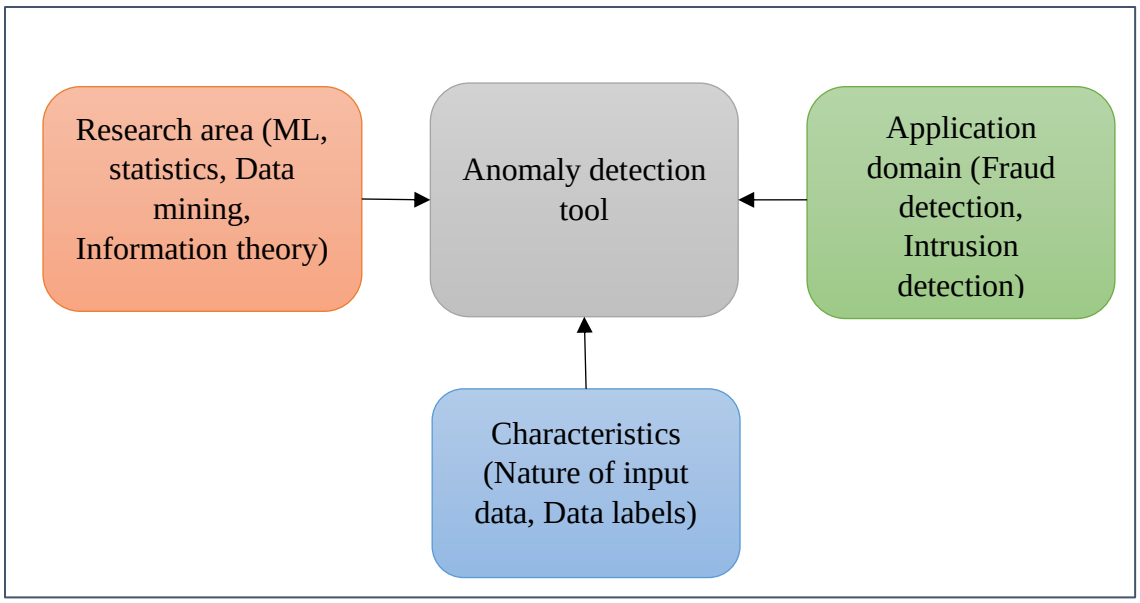


Figure 2.1 Overview of Anomaly detection tool

This topic is strongly related to intrusion detection, as changes in typical behaviour might point to intentional or accidental attacks, flaws, or malfunctions. ML approaches for precise anomaly identification have huge potential because both research fields are similar. This also implies that the issues in these two fields are identical. The benefit of using ML-based approaches for detecting known and undiscovered abnormalities Nevertheless, they note that it is difficult to distinguish between normal and aberrant behaviour.

Going a step further, Omar et al. (2018) [8] discuss the challenge of differentiating between normal and truly deviant behaviour while still open-ended. Furthermore, they discover that supervised methods outperform unsupervised ones when the evaluation solely includes known attacks. This is consistent with studies in other domains that show supervised techniques perform better. Support vector machines (SVM), clustering, and an ant colony algorithm were incorporated to determine whether an attack on the network was normal. With this combined approach, the accuracy rate was noticeably greater. SVM with DGSOT. The abovementioned strategy contrasted with the Rocchio bundling and random selection methods. The outcomes significantly outperformed the Rocchio bundling method.

Dehraj and Sharma et al. conducted a survey on autonomous software systems that takes a more general approach (2019) [9]. It gives a detailed explanation of autonomic decision-making

and its importance for several applications. These applications cover massive volumes of data, wireless sensor networks, the Internet of Things, intrusion detection, cloud-based data protection, and other areas where human supervision could be more practical.

Fernandes et al. (2019) [10] presented an article regarding artificial immune systems as a different general approach. The article explains applicable principles and reviews several studies using such methods to address computer security issues. Both strategies either don't pay attention to the particular methodological components of anomaly detection or don't consider the unique requirements of cloud computing systems. There are further examples that go beyond cloud-based networks and anomaly detection.

Ramachandra et al. (2016) [11] conducted a survey on the topic of anomaly detection for cloud computing infrastructures, which is the sole existing study on this subject. The results of this investigation shed light on how log data might be valuable in identifying abnormalities in cloud computing. The study's scope is limited to the identification of anomalies in log data, with a specific emphasis on intrusion detection. As a result, a comprehensive understanding of the subject matter cannot be obtained from the examples provided. Prashanthi and Keiko et al. (2019) [12] mentioned these three topics as vulnerability and risk areas. (IaaS). According to the article's author, "new virtualization-aware security procedures are required to provide preventative security solutions for the cloud system." The security architecture supporting cloud computing includes an IDS. This is due to the way it serves as a protective measure against unauthorized access. Mehmood et al. examined the benefits and drawbacks of numerous intrusion detection systems integrated into cloud computing to defend against cyberattacks.

Katsaros et al. (2020) [13] offer a service-oriented system for monitoring virtual and physical infrastructure data. Nagios is given an extension in the form of a RESTful interface inside the parameters of the method that has been suggested. The mosaic API is an application programming interface that may be used to create a monitoring system specifically designed for the cloud. The monitoring infrastructure emphasizes scalability, multi-tenancy, dynamism, and simplicity. The majority of the monitoring systems mentioned above focus on one functional layer. They are inappropriate for real-world domains where modifications to one layer impact other categories.

2.2 Cloud resource usage planning

Most of the time, rule-based and machine learning (ML) methods are used in cloud resource utilization planning. On the other hand, predefined rule systems do not account for dynamic changes in the environment and disregard the prediction component entirely. This is because such systems do not account for the likelihood of error. As a result, they are noticing a decrease in total utilization. Machine learning-based prediction systems are increasingly popular owing to their relatively high precision and adaptability. This is because these algorithms enable dynamic resource utilization planning. As a result, the primary focus of our research will be on approaches that employ machine learning techniques.

The authors of [14] suggest a system based on learning automata that can accurately estimate the stresses placed on virtual machines. To generate more accurate predictions, the system combines various machine learning algorithms. The method was validated using data from around 500 host machines and over 1,000 virtual machines. The data were collected from computers utilized for various purposes over twenty-four hours at five-minute intervals. Compared to the other four prediction algorithms studied, the solution performed the best.

The article [15] presents an ensemble model that can forecast workload and compare it to the baseline. The trials were conducted to make projections that were produced a few minutes in advance. The incorporation of machine learning algorithms, specifically K-Nearest Neighbours (KNN) and Decision Trees, in the proposed solution resulted in a notable enhancement in accuracy. The resulting ensemble exhibited a significant improvement in the outcomes. . The initial analysis found a Root Mean Square Error (RMSE) of 0.37, and the suggested technique demonstrated a reduction of 6.22% in errors connected with CPU utilization and a decrease of 17.3% in errors associated with RAM usage.

The authors of [16] study neural networks' ability to effectively estimate the utilization of many resources. They describe a model that trains network weights using a combination of the Genetic Algorithm (GA) and the Particle Swarm Optimisation (PSO) techniques and then makes predictions (a few minutes in advance) using a Functional Link Neural Network (FLNN). The findings of the studies show that the combination of models developed gives more accurate results than conventional methods.

2.3 Related works:

Several studies on DDoS solutions are discussed. Similarly, the study we undertook conducted a more in-depth analysis and provided more exact technical information than earlier investigations. According to Cao et al. (2017) [17] research, a cloud virtual machine (VM) centre should be able to supply multiple tolerable service applications at the same time. This objective is achieved by using the same physical hardware. A VM service utility function has been designed to estimate the future load of service requests using the grey wave forecasting technique. This function takes into account both resource requirements and service priority. Each virtual machine (VM) hosted on the physical computer will dynamically configure the underlying physical resources to boost the physical computer's service utility value.

According to Shafi et al. (2019) [18], anomaly detection in the cloud where analytical programs run and in IoT devices is only possible due to the limited computing power of IoT devices. Software Defined Networking (SDN) makes this claim because its centralized management promises improved network management. Collocated fog infrastructure and cloud infrastructure are two different scenarios that can be controlled by a framework for software-defined machine learning (ML) anomaly detection (network edge). Compared to cloud infrastructure, fog nodes that use machine learning for anomaly detection produce superior computational (attack detection) outcomes.

According to Cholakoska et al. (2021) [19], one of the most critical and pressing concerns that the modern world must face is increasing the security of the Internet of Things (IoT). A fundamental difficulty that must be overcome is the ability of the many diverse devices that can be connected to the Internet of Things to interact securely without sacrificing their overall performance. This ability has become increasingly important as the Internet of Things has become more popular. Most gadgets have restricted power levels and cannot do intricate computer calculations. The utilization of technologies that can identify abnormalities and intrusions becomes obligatory. Identifying anomalies in networks like this is the focus of this article, which presents several distinct machine-learning approaches. The findings that were gathered demonstrate high accuracy and precision (97%), in addition to a rapid execution speed.

The Internet of Things (IoT) system is vulnerable to a wide range of attacks and anomalies, which could eventually lead to the system failing. According to Sahu et al. (2020) [20], the use of IoT infrastructure has increased exponentially across all industries, increasing threats and attacks. This study anticipates several abnormalities based on an attribute extrapolated from the data set. The study was carried out to investigate a problem. The performance of two distinct machine learning categorization models is compared here. Both of these models are used. Regression analysis based on logarithms and artificial neural networks are used as classification algorithms. This work can be utilized to prevent attacks and spot threats and anomalies that occur in the IoT solutions.

Sa et al. (2022) [21] suggested that the Internet of Things (IoT) environment is characterized by a significant volume of data generated by sensors and other connected devices. The primary obstacle confronting applications reliant on the IoT pertains to the security vulnerabilities arising from the extensive volumes of data that are in transit. Anomalies in data may arise due to various factors such as information theft, information snooping, or IoT device malfunctions. The present study introduces a novel approach for detecting network anomalies, which leverages the random forest machine learning algorithm in a distributed setting (i.e., a cluster of machines) through the utilization of SPARK. The aim is to develop a technique that can effectively and efficiently identify anomalies within vast quantities of data on the UNSW-NB15 Internet of Things dataset.

Xu et al. (2014) [22] claims that in cloud computing, achieving a reasonable balance in resource utilization, overhead costs, and time restrictions requires effective scheduling and management of virtual machine resources to address the challenge of virtual machine resource scheduling. By efficiently scheduling and controlling virtual machine resources, this can be achieved. After a complete multi objective review of the virtual machine resource scheduling model was suggested, the problem was resolved using the conventional multiobjective optimization approach. Machine learning can be used to spot anomalies in the data that define the transport component contained inside the cellular network. This was one of the potential applications for machine learning, and it was a crucial one.

The ensemble-based Isolation Forest and the autoencoder neural network are fundamental algorithms for finding abnormal behaviour patterns. These two algorithms are combined when used together. The time required for domain experts to find abnormalities inside the network can be significantly reduced using machine learning. One of the algorithms that were put through its paces was able to differentiate between key performance indicators that were normal and those that were abnormal, in addition to being time-efficient (KPIs). This capability is necessary for conducting root-cause analysis, and it entails employing a drill-down method to establish which component in the system is malfunctioning.

According to Kaur et al. (2013) [23], intrusion detection has become increasingly common over the past two decades, mainly when an incursion represents an effort to break into or misuse the system. The first type, known as misuse or signature-based detection, and the second type, known as anomaly detection, are the two primary sorts based on invasions. In this article, we cover anomaly detection approaches based on machine learning, one of the different kinds of techniques available.

According to Russo et al. (2021) [42], the approach known as anomaly detection entails finding unusual data samples inside datasets. Unlabelled datasets are necessary for supervised machine learning models. Despite the availability of various tools and machine learning models for automated anomaly detection in academic research, the research community focused on environmental systems lacks a thorough, impartial, and organized comparative study. The results show the enormous value of expert-based data annotation for automatically detecting abnormalities.

The scalability of the top-performing techniques and how they stack up for field trial viability. Mitra et al. (2020) [25] proposed a strategy that was based on machine learning to locate LSM flaws (ML). This machine learning strategy uses a set of techniques that, as they progress, become increasingly complicated regarding their capacity to identify aberrant values as the level of complexity rises. Some approaches that use randomized train/test splitting have data leakage issues that can be resolved using non-randomized training and validation data splitting.

According to Cheng et al. (2021) [26], a double auction is one of the most promising methods for allocating virtual machine (VM) resources in two-sided cloud marketplaces since it

can boost VM resource usage. However, most cloud auction systems ignore the protection of bidder privacy and trade fairness during the auction process and instead presume that the auctioneer can be trusted entirely. According to earlier studies, some cryptographic tools may be able to tackle the problems mentioned above, but their subpar performance makes it challenging to put those strategies into reality. An intelligent contract-based fair-trade method prevents bidders from abandoning an auction without paying the price. The thorough studies show that SF-DAC reduces computation and transmission expenses by order of magnitude compared to earlier technologies.

According to Gadal et al. (2022) [27], a crucial component of Data Mining is anomaly detection, which reveals hidden patterns in the data most susceptible to attack. This aids in the detection of network intrusion. The accuracy of the anomaly detection rate can be increased using algorithms like minimal sequential optimization (SMO) rating and hybrid K-mean arrays. An algorithm for detecting anomalies using machine learning (ML). With ML, detection rates are increased, false-positive alarm rates are decreased, and intrusion classification accuracy can be increased. Future data mining systems may consider hybrid algorithms for anomaly detection because real-time processing is anticipated to slow down drastically in these systems. The hybrid algorithm's ability to generate appropriate detector numbers with negligible FAP and reasonable detection accuracy is used as justification. Given the low FAP, it is generally anticipated that this method will take less time to pre-process and process data than other algorithms.

Data must move quickly and securely for high-speed Internet through optical networks (such as fiber cuts). Abdelli et al. [28] created a single system that combines an attention-based bidirectional gated recurrent unit algorithm and an autoencoder-based anomaly detection method (2022). The researchers made this system. When the autoencoder finds something wrong, the second method will find the problem and determine where it is. Real-world operational data and a wide range of attack anomalies will be used to test how well our suggested method works and see if it can be used in the real world. Still, optical fibers, which are used to send data and connect billions of people worldwide, can be affected by physical attacks like visual eavesdropping (also called "fiber tapping") and hardware failures. Such abnormalities may interrupt networks, resulting in significant financial and data losses, jeopardizing optical networks' secrecy by allowing unwanted access to the data transmitted, or gradually impairing network performance.

According to Miao et al. (2018) [29], the utilization of cloud platform centres is recommended for the implementation of anomaly detection techniques. This is due to the allocation of processor resources based on peak load demand, the availability of a unified strategy, and the sudden surge in abnormal rate caused by a shift in resource demand. To alleviate the resource demand-induced abnormal rate increase, this was done. This suggestion was made because anomaly detection methods and cloud platform centre applications should be offered. Cloud platform centre ADGWT detects grey waveform anomalies. The grey waveform detection algorithm enables the early detection of irregularities that could eventually affect a virtual machine. The VM's utility function, which is unusual in nature, serves the purpose of achieving balance between the resource requirements and the significance of detecting anomalies. Each virtual machine is constantly configured to maximize the virtual computer's abnormal utility value, which results in each virtual machine having a different configuration. The simulation experiment shows that the suggested strategy can boost the cloud platform processors' abnormal rate and speed up user requests.

According to Lin et al. (2016) [30], one of the three essential core service models that may be obtained through cloud is Infrastructure as a Service (IaaS). It provides users access to computation and storage resources through virtual machine instances. Comprised of virtual machine instances, this platform offers clients access to various computational and storage resources. The rapid growth of cloud computing has resulted in an increase in the number of application systems that are being set up on the IaaS cloud platform. When IaaS cloud computing platforms experience abnormalities, all application systems are rendered incapable of functioning correctly as a direct consequence of this. It has been suggested that the IaaS platform implement a VM instance intrusion detection system to discover virtual machine instances that exhibit strange behaviours. This would increase the dependability of the IaaS cloud computing platform. The data gathering, transmission, storage, and anomaly detection modules comprise the virtual machine instance system. According to experimental data, the suggested virtual machine instance anomaly detection method performs well.

According to Shi et al. (2022) [31], CC represents a major step in the advancement of distributed computing as a commercial deployment with exciting prospects. One crucial service mode in cloud computing is "infrastructure as a service," or IaaS. It uses virtualization technology

to combine enormous amounts of resources from several locations to form a single pool of resources. As a result, it is now possible to centralize resource management and utilization. When using IaaS, all resources are made available via virtual computers (VM). Changing the distribution of virtual machines is required to use existing resources better, save customers money, and reduce the time they spend computing—a cutting-edge multi objective optimization technique for dynamic resource allocation to ensure distribution stability across several virtual machines. When calculating the cost of relocating virtual machines and the stability of new virtual machine placement states, it is critical to consider both the current state and the data that may be used to forecast the future load of each application. The findings of the study can be used to reach the following conclusion.

Yan et al. (2015) [32] proposed that the technique for providing resources to virtual machines be based on cloud computing company characteristics. Based on the user's description of the business elements and the cloud state, this technique provides settings for the resource scheduler computing data centre and the cost distribution of cloud resources. After incorporating the system model for the algorithm, the model is developed based on the relevant problem. Subsequently, the algorithm's resource allocation scheme is generated, and its performance is simulated to verify its outcomes. The findings indicate that when implemented in a cloud-based platform, the proposed algorithm can improve resource utilization while significantly lowering the computational characteristics of various business users' cloud computing costs and improving the cloud user service experience.

Chapter 3

METHODOLOGY

This thesis aims to develop a framework for anomaly intrusion detection in a cloud context. The current hypothesis states that by utilizing past user information, an algorithm capable of detecting abnormalities might be constructed using an existing machine learning method and also provides an explanation of the various ML models employed for the purpose of detecting attacks. This chapter also provides an in-depth explanation of the proposed cloud anomaly implementation element. The proposed cloud anomaly IDS is designed in stages and modules discussed in this section.

3.1 System Architecture:

As NSL KDD offers many features while efficiently eliminating redundancy, a modified version of the KDD '99 Dataset is used to develop an effective anomaly detection system. The Scikitlearn Python library is generally used for analysis. ML Algorithms are developed to evaluate the efficacy of Python-based anomaly detection. The relationship between label division and algorithm performance was examined to improve the models while focusing on Feature Selection and Feature Engineering. Comparable machine learning models are utilized, and the outcomes are evaluated using f1-scores and accuracy, as illustrated in Figure 3.1. Nonetheless, because the labels are skewed, the f1-score (weighted) takes primacy.

The dataset has 42 features, 41 of which are original datasets. The characteristics concern network connections formed, information shared details, and information about faults that occur during the established connection. The labels have been classified as "normal" in the case of an authentic connection and divided into four "attack" classes in the event of an intrusion; "DoS," "Probe," "R2L", and "U2R", each with its own set of attack types. The models are tested on:

1. The original label types as in Multi-Class Classification,
2. As a binary classification, with "Attack" and "Normal" serving as the new labels,
3. As Multi-Class Classification, wherein the labels are partitioned into four sub-classes of attacks and the corresponding models are trained.

The proposed cloud anomaly IDS is designed in steps and modules, explained in this section, and illustrated in Figure 3.1.

3.1.1 Dataset:

In order to effectively train machine learning algorithms, a significant quantity of data is necessary. The quality and quantity of data are crucial factors in every machine learning evaluation. Undoubtedly, these concerns are significantly reliant on data: higher quality data often exceeds more efficient algorithms, which are of crucial significance. The NSL-KDD dataset [34] has solutions to most of these problems.

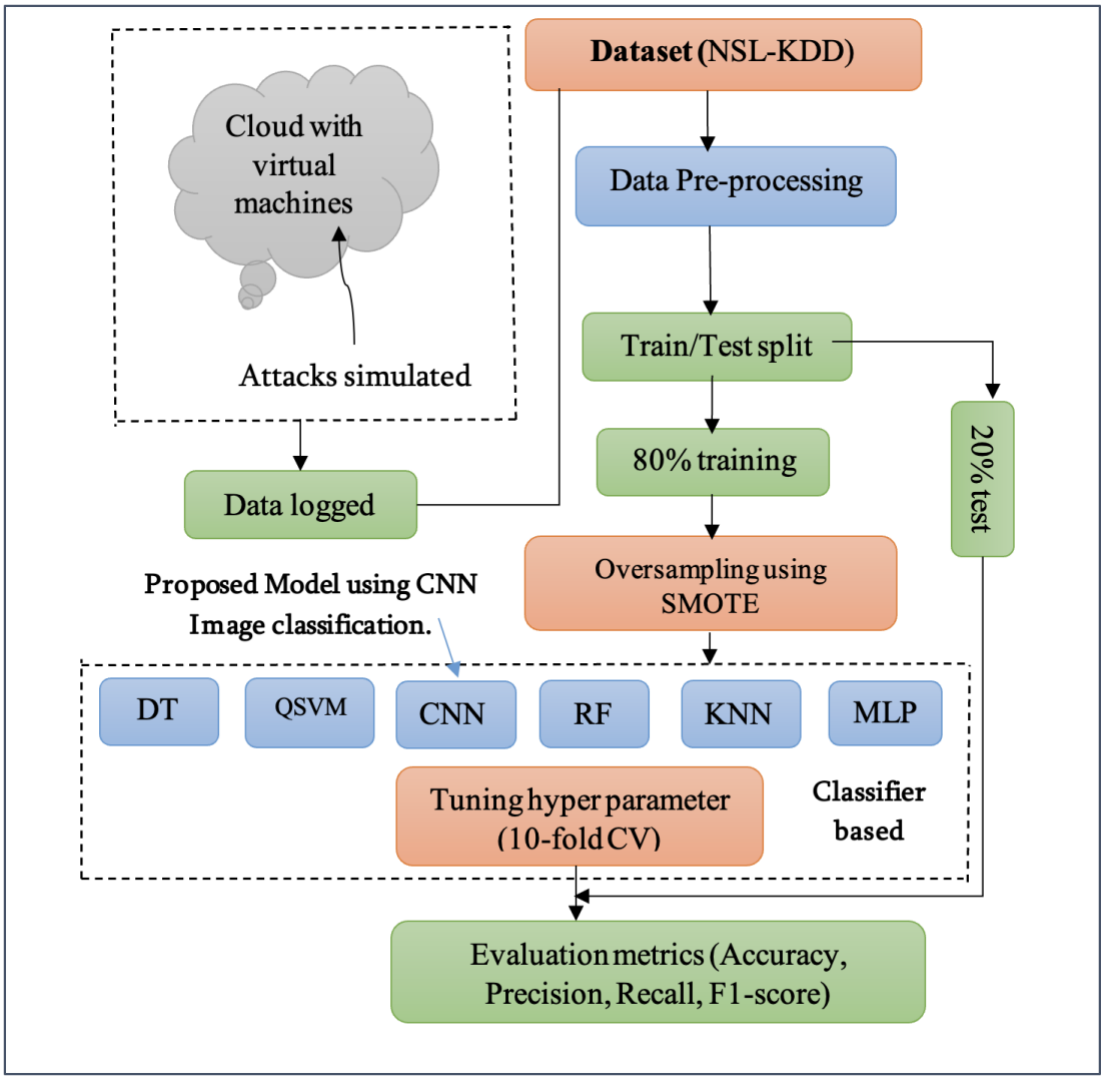


Figure 3.1 Anomalies Detection framework using machine learning

The KDD Cup 99 dataset eliminates unnecessary and redundant entries, constituting 78 percent and 75 percent of the total data, correspondingly. As a result, the number of train and test set connections drops from 805050 in the KDD Cup 99 to 148517 in the NSL-KDD. Another criterion

for categorizing relationships is their degree of difficulty. This category may help classifiers recognize the most difficult-to-see attacks during training. Regrettably, the NSL-KDD is imperfect and has many of the same issues as the KDD Cup 99. For example, the attacks contained within this dataset are deemed to be somewhat out of date and do not effectively reflect the threats that might be encountered in a modern network. Additionally, the dataset was created in a way that makes it easy to manipulate and change the information, which makes it less reliable for studying cyber-attacks. Table 3.1 compares the occurrences of the different result labels in the dataset.

Table 3.1: Frequency of outcome labels

KDD'99		NSL-KDD	
Outcome	Count	Outcome	Count
smurf.	280790	normal	77054
neptune.	107201	neptune	45871
teardrop.	979	snmpgetattack	178
pod.	264	httptunnel	133
perl.	3	sqlattack	2
spy.	2	spy	2

3.1.2 NSL-KDD Dataset Analysis

The given dataset contains information about Internet traffic that has been monitored by an intrusion detection network. Each record in the dataset includes 43 features. Out of these 43 features, 41 features are related to the traffic input, while the remaining two features are labels. One that indicates if the input is normal or an attack, and another that provides a score based on how severe the traffic input is. Table 3.3 enumerates the subcategories of each attack type that were identified in the dataset.

This dataset contains occurrences of multiple categories of attacks, but their distribution is unbalanced. The major proportion of entries within each dataset pertain to normal network traffic, whereas instances of U2R and R2L attacks are highly infrequent. This distribution accurately reflects the current landscape of internet traffic attacks, in which DoS attacks are the most prevalent and U2R and R2L attacks are relatively uncommon. DoS differs from the other three attacks in

that it endeavours to shut down a system by blocking all traffic flow, whereas the other three aim to infiltrate the system stealthily.

The KDDTest-21 is a version of the KDDTest+ dataset that excludes the traffic records with the highest difficulty score, 21. It is important to observe that the traffic records in KDDTest-21 are already part of the test dataset, as it is a subset of it. In Fig. 3.2, the Attack Class distribution of the Test+ dataset is represented as a histogram.

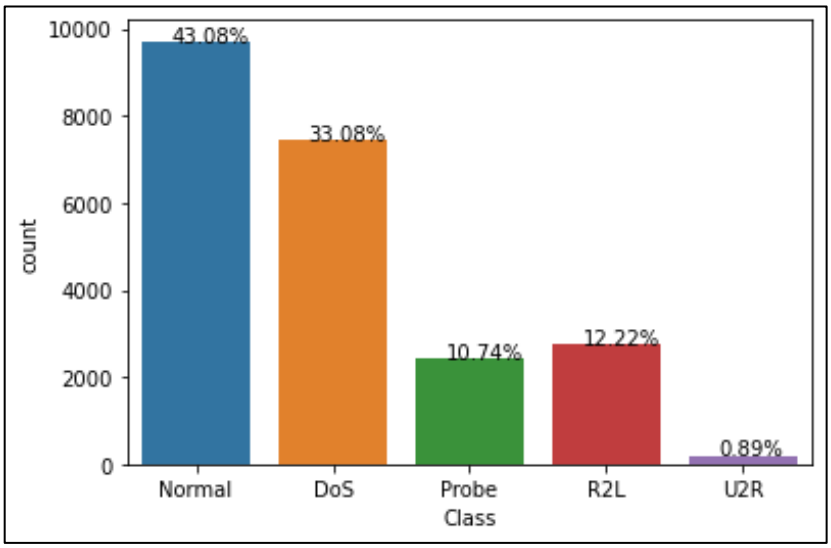


Figure 3.2 Attack Class Distribution KDD-Test+ dataset.

Table 3.2: KDD Train & Test Data Distribution table

Dataset	Number of Records					
	Total	Normal	Dos	Probe	U2R	R2L
KDDTrain+	125973	67343 (53%)	45927 (37%)	11656 (9.11%)	52 (0.04%)	995 (0.85%)
KDDTest+	22544	9711 (43%)	7458 (33%)	2421 (11%)	200 (0.9%)	2654 (12.1%)

3.1.2.1 Dataset Split

In preparation for model training and evaluation, the KDDTrain+ dataset is initially partitioned into two sections utilizing the train-test split methodology. The process of fitting a model begins with the training subset, wherein the ideal model develops assumptions and establishes their correlation with the predicted outcomes. The process of evaluation involves providing the test data into the model and comparing the expected attack categories with the

accurate labels. The process of partitioning the complete dataset into two distinct subsets, namely an 80% training subset and a 20% testing subset, is referred to as data splitting. Table 3.2 represents the Data distribution across both KDDTrain+ and KDDTest+ datasets.

Table 3.3: Different subclasses of each class of attack

Classes	DoS	Probe	U2R	R2L
Sub-Classes:	apache2	ipsweep	buffer overflow	ft write
	back	mscan	loadmodule	guess passwd
	land	nmap	perl	httptunnel
	neptune	portsweep	ps	imap
	mailbomb	saint	rootkit	multihop
	pod	satan	sqlattack	named
	processtable		xterm	phf
	smurf			sendmail
	teardrop			Snmpgetattack
	udpstorm			spy
	worm			shmpguess
				warezclient
				warezmaster
				xlock
				shoop
Total:	11	6	7	15

Each traffic record in the dataset contains a number of features that provide information about the Intrusion Detection System's (IDS) interaction with the traffic input. These characteristics can be classified into four main categories: intrinsic, content, host-based, and time-based.

3.1.3 Data Cleaning and Pre-processing

Prior to dataset processing, a pre-processing step is necessary to adequately handle the presence of both numerical and non-numerical instances. The step stage has become redundant as the NSL KDD dataset has been used. However, it is essential to perform data cleaning at this point to eliminate any potential duplicate entries that may remain in the dataset. The implementation of the transformation for the sci-kit-learn estimator (classifier) is typically achieved through the use

of a one-of-K or one-hot encoding technique, due to its compatibility with numerical inputs. By means of making use of this approach, every category characteristic that comprises m potential inputs is transformed into n binary characteristics, where only one of those characteristics is active at any particular point in time.

Null Values elimination: All critical analysis steps, such as charting and model fitting, are comprehended through the lens of null values. Because the presence of null values in the data can lead to incorrect conclusions, they must be eliminated from the set using the *dropna*() method if they are present.

Class labelling:

This process transforms multiple class labels into binary labels, simplifying the data retrieval process and improving the effectiveness of model training.

Data Transformations: The initial set of transformations deal with the classification based on the "class" label of each record in the dataset. It adds a column that encodes 'normal' values as 0 and any other attacks value through 1-4. We will use this as our classifier on our model that identifies any attack. Irrespective of the subclass or type of attack, they will all be categorized as either Dos: 1, Probe: 2, R2L: 3, U2R: 4, and normal connections as 0. To create a more detailed prediction model, we categorized each attack by its attack type, as shown in Table 3.2. A dictionary containing all attack classifications is used to map each subclass of attack to its primary class.

Data Inspection: To extract meaningful insights from the dataset, the features are first analysed for correlation. The resulting relevant features are then used to train the perplexity-based classification algorithm. Based on protocol is one way to establish initial buckets for categorizing our data during network traffic analysis. Below Fig 3.3 represents the attack by protocol classification. Based on our data analysis, this reveals that the majority of assaults will target a particular protocol. Also, there are a number of cross-protocol assaults (Satan, nmap, and ipsweep). The distinction between each protocol variant is the focus here. Our first impression is that protocol may be helpful for identifying the type of traffic we are observing and is an essential feature for our model prediction.

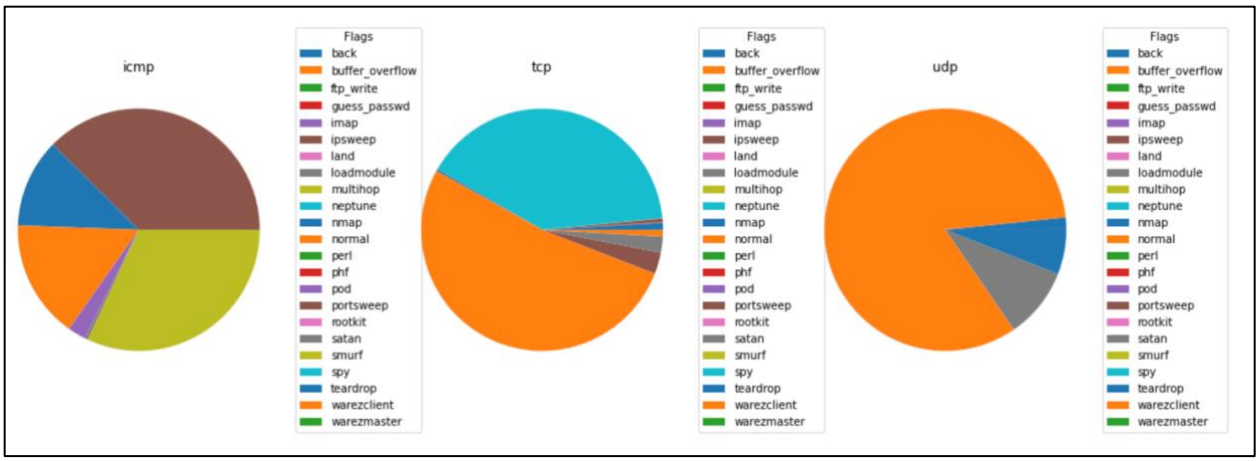


Figure 3.3 Attack Classification based on the service protocols

3.1.4 Features scaling

Rescaling or normalization of features is often necessary in machine learning algorithms to avoid the influence of features with larger values on the final results. Compute the mean value for each feature, and obtain the standard deviation. Once the scaling process is complete, each feature's mean will be zero, and the standard deviation will be one. Once the data set's features correlate, they will be trained into the classification algorithm to extract significant features from the data. Regardless of the type of attack used, the data set will be converted to binary form to be classified; each attack will be named 1, while the anomaly will be 0.

Import *LabelEncoder* is used to import label encoder from the sklearn packages, while pre-processing is imported using the pre-processing code from *sklearn. preprocessing.LabelEncoder*() was used to load the dataset column that must be labelled and encoded. The transform () technique normalizes the values in the label column. The intended value must be provided within the transform method to obtain the encoded value. The *fit_transform*() function is used to return the label column's encoded value. Following the label modification, the next phase is initiated.

3.1.5 Features Selection

Feature selection is a method to choose important data while avoiding redundancy or irrelevant information. This technique helps to identify the most significant data that represents a specific situation while minimizing the loss of accuracy. Two alternative reasons for restricting the number of characteristics were investigated: In the context of feature selection, it's important

to exclude irrelevant features that may spuriously relate to target classes without reflecting the true nature of the relationship. Overfitting, a common problem in decision tree classifiers, is also related to this issue. Second, many characteristics may dramatically increase calculation time without improving the classifier.

The utilization of ANOVA F-test for Univariate Feature Selection is a statistical approach that aims to determine the most significant features within a given dataset. The technique can be employed to determine the most noteworthy classification-related features in the context of the NSL-KDD dataset. Utilizing the ANOVA F-test, the F-value and p-value of each feature are calculated individually. The F-value quantifies the ratio of between-group variance to within-group variance, whereas the p-value indicates the statistical significance of the F-value. Subsequently, these above characteristics may be used to train a machine learning algorithm in order to predict the category of assault within the NSL-KDD dataset.

After the univariate selection process, a feature elimination technique known as recursive feature elimination (RFE) was applied with the number of features as a parameter to choose the relevant features from the internet traffic data. Recursive Feature Elimination (RFE) is utilized not only for acquiring the features that were previously chosen, but also for obtaining their respective ranks. For each type of assault, the authors collected a different number of features: Twelve for DoS, Fifteen for Probe, Thirteen for R2L, and Eleven for U2R. Finally, 13 significant features were identified for each attack category through a thorough investigation of data analysis and consideration of the Recursive Feature Elimination (RFE) results.

3.2 Multi-Class Attack Classification:

Figures 4.3 and 4.4 show the train data and test attack count percentage graphs respectively. It is clear that in train data attack normal class is 43.08% AND DoS class is 33.08%, whereas in test data attack normal class is 53.48% AND DoS class is 36.46%.

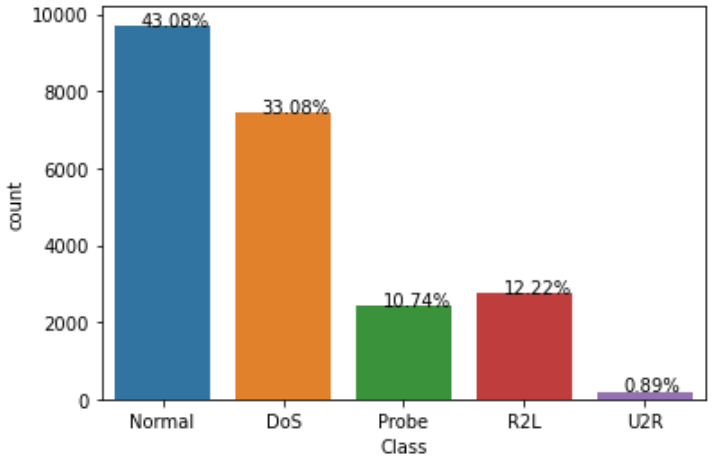


Figure 3.4 Train data attack count graph

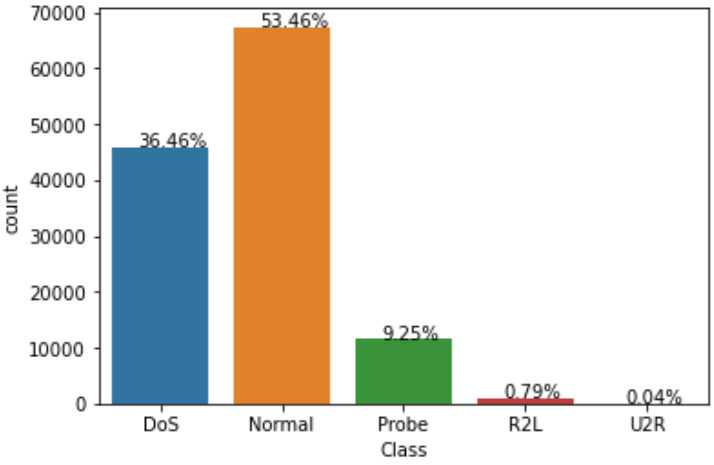


Figure 3.5 Test data attack count graph

3.2.1 SMOTE Oversampling

Synthetic Minority Over-sampling Technique, or SMOTE for short, is a machine learning technique for addressing unbalanced datasets. Datasets that are unbalanced lack representation for the minority class because one class is more prevalent than the other. SMOTE aids in resolving the issue of class imbalance and enhances the performance of the classifier by producing more representative data for the minority class. SMOTE uses the minority class instances already in existence to create synthetic data points for the minority class in order to correct this. The program chooses a representative of the minority class at random and determines its closest neighbours. Interpolating between the chosen instance and its neighbours in the feature space, it then creates

new instances. Until the desired ratio of minority to majority class instances is attained, this process is repeated.

Based on the Fig 3.3 and 3.4, it can be interpreted that the NSL-KDD dataset had imbalanced classes that affects the performance of the models in accurately classifying the attacks. Although the NSL-KDD dataset is an enhanced version of the previous KDD'99 dataset, it still exhibits class imbalance, whereby the normal class constitutes a significant majority of instances, while the remaining instances represent various attack types. To mitigate the issue of imbalanced class distribution, SMOTE can be employed to boost the number of observations in the minority categories. The minority class are first identified based on the analysis, which in this case are the U2R and R2L records, and apply the SMOTE algorithm only to the training set's minority class. The method aids in generating a dataset that is more inclusive, thereby enhancing the classifier's efficacy in handling the minority classes.

3.3 Classifier Models

The framework employs six algorithms from which the best one would be the opted for the validation analysis. The classifiers are Random Forest classifier (RFC), K-Nearest Neighbour (KNN), Decision Tree classifier (DT), Quadratic Support Vector Machine (QSVM) and MultiLayerPreceptron (MLP), which is a neural network classifier. Our suggested deep learning multi-class model is built on a Convolutional Neural Network (CNN) for classifying attacks, and we evaluate how well it performs in comparison to existing models.

3.3.1 Random Forest classifier:

One suitable method for classifying attacks on the NSL-KDD dataset is the Random Forest algorithm. This is because it can effectively handle datasets with many features and prevent overfitting. A bagging technique version, the random forest algorithm. The building of a statistically independent forest of decision trees necessitates the use of bagging and feature randomization. Feature randomization produces a random set of features, resulting in significantly lower correlation in decision trees. One of the primary distinctions between decision trees and random forests is the way they consider feature splits. While decision trees evaluate all potential feature splits, random forests only consider a portion of them. A group of decision trees is used to

train the Random Forest model on the training set. Each decision tree is built by choosing at random a subset of features and then splitting the data according to the values of those features and subsequently processes the data by checking certain features at each node and following the corresponding branch until reaching a leaf node, which contains the predicted label. The outcome is a collection of decision trees that forecast the category of an instance's class label to classify network traffic into multiple attack groups like DoS, probing, R2L, and U2R.

Before training, the three basic hyperparameters employed by random forest techniques must be specified. Among these factors are:

- Size of each node.
- Total number of trees.
- Total number of sampled features.

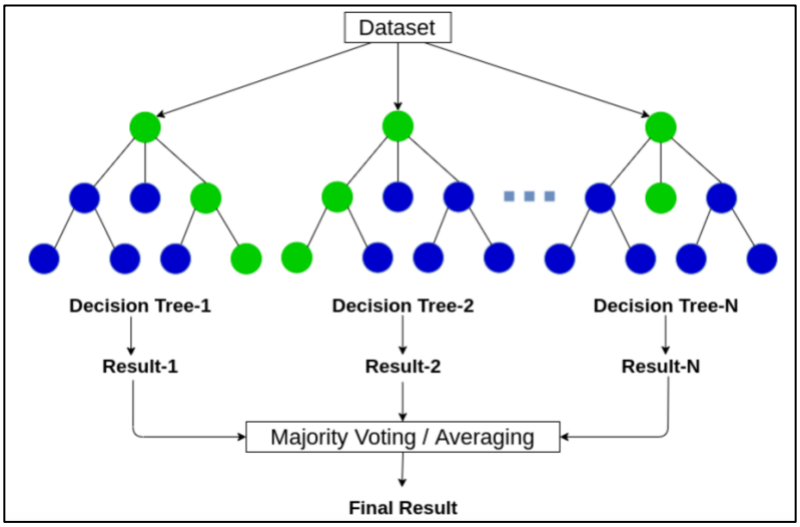


Figure 3.6 Random forest classifier model'

3.3.2 Decision tree classifier:

In order to categorize data, a DT can be used to iteratively partition the features present in the dataset. Decision tree classifiers take the form of a flowchart or a tree, as depicted in Figure 3.3. The attribute values determine the tree divisions, and each node in the flowchart or tree indicates a feature in the dataset. The decision tree algorithm looks at the entire NSS-KDD dataset and tries to identify the most important feature that can help to differentiate the attacks. This is done by calculating a metric called Information Gain for each feature. The decision tree's root node

is determined by selecting the feature that has the most Information Gain. Subsequently, the dataset is then split into two subsets according to the values of the root node feature. For instance, if the root node feature is "number of failed login attempts", the dataset may be split into two subsets: one with failed login attempts greater than a certain threshold, and one with failed login attempts less than or equal to the threshold. For example, the decision tree may start by partitioning the dataset based on the feature "number of failed login attempts". The program can additionally partition the data by "protocol type" if the number of failed login attempts exceeds a threshold that has been set. If the protocol type is TCP, the attack may be classified as a DoS attack, while if the protocol type is UDP, the attack may be classified as a probing attack.

According to [35], the Gini impureness dictated the attribute order in the decision tree for this study. The Gini impurity (GI) measures the degree of noise in the dataset's attributes. Lower Gini impurities are related to nodes further up in the decision tree hierarchy and vice versa. The mathematical form to calculate GI is:

$$Gini(q) = 1 - \sum_{l=1}^{M} q_l^2 \qquad (3.1)$$

In equation (3.1) q_l denotes probability function where an instance is labelled by class l in the NSL-KDD datasets between M classes.

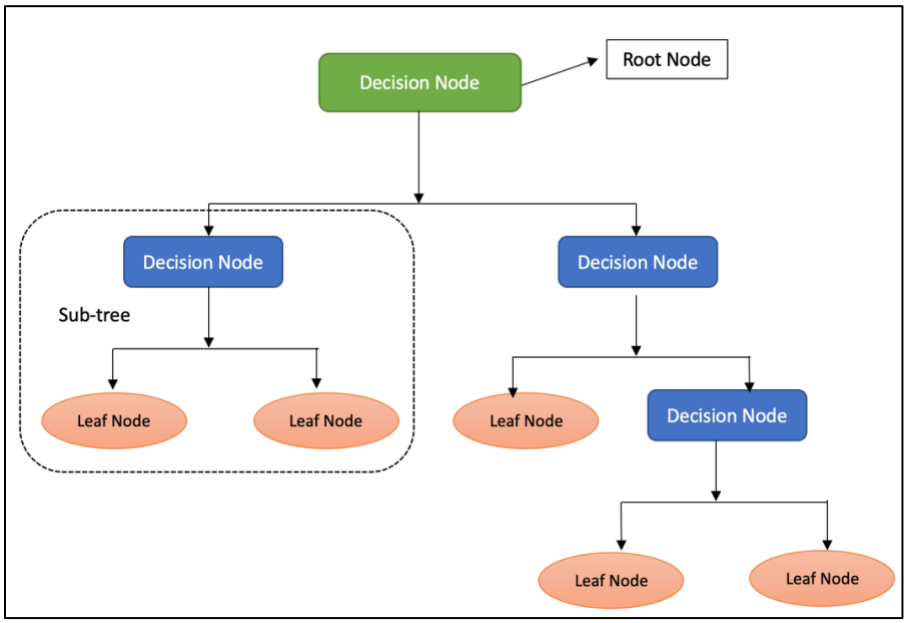


Figure 3.7 Decision tree classifier model

3.3.3 K-Nearest Neighbour (KNN):

According to [36], the K-nearest neighbour approach is the most robust algorithm in pattern recognition and data classification processes. The basic concept behind K-nearest neighbour algorithms is the distance functions or the similarity measure. It stores the state of all the instances and classifies the newly defined instances using the similarity measure. An instance relating classified dataset based upon the majority of votes with the help of neighbour classes. Both training and test datasets have their distance measure computed. In the first stage of the process, we calculate the distance between each instance by picking a value for k. Figure 3.4 shows the model of clustering techniques.

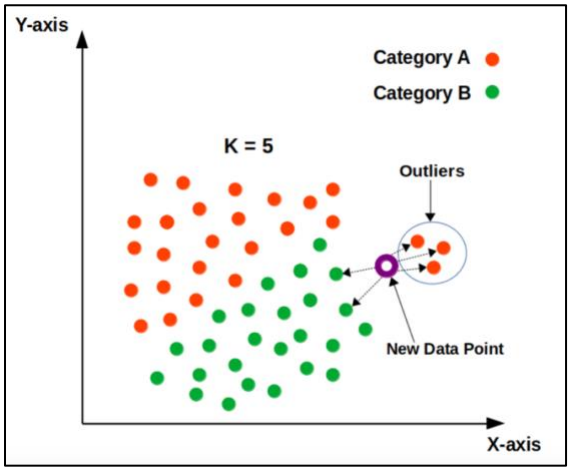


Figure 3.8 Clustering Techniques

3.3.4 Quadratic Support Vector Machine (QSVM):

The research work describes the use of Quadratic Kernel-free Non-linear Support Vector Machine (QSVM) for multi-classification. It explains that a quadratic function (W, b, c) is utilized to separate the data classes in a non-linear manner:

$$f(X) = \frac{1}{2}X^T W X + b^T X + c$$

$$W = W^T = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{12} & w_{22} & \cdots & w_{2m} \\ \vdots & \vdots & & \vdots \\ w_{1m} & w_{2m} & \cdots & w_{mm} \end{bmatrix}; \; b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}.$$

It is not required to use a dual optimization form or the Kernel technique. The results of simulations conducted on two-dimensional data sets indicate that the Quantum Support Vector Machine (QSVM) is capable of effectively separating data i.e., both linear and non-linear in nature. The decision surfaces are flexible and can take on different shapes, such as hyper-planes or hyper-spheres.

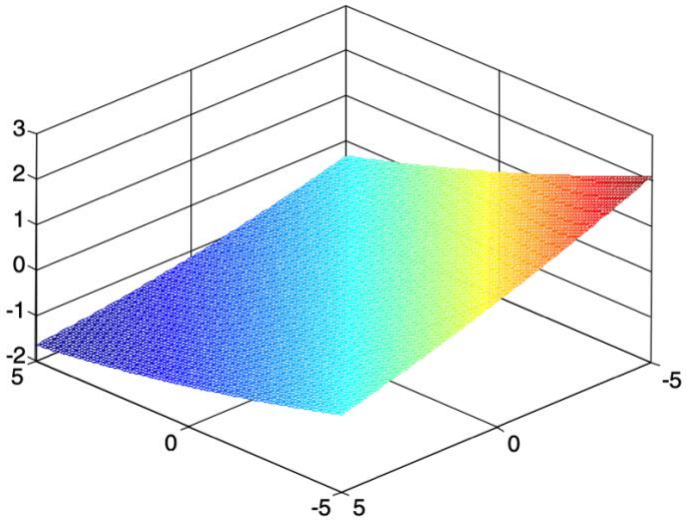


Figure 3.9 A three-dimensional plot of the quadratic function.

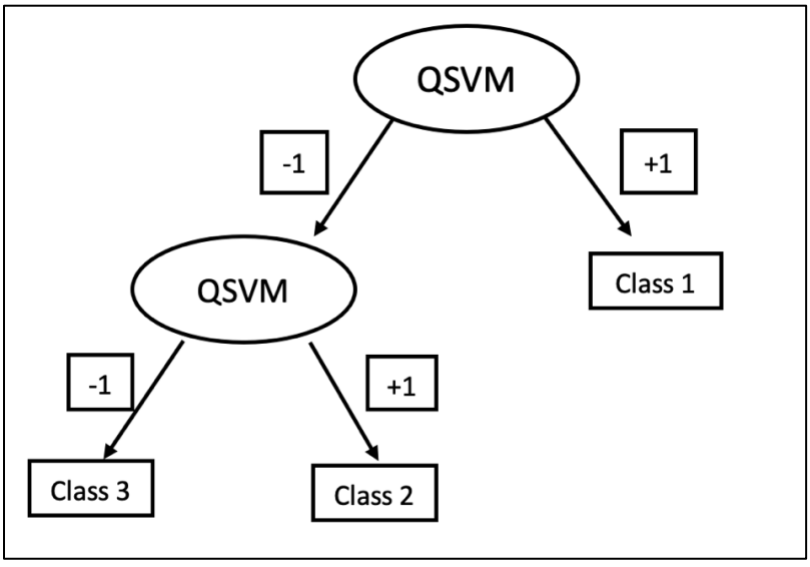


Figure 3.10 Visualization of the QSVM for multiclassification

New instances are added to the clusters based on functionality after they have been formed. The target labels of the newly generated instances determine their placement on either side of the gap

during the mapping procedure. By using a non-linear decision boundary, the classifier can capture more complex relationships between the features and the class labels, improving its accuracy. Figure 3.6 depicts the model of multiclassification using Quadratic support vector machines.

3.3.5 Autoencoders:

Autoencoders are a class of artificial neural networks that have the ability to learn insights on how to compress input data by encoding it into a lower-dimensional representation and subsequently decoding it back to its original dimensions. They are commonly used in unsupervised learning tasks, such as data compression, anomaly detection, and data denoising. Autoencoders have the potential to be utilized in multi-class classification tasks. In such tasks, the model is trained to acquire a compressed version of the input data and then utilize a classifier to predict class labels based on the encoded features. This method can be beneficial when working with high-dimensional data, where it may be challenging to classify the data without first extracting meaningful features.

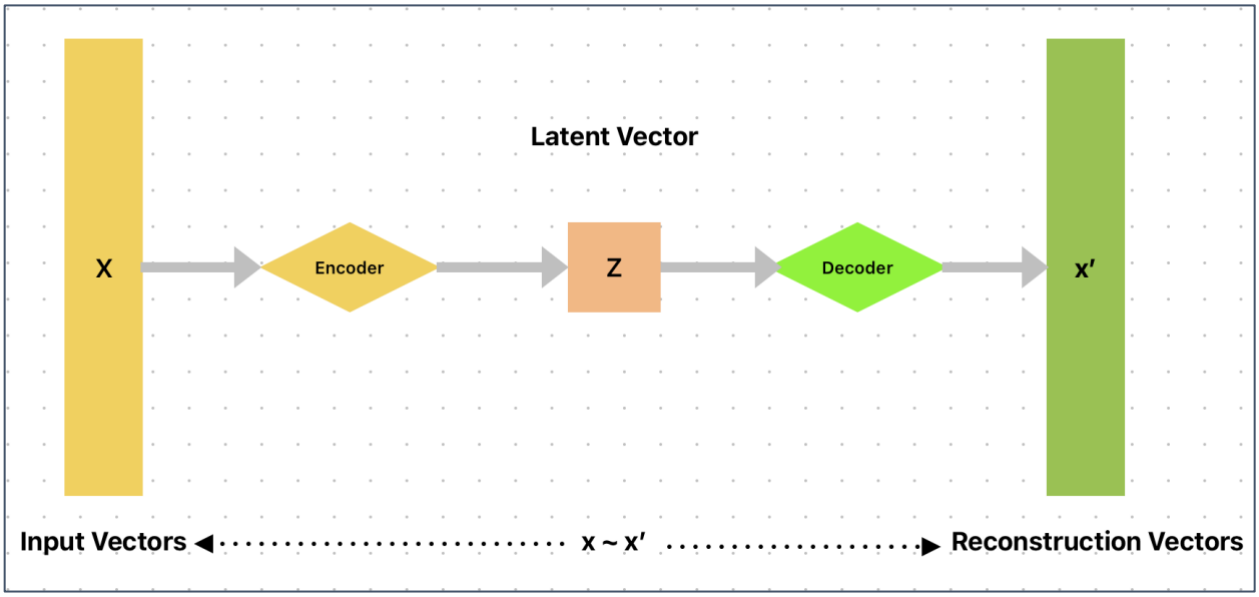


Figure 3.11 Autoencoder model Architecture

During the training process of an autoencoder-based multi-class classifier, the model learns to reconstruct the input data using compressed features, while simultaneously learning to classify attacks based on these features. The model is optimized to minimize the loss function, which combines the reconstruction and classification errors. The dissimilarity between the manner in which input and output are represented is commonly referred to as the reconstruction error, which

denotes the discrepancy between the input vector and the output vector. Anomaly detection is a suitable application of the Autoencoder.

3.3.6 MultiLayerPreceptron Algorithm:

The classification of network traffic as normal or malicious using the NSS-KDD dataset can be achieved through the utilization of the Multi-Layer Perceptron (MLP) algorithm. This algorithm uses a neural network that consists of multiple layers of interconnected nodes. The neural network's input layer is responsible for receiving the network traffic's features as inputs. These features may include protocol type, service, source and destination IP addresses, among others. The input data is processed by the hidden layer(s) of the neural network through non-linear transformations. The neural network's output layer generates a binary classification outcome that denotes whether the network traffic is normal or attack. The Multilayer Perceptron (MLP) algorithm employs the backpropagation technique to facilitate the training of the neural network. This involves modifying the weights between the nodes to reduce the discrepancy between the anticipated and factual classification outcomes on the training dataset. The goal is to reduce the variance between the expected and actual attack classifications.

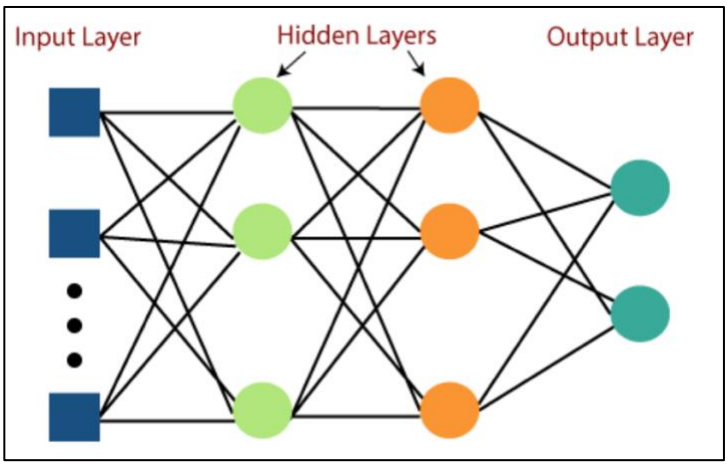


Figure 3.12 Neural network model

The present investigation involved the utilization of NSL-KDD datasets for training purposes, incorporating a total of three hidden layers. The 1st and 3rd hidden levels contained 15 nodes, whereas the second layer had 100. The activation function for all levels (excluding the

output layer) was ReLu. Because the job requires categorical categorization, the softmax function was employed to activate the output layer.

3.3.7 Multi-classification using Convolutional Neural Network (CNN):

The Convolutional Neural Network (CNN) is a widely used deep learning framework that exhibits notable effectiveness in tasks related to image recognition and processing. Convolutional Neural Networks (CNNs) possess a unique capability of learning hierarchical feature representations from unprocessed input images, unlike other classification models. This feature can significantly reduce the pre-processing time. The aforementioned task is achieved by utilizing convolutional layers, which employ filters to identify and extract local patterns present in the images. Convolutional neural networks (CNNs) are capable of recognizing and classifying images based on their distinctive features by assigning varying degrees of significance to different objects and characteristics present within the images.

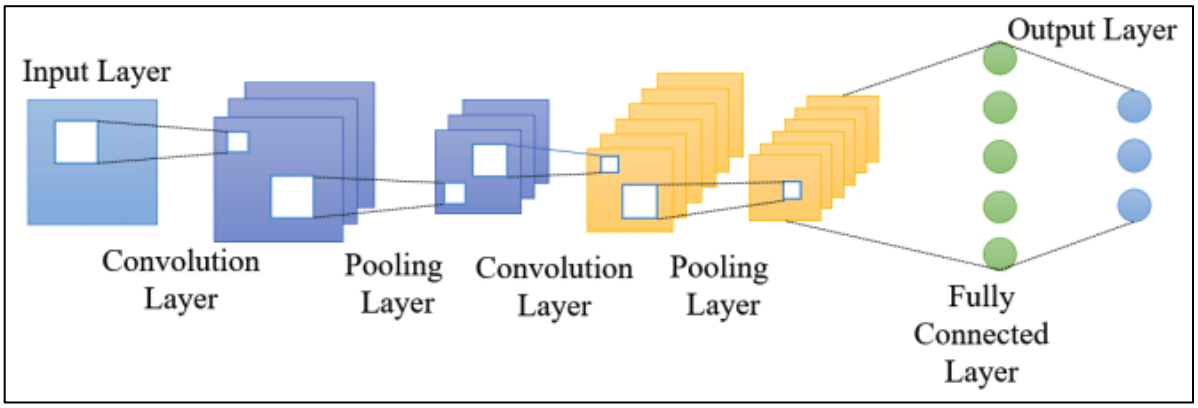


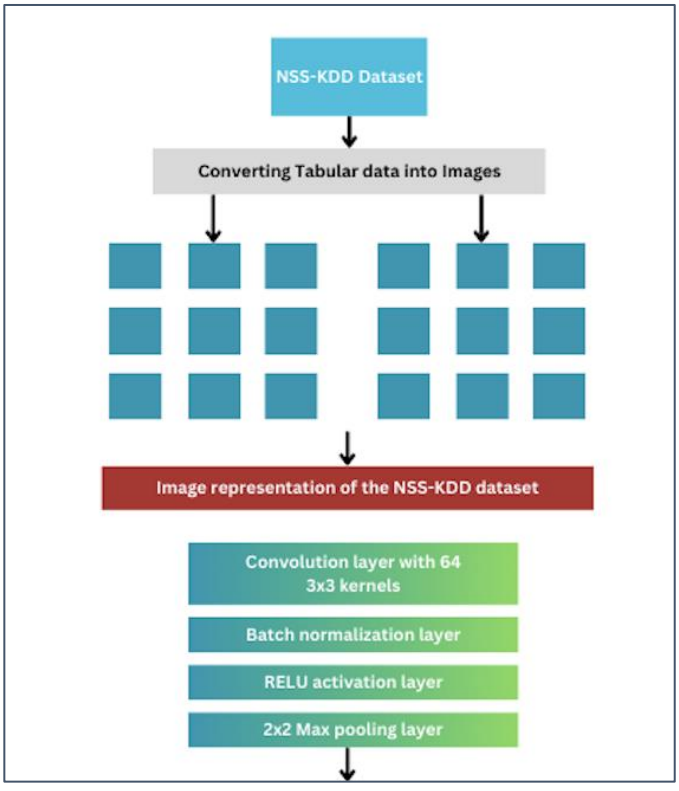
Figure 3.13 Convolutional Neural Network Architecture

Figure 3.8 shows the primary layers within a conventional Convolutional Neural Network (CNN). The first one is the Convolutional Layer which is responsible for applying filters to the input image in order to extract various features, including but not limited to edges, shapes, and textures. The feature maps generated by the filters illustrate unique attributes that are intrinsic to the image. The Pooling Layer constitutes the next layer in the neural network architecture, wherein the feature maps generated by the Convolutional Layer go through a downsampling operation with the purpose of decreasing their dimensionality. This helps in preventing overfitting. The ReLU layer is a crucial component of CNNs that applies the Rectified Linear Unit (ReLU) activation function to the output of the previous layer. By introducing non-linearity to the CNN, the ReLU layer facilitates the extraction of more complex and informative features from input images.

The penultimate layer is the Fully Connected Layer that enables the connection of all neurons from the previous layer to all neurons in the next layer, producing the final output of the CNN. This output is then used for classification purposes, allowing the CNN to classify the input image into different categories based on the features extracted from the previous layers. Additionally, the Softmax Layer transforms the output of the fully connected layer into a set of probabilities for each class.

3.3.8 Proposed Model Architecture:

The utilization of Convolutional Neural Networks (CNNs) has proven to be efficacious in numerous applications where significant data insights are embedded in the feature sequence, such as speech and imaging. Tabular data is typically not suitable for Convolutional Neural Network (CNN) based modelling as it does not have any spatial correlation among its features. In order to tackle this issue, we propose a new model that transforms tabular data into visual representations through a process of feature-to-pixel mapping. This allocation is designed to ensure that comparable features are grouped together within the resulting image. In certain scenarios, it is feasible to reorganize the arrangement of features within a two-dimensional space to more effectively depict the associations between said features, such as similarities or categories.



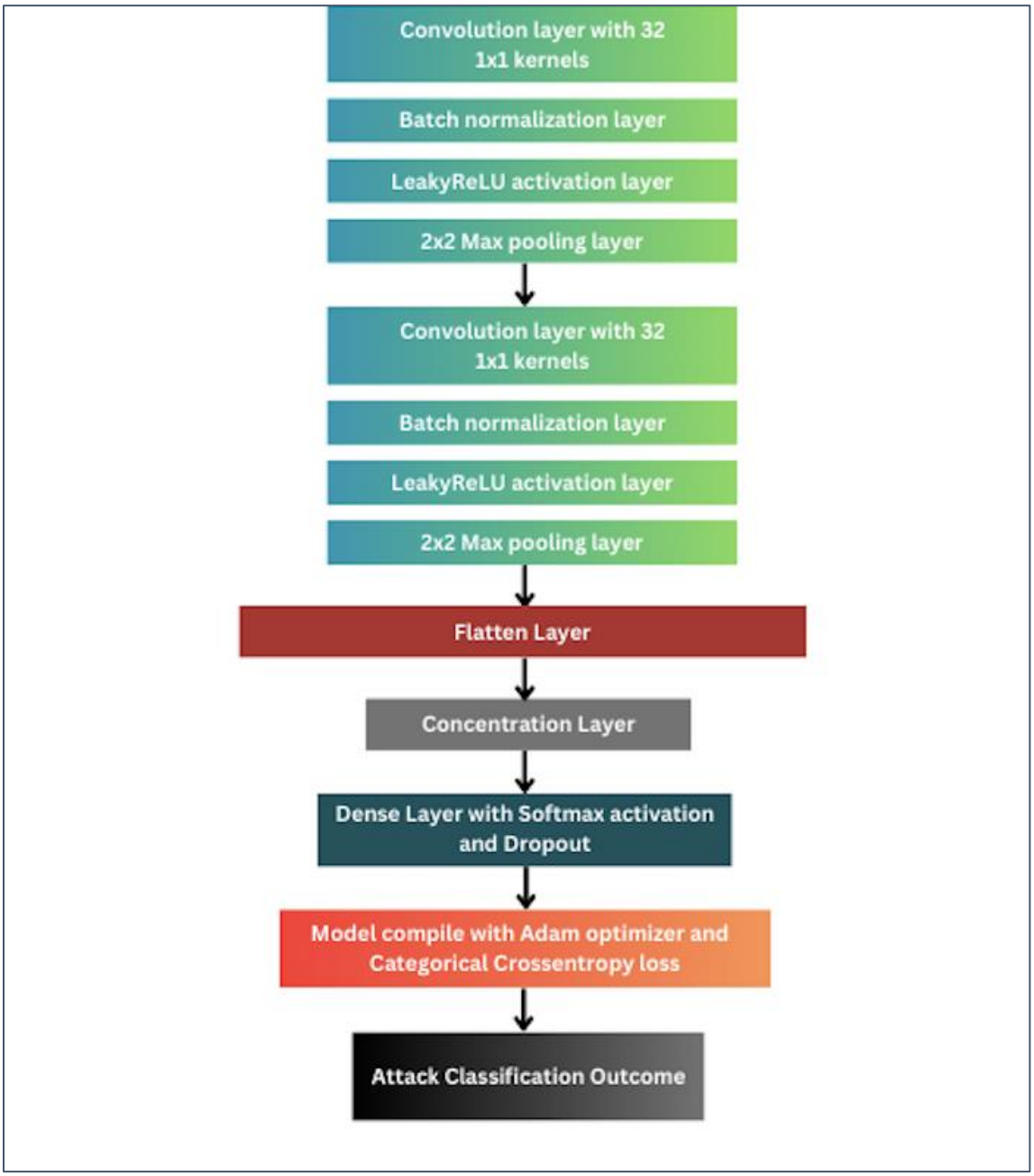


Figure 3.14 Proposed Model Architecture

Our proposed model for converting tabular data into images has several benefits. One such significant advantage of using CNNs is their ability to work with images, which can be viewed as 2D arrays of pixels. When tabular data is transformed into images, CNNs can be employed for feature extraction, allowing them to identify crucial patterns and features within the image to make

predictions about the target variable. This approach can result in decreased data dimensionality and enhanced model performance.

Another advantage of converting to images is the ability to use data augmentation techniques to expand the training set and enhance the performance of the model. By applying transformations like rotation, scaling, and cropping to the image, we can generate new images that retain the essential characteristics of the original data while introducing variations. These augmented images can be employed in training the CNNs, which can help prevent overfitting and improve the model's ability to generalize. Furthermore, CNNs are well-suited for capturing non-linear relationships in the data, which is often the case with tabular data. By converting tabular data into images, we can effectively represent these non-linear relationships and enable the CNNs to learn them. This can lead to improved model performance and the ability to capture complex patterns in the data. Also in view of the Visualization, the converting the linear data into images can help us better in visualizing the patterns and features detected by the CNNs, which can aid in model interpretation and exploratory data analysis.

Tuning the hyper parameters of the algorithm:

The model parameter is the configuration set to default by the program, and the value can be seen from the dataset. This parameter can be learned from the model developed by the algorithm in machine learning. The model generated shows a good skill than it was because of the parameters. Once the model is generated, the parameter is saved with the model. The historical data are also used to learn efficient parameters by creating a model.

Hyper parameter:

The hyperparameter is a configuration that is changed from the model and the values the dataset cannot generate. The usage of hyperparameters can find the estimation of parameters. The researcher or programmer could specify this. The best values cannot be found easily and could be created with a trial and error method. Only through self-discovery the hyperparameters for the given problem can be identified. The hyperparameters are tuned mostly for the predictive modelling problem. In this proposed framework, hyperparameter tuning is used for the practical purpose of efficient model generation.

MLP classifiers have some important parameters like

- Solver uses 'Adam' default one because it handles the large dataset very efficiently.
- random_state is set to '1' to produce the same results across different calls.

Random Forest Classifier important parameters that are tuned are:

- n_estimators=10 will potentially decrease the model's building time and the number of trees in the ensemble.
- max_features='auto' will sqrt the features to get the good default values.
- min_samples_split, max_leaf_nodes, max_depth, and min_samples_leaf tuning these parameters will reduce the model's size and produce optimal results.

After tuning the model's hyperparameters, it is checked with the Cross-validation 10-fold method by training the model with the training data. This will show how the hyperparameter works on the model.

Performance metrics

Various metrics are employed to characterize the effectiveness of a classifier. Table 3.4 presents a summary of the four prospective outcomes of detection.

Table 3.4: Confusion Matrix

	Predicted		
Actual		Normal	Attacks
	Normal	TN	FP
	Attacks	FN	TP

Recall refers to the ratio of accurately identified attacks to the total number of attacks.

$$Recall = \frac{TP}{TP + FN} \qquad (3.2)$$

Accuracy is a metric used to measure the proportion of correctly identified outcomes in both normal and attack traffic. In multiclass classification, accuracy is determined by dividing the size of the intersection of the label sets by the size of their union.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (3.3)$$

Precision refers to the ratio of correctly identified attacks to the total number of identified attacks.

$$Precision = \frac{TP}{TP + FP} \qquad (3.4)$$

The F1-score is a statistical measure that represents the harmonic mean of precision and recall. (previously called “detection rate”).

$$F1 - score = \frac{precision \times recall}{precision + recall} \qquad (3.5)$$

Summary:

This chapter explains the implementation of the proposed study, and it also presents the results of the classifiers. The dissertation predicts the output of the six classifiers in different views to establish an efficient result. The result is highly improved than the existing methods. Then the cloud data also predicts efficient outcomes, showing that the proposed model accurately predicts attacks. The next chapter checked the hyperparameter tuning to determine whether they fit the model exactly using the cross-validation method. This is further continued in the next chapter for a clear view of the proposed cloud Anomaly IDS.

Chapter 4

RESULTS AND DISCUSSION

The objective of this project is to create a classifier model for anomaly detection using machine learning and neural networks, specifically on the NSLKDD+ dataset. Additionally, classification model addresses the challenges associated with anomaly detection through multi-class classification. In Chapter 4, six distinct machine learning models were trained and subsequently analyzed for each classification methodology. The hardware specs for this investigation include 8 GB of random access memory (RAM), a 2.5 GHz central processing unit (CPU), a 630 MHz graphics card, with an open source Jupyter environment within the Anaconda distribution.

Metrics from Chapter 3 are evaluated when deciding whether or not the proposed remedy can effectively solve the current problem of resource attacks on virtual machines in cloud servers. The parameters reviewed here include accuracy, precision, recall and f1-score. Furthermore, an analysis of model performance is conducted through making use of a learning curve, scalability curve, mean squared error curve, and model loss curve derived from the training history. Ultimately, a comparison is made between the mean absolute error, median absolute error, mean squared error and overall performance of the present study and those reported in prior research.

4.1 EXPERIMENTS:

4.1.1 Quadratic Support Vector Machines (QSVM)

The 'SVC' function is utilized to instantiate an QSVM classifier, whereby the parameter 'kernel' is set to 'poly' to indicate the use of a polynomial kernel. The 'gamma' parameter is also specified and set to 'auto', thereby enabling the kernel coefficient to be automatically determined.

	precision	recall	f1-score	support
Dos	0.96	0.94	0.95	11484
Probe	0.96	0.61	0.74	2947
R2L	0.00	0.00	0.00	274
U2R	0.00	0.00	0.00	15
normal	0.91	1.00	0.95	16774
accuracy			0.93	31494
macro avg	0.56	0.51	0.53	31494
weighted avg	0.92	0.93	0.92	31494

Figure 4.1 Classification report for QSVM Classifier

The evaluation of the model's performance involved examining the precision, recall, f1-score, and support metrics for each class label. The outcomes demonstrate that the model displayed excellent precision and recall values for the 'Dos' and 'Normal' categories, while the 'Probe', 'R2L', and 'U2R' labels were associated with relatively lower values. The model's overall accuracy was 92%, indicating that it predicted the correct class label for 92% of the instances in the test dataset. However, the model's ability to predict the 'R2L' and 'U2R' class labels was weak, as shown by the f1-score and recall values of 0.00. Consequently, further improvements may be required to enhance the model's ability to predict these two class labels. Fig 4.2 shows plotting a line chart with two lines, one for the predicted attacks and one for the actual attack classifications. The colour blue is used for the predicted attacks line, while light coral is used for the actual attack classification line.

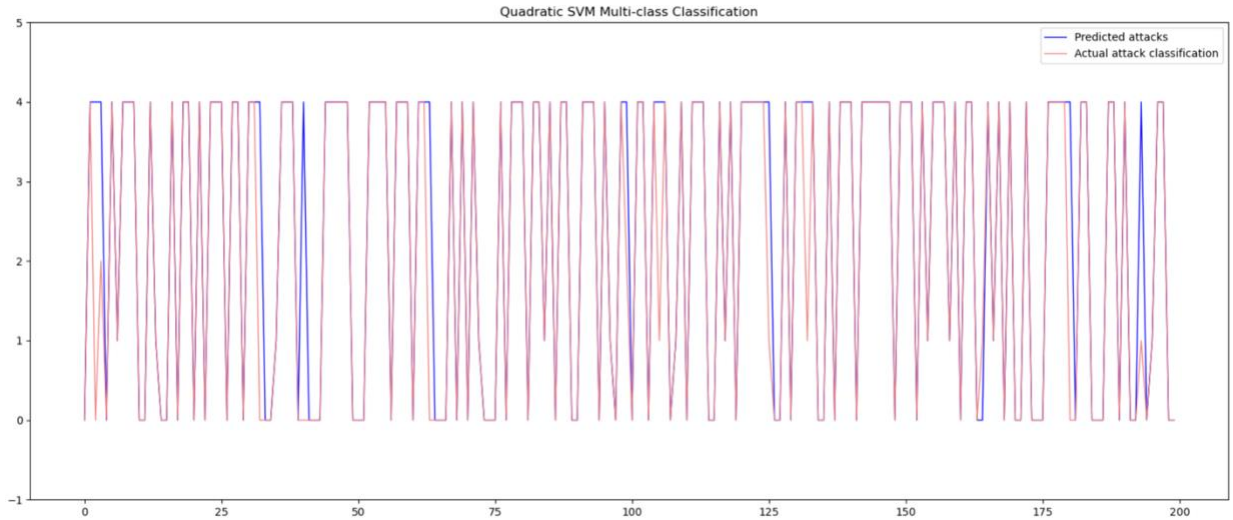


Figure 4.2 QSVM: Predicted Vs Actual attack

4.1.2 Decision Trees and KNN Classifiers

We used the DecisionTreeClassifier class from Scikit-learn and setting the criterion parameter to gini and the max_depth parameter to 5. This means that the decision tree will split the data based on the gini impurity index and the maximum depth of the tree will be limited to 5 levels. The model achieved an overall accuracy of 81%, indicating that it is relatively effective in identifying network attacks. The model's accuracy in identifying normal traffic was high at 94.6%, suggesting that it is good at recognizing normal network activity. The accuracy for identifying DoS and Probe attacks was also relatively high at 86.4% and 88.5%, respectively. However, the

accuracy for identifying R2L attacks was low at 41.9%, indicating that the model may require further improvement in identifying this type of attack. Additionally, the accuracy for identifying U2R attacks was 0%, which suggests that the model was not able to detect any instances of this attack type.

Subsequently we performed k-nearest neighbours algorithm on the NSL-KDD dataset, where KNN would involve identifying the k nearest neighbours of a new data point based on their distance, and the class label of the new point was then assigned based on the majority class label of those k neighbours. We tuned the hyperparameter k which is classifier object with a value of 5. The model attained an overall accuracy rate of 77.5%. The accuracy of identifying normal traffic was relatively high at 96.7%, while the accuracy for DoS and Probe attacks was moderate at 82.8% and 71.8% respectively. The model exhibited a low level of accuracy, specifically 6.1%, in its ability to correctly identify R2L attacks. Furthermore, it was unable to successfully detect any U2R attacks. The findings indicate that there is a possibility that the model may require additional enhancements to more accurately detect complicated attacks such as R2L and U2R.

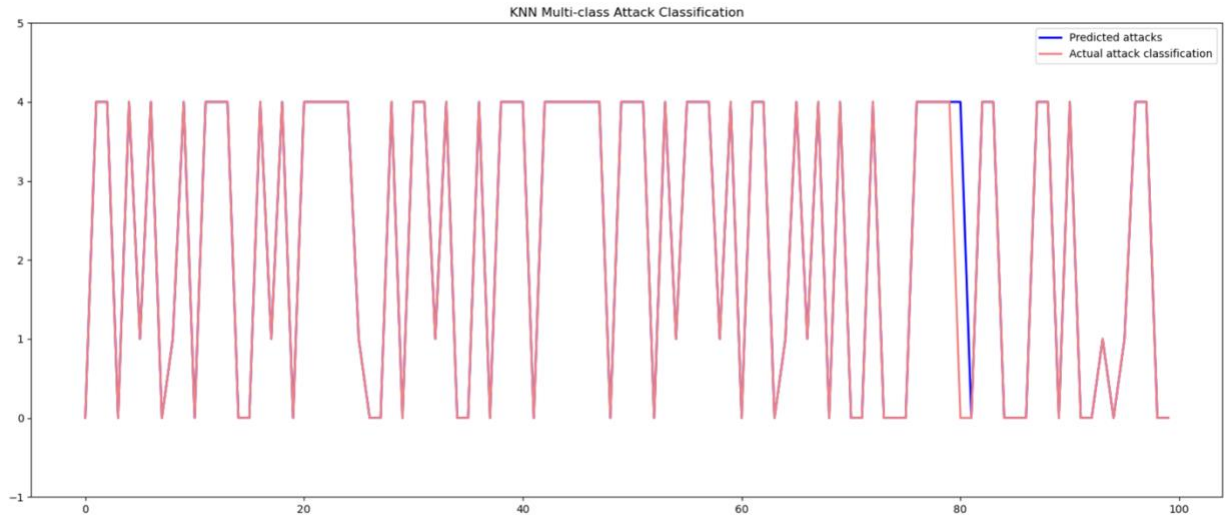


Figure 4.3 KNN: Predicted Vs Actual attack

The visual representation presented in Figure 4.3 illustrates the process of generating a line graph that comprises of two separate lines. One line denotes the projected assaults, whereas the other line denotes the factual categorizations of attacks. The expected attack line is denoted by the colour blue, while the actual attack categorization line is illustrated by the other shade. As you can see although there is slight deviation from the actual line, the model tends to overfit. While the

"normal" label was correctly classified by both the Decision Tree and KNN classifiers, their performance in classifying the U2R and R2L attacks was found to be lacking.

4.1.3 Random Forest Classifier

The model initializes a Random Forest Classifier model using the scikit-learn library in Python and is defined with 10 trees (n_estimators) and the criterion of splitting is set to entropy.

	precision	recall	f1-score
Normal	0.75	0.97	0.85
DoS	0.89	0.83	0.86
Probe	0.59	0.72	0.65
R2L	0.96	0.06	0.11
U2R	0.21	0.02	0.14
accuracy			0.78
macro avg	0.6	0.52	0.52
weighted avg	0.8	0.78	0.73

Figure 4.4 RFC Classification Report

The table displays the precision, recall, and F1-score metrics for each attack type, along with the overall accuracy and macro and weighted average scores. The findings indicate that the model exhibits excellent accuracy in identifying Normal and DoS attacks, as proven by higher precision, recall, and F1-score metrics. However, the model's performance in detecting Probe, R2L, and U2R attacks is relatively lower, as shown by lower F1-score values. The classification of each attack is interpreted as follows; Accuracy of normal = 97.2 %;Accuracy of DoS = 84.0 %;Accuracy of Probe = 73.4 % Accuracy of R2L = 6.59 %. But Overall, the model achieves a weighted average F1-score of 0.78, indicating that it has room for improvement in detecting certain types of network attacks.

4.1.4 Autoencoder (AE Classifier)

The architecture of the Autoencoder includes an input layer that is shaped according to the input data, followed by two fully connected layers. The first fully connected layer consists of 48 neurons and employs the sigmoid activation function, while the second fully connected layer comprises 24 neurons and uses the hyperbolic tangent activation function.. The output layer with

5 neurons and sigmoid activation function is also added. The model has been compiled utilizing the Adam optimizer, mean squared error loss function, and accuracy metric.

In this case, the fit method is used to train the Autoencoder model on the input data predictions and target data y_test. The epochs parameter is set to 200, specifying the number of times the model will be trained on the entire dataset. The batch_size parameter is set to 700, which is the number of samples that the model will process at once before updating the parameters. Finally, the validation_split parameter is set to 0.2, i.e., 20% of the input data will be used for validation during the training process.

The results of the test show that the Autoencoder model has achieved a loss value of 0.064 and an accuracy of 86.96%. A lower loss value signifies that the model is proficient in minimizing the variance between the predicted and actual output. Conversely, an increased accuracy value indicates that the model is proficient in accurately categorizing the input data. The model comparatively did better but other models could outperform this. Figure 4.5 and 4.6 represent the Accuracy and Loss with respect to Epoch for the KDDTrain and test datasets. As the epochs increase, accuracy increases and it saturates for any further increase in epoch. Similarly, the loss value decreases and saturates over a threshold value for any further increase in epoch.

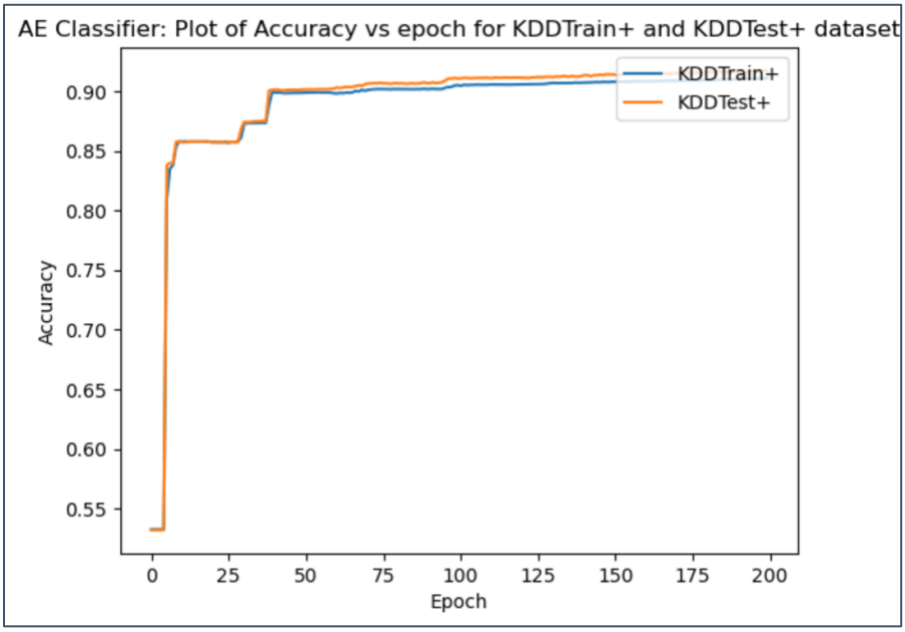


Figure 4.5 AE Classifier: Plot of Accuracy with respect to Epoch

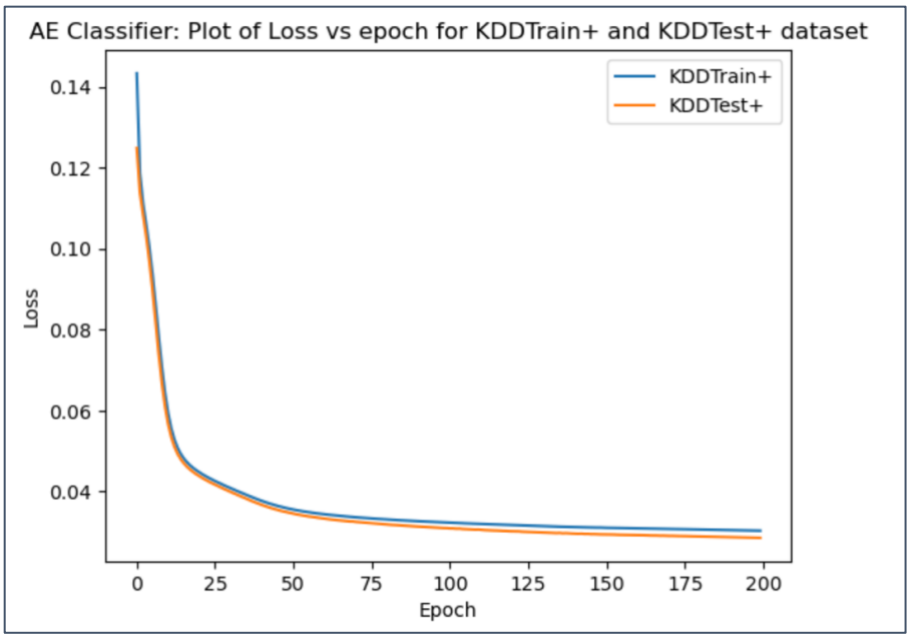


Figure 4.6 AE Classifier: Plot of Loss with respect to Epoch

4.1.5 MultiLayerPerceptron (MLP) Classifier

The MLP model initializes a sequential model that contains two layers: an input layer and the first hidden layer with 50 neurons and a ReLU activation function, followed by an output layer with 5 neurons and a softmax activation function. We evaluate this multi-layer perceptron (MLP) model using categorical cross-entropy loss as the loss function, the Adam optimizer, and accuracy as the performance metric. Based on the test results, the MLP model demonstrated excellent performance in classifying network intrusion attacks. The model achieved a low loss value of 0.0898, indicating that the predicted class probabilities were similar to the true class probabilities. Additionally, the high accuracy of 96.95% on the test dataset suggests that the model was able to correctly predict the class labels for a large proportion of the test instances, demonstrating the model's effectiveness.

Overall, the test results provide evidence that the MLP model is a reliable approach for intrusion detection. Figures 4.7 and 4.8 depict the accuracy and loss values in relation to epoch for the KDDTrain and test datasets. As the number of epochs increases, the level of accuracy also increases until it reaches a point of saturation beyond which further increases in epoch do not lead to any significant improvement in accuracy. Likewise, the value of loss exhibits a decline and

reaches a saturation point beyond which any subsequent increase in epoch does not lead to a further decrease in the loss value.

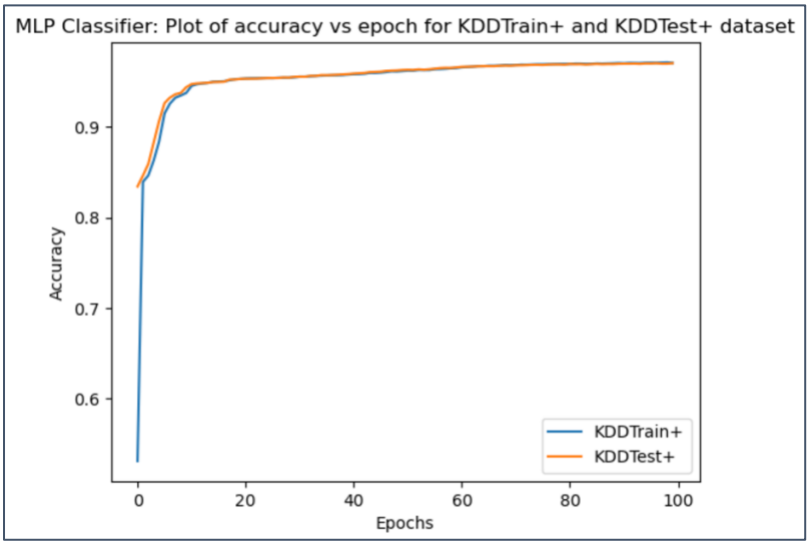


Figure 4.7 MLP Classifier: Plot of Accuracy with respect to Epoch

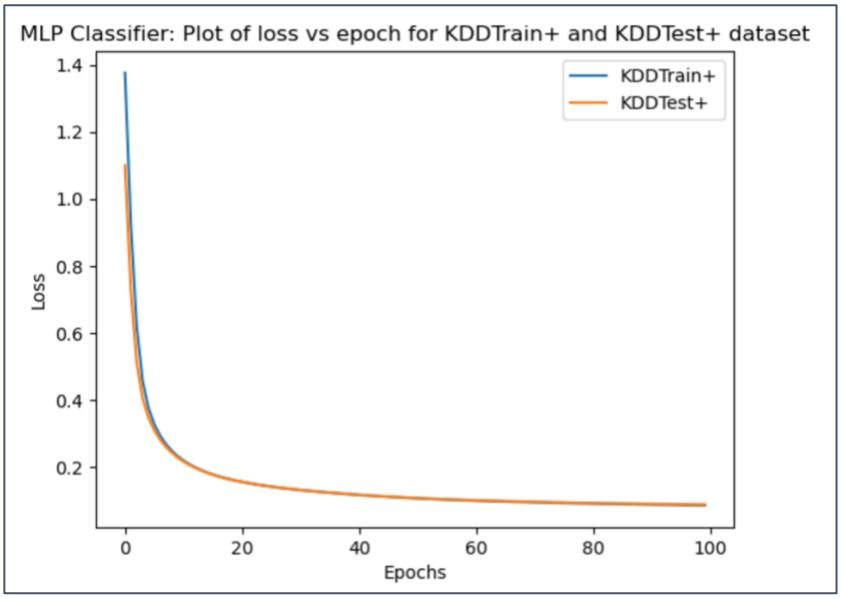


Figure 4.8 MLP Classifier: Plot of Loss with respect to Epoch

Additionally, The MLP model achieved high scores for recall, precision, and F1, indicating that it performed well in predicting the correct class labels for the test dataset. Specifically, the model achieved a recall score of 0.967, indicating that it correctly identified 96.7% of the instances for the target class. The F1 score takes in to account both precision and recall, was found to be 0.97. The model also exhibited high precision, with a score of 0.972, which indicates that it

predicted an instance to belong to the target class, i.e., with either normal to a normal class and attack to the corresponding attack class, we can imply that it was correct 97.2% of the time.

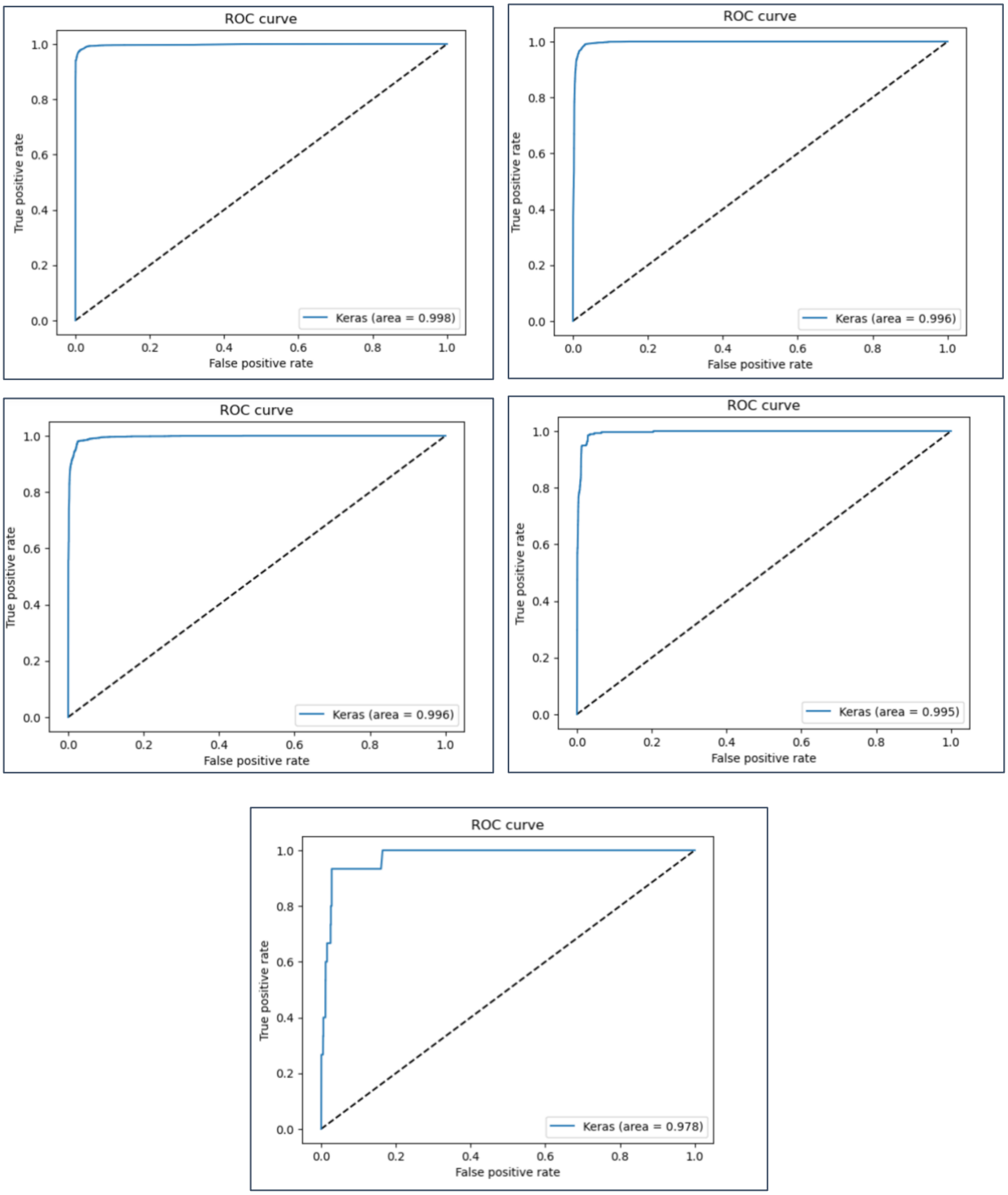


Figure 4.9 MLP Classifier: ROC Curves with corresponding Area values for each class.

Fig 4.9 represents the ROC curves for each of the classes, The ROC areas represent the model's ability to distinguish between positive and negative classes. A higher ROC area indicates better predictive accuracy. In this case, the ROC areas for normal, DOS, probe, and U2R attacks are very high (0.998, 0.997, 0.996, and 0.996), indicating that the model is highly accurate in distinguishing these attacks from normal network traffic. However, the ROC area for R2L attacks is comparatively lower at 0.97, suggesting that the model may not be as effective at detecting this specific type of attack. Overall, the high ROC areas for most attack types suggest that the model is effective in identifying most network attacks.

4.1.6 Proposed Model:

Convolutional Neural Networks (CNN) using Conversion of tabular data into images: The concept involves transforming two-dimensional data into images, which can be leveraged by Convolutional Neural Networks (CNNs) to improve the accuracy of predictions relative to models trained on 2-D data. The transformation process entails the conversion of discrete data points in a table format into visual representations, where the features and their corresponding numerical values are represented by pixels and their respective dimensions. The images in the proposed model for tabular data have a common pixel or set of pixels representing a feature in all samples, with variations in pixel intensities across images.

4.1.6.1 Feature transformation:

In our dataset, we have certain features such as *num_outbound_cmds*: represents the number of outbound commands in an FTP session, this feature is always zero, which suggests that it may not be a useful feature for detecting network attacks. There are couple of other parameter such as *num_failed_logins*, *num_file_creations* and *num_shells* which tend to have more zeroes. Additionally, upon conducting one-hot encoding, the data is often populated with binary values, thereby increasing the number of columns.

In our proposed model we tend to rearrange the highly correlated features towards the centre of the tabular data. One advantage of rearranging significant features towards the centre columns of a dataset is that it can make it easier for the CNN to learn spatial relationships between the features, which can be important for certain image-related tasks such as object recognition or segmentation. Also, by placing significant features towards the centre of the image, the CNN can

effectively scan the input with a receptive field that covers a larger portion of the significant features, which can help it identify patterns and relationships between them more easily.

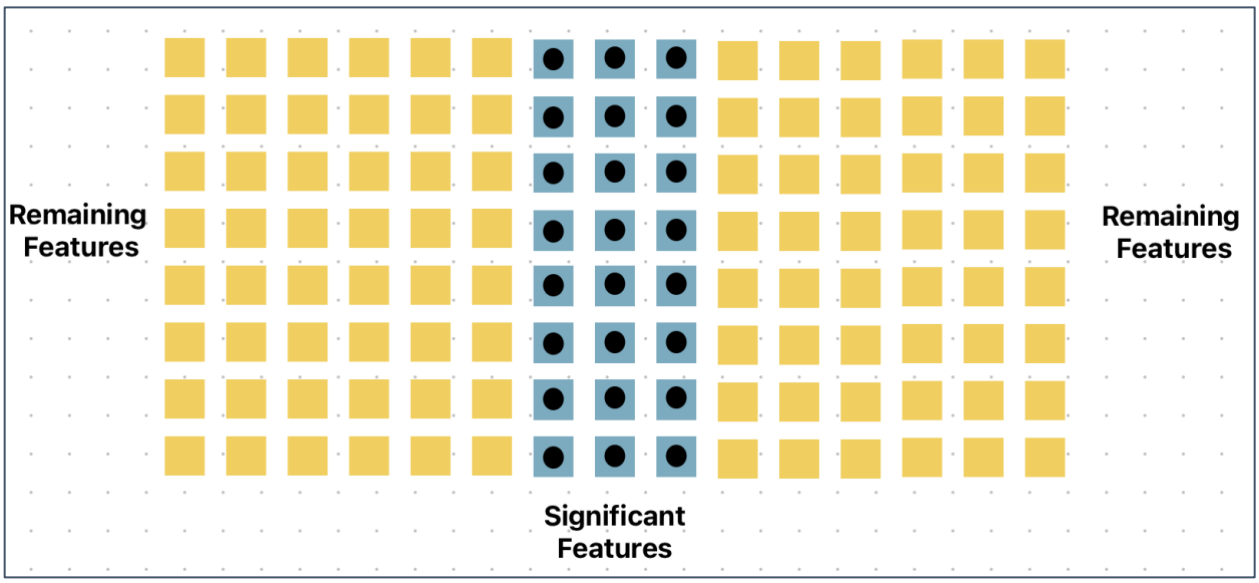


Figure 4.10 Central rearrangement of the input features.

The architecture of the CNN model is developed with three convolutional layers and one dense layer. Upon feature selection and one hot encoding the we have total of 90 features that needs to the trained. The first layer, input layer specifies the shape of the input data that will be fed into the network for a neural network with. The first convolutional layer applies 64 filters with a kernel size of 3x3. The output of this layer is then passed through max pooling with a pool size of 2x2. The second convolutional layer applies 32 filters with a kernel size of 3x3 and then passes through dropout regularization. The output is then passed through max pooling with a pool size of 2x2. The third convolutional layer applies 16 filters with a kernel size of 3x3. The output is then flattened to a 1D vector and passed through a dense layer with 5 neurons. The model consists of three convolution layers with 3x3 kernels, followed by batch normalization, ReLU and LeakyReLU activation, and maximum pooling layers. The subnetwork processes the input image representations and outputs feature maps. These feature maps are then flattened and concatenated before being fed into a fully connected network.

```
Model: "sequential_34"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_60 (Conv2D)          (None, 8, 7, 64)          640

 max_pooling2d_45 (MaxPoolin  (None, 4, 3, 64)         0
 g2D)

 dropout_22 (Dropout)        (None, 4, 3, 64)          0

 conv2d_61 (Conv2D)          (None, 4, 3, 32)          2080

 dropout_23 (Dropout)        (None, 4, 3, 32)          0

 max_pooling2d_46 (MaxPoolin  (None, 2, 1, 32)         0
 g2D)

 conv2d_62 (Conv2D)          (None, 2, 1, 16)          528

 flatten_33 (Flatten)        (None, 32)                0

 dense_141 (Dense)           (None, 5)                 165

=================================================================
```

Figure 4.11 CNN Model Summary.

```
Epoch 95/100
16/16 [==============================] - 0s 4ms/step - loss: 0.0885 - accuracy: 0.9700 - val_loss: 0.0906 - val_accur
acy: 0.9688
Epoch 96/100
16/16 [==============================] - 0s 4ms/step - loss: 0.0883 - accuracy: 0.9703 - val_loss: 0.0905 - val_accur
acy: 0.9693
Epoch 97/100
16/16 [==============================] - 0s 4ms/step - loss: 0.0881 - accuracy: 0.9705 - val_loss: 0.0900 - val_accur
acy: 0.9689
Epoch 98/100
16/16 [==============================] - 0s 4ms/step - loss: 0.0878 - accuracy: 0.9703 - val_loss: 0.0900 - val_accur
acy: 0.9691
Epoch 99/100
16/16 [==============================] - 0s 4ms/step - loss: 0.0875 - accuracy: 0.9706 - val_loss: 0.0896 - val_accur
acy: 0.9691
Epoch 100/100
16/16 [==============================] - 0s 6ms/step - loss: 0.0872 - accuracy: 0.9702 - val_loss: 0.0897 - val_accur
acy: 0.9690
```

In our study, we conducted a comparative analysis between the predictive performance of Convolutional Neural Networks (CNNs) trained on various image representations and that of alternative prediction models trained on the raw data in a 2-D format. As per figure 4.12, we can see that there is very slight deviation of the predicted attack vs the actual attack achieving an accuracy of 0.97. The results showed that CNNs had the highest average prediction performance across cross-validation.

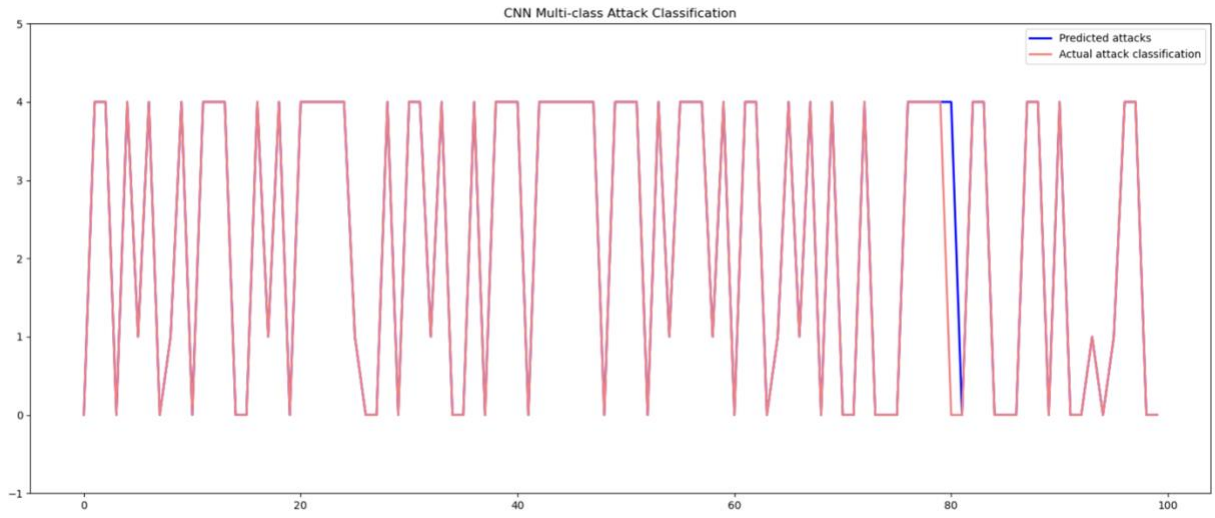


Figure 4.12 CNN Model – Actual Vs Predicted Values plot

4.2 Feature Selection accuracy results

The process of feature selection resulted in a reduction in the quantity of input variables utilized for building a predictive model. This has the dual impact of lowering the time complexity and boosting the overall classifier robustness. Some useful features are selected depending on the ranking score. The feature with the strongest correlation will be considered significant, while those with a more negligible correlation will be considered insignificant. Table 4.1 depicts the accuracy for all five models after applying FS with RFC, QSVM, DT, KNN, MLP, Autoencoder and our proposed model.

Table 4.1: Performance comparison of the algorithms (with RFC, QSVM, DT, KNN and Autoencoder, MLP and proposed model).

Method	Accuracy	Precision	Recall	F1=score
RFC	0.84	0.86	0.84	0.84
Quadratic SVM	0.92	0.92	0.93	0.92
DT	0.81	0.83	0.81	0.78
KNN	0.78	0.80	0.78	0.73
Proposed Model	0.97	0.97	0.96	0.96
Autoencoder	0.86	0.77	0.89	0.83
MLP	0.96	0.97	0.96	0.95

RFC has an accuracy of 0.84, corresponding to 84%; Quadratic SVM has an accuracy of 0.92, related to 92%; DT has an accuracy of 0.81, corresponding to 81%; KNN has an accuracy of 0.78, corresponding to 78%; and MLP has an accuracy of 0.96, related to 96%; and Autoencoder has an accuracy of 0.86, corresponding to 86%;. As a result, the MLP method is particularly efficient compared to the other algorithms on performance parameters. The MLP method costs around 5% and 10% less than the different algorithms, as shown in Figure 4.13. In light of this, the proposed algorithm outperforms the prior methods regarding feature selection.

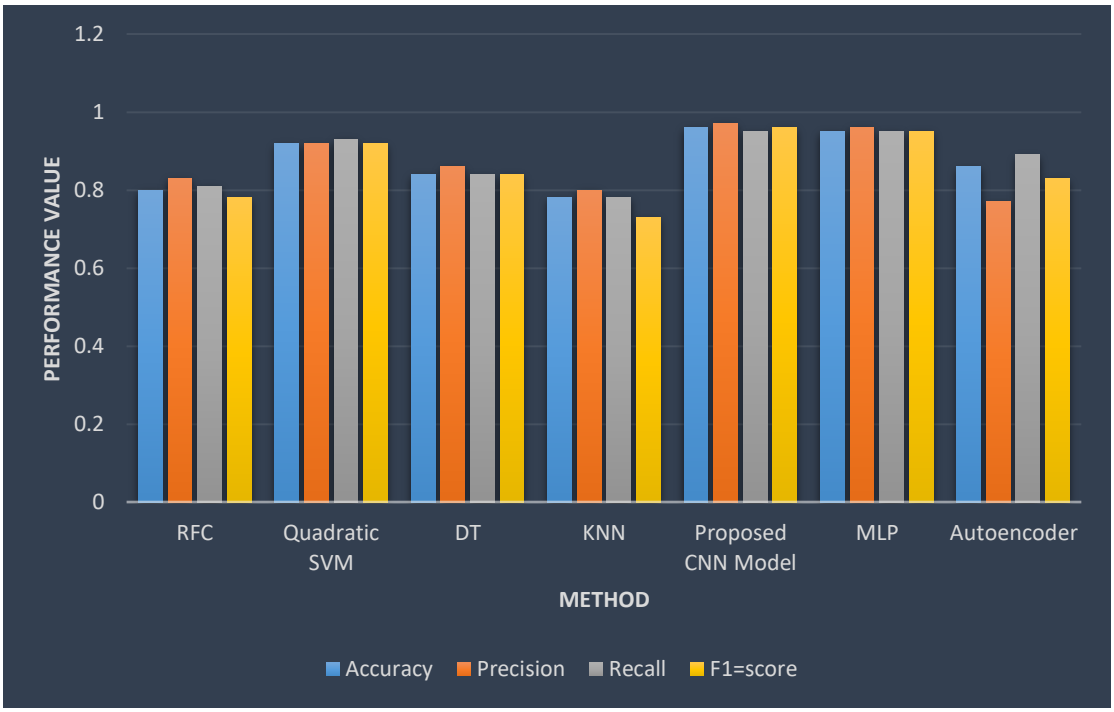


Figure 4.13 Performance comparison of feature selection with RFC, Quadratic SVM, DT, KNN, MLP and Autoencoder

Method comparisons

Section 4.3 contains tables that compare the suggested technique to recent studies that used the NSL-KDD dataset. A tenfold cross-validation test was performed on the training dataset to obtain the data. Table 7 compares the proposed strategy's accuracy to the preceding methods' accuracy using a comparable NSL KDD dataset and class categorization. This comparison is one of Table 7's features. The proposed method outperforms four of the five previously conducted studies utilized as comparisons, providing a DoS class accuracy score of 99.95% and achieving greater accuracy overall. Also, the proposed method outperforms four previous investigations combined, with 99.56 percent as the greatest possible score in the Probe class. Nonetheless, [32], their accuracy score in R2L and U2R drops substantially, whereas our proposed technique achieves a consistent value of 78.06% for R2L and 74.05% for U2R. The proposed method produces a constant accuracy score for each category and exceeds the accuracy levels as compared with majority of the previously examined research.

Table 4.2: Accuracy comparison using the NSL-KDD dataset with previous work.

Dataset	Author	Methods	Data Representation	Model Accuracy
NSS-KDD	Raman, et al. [43]	HG-GA SVM	Tabular Data	96.72
	Padmaja, et al.[24]	ANN	Tabular Data	94.50
	Chu, et al. [40]	SVM-RBF	Tabular Data	97.02
	Ingre, et al. [35]	Decision Tree (DT) based CART	Tabular Data	90.03
	Salama, et al. [41]	DBN-SVM	Tabular Data	92.84
	Wang, X et al. [38]	KNN	Tabular Data	87.65
	Proposed Model	CNN	Image Representation	97.05

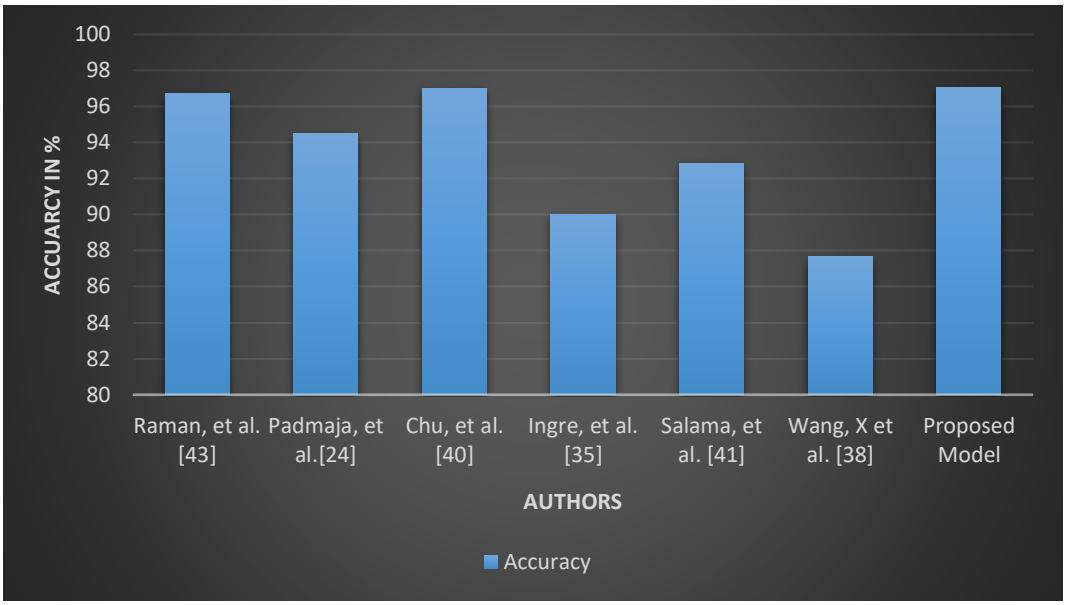


Figure 4.14 Performance comparison of proposed work and contemporary techniques.

Chapter 5

CONCLUSION AND FUTURE SCOPE

Conclusion

This thesis focuses on creating efficient methods for deploying anomaly-based intrusion detection systems (IDSs) using neural network classifiers for the NSL-KDD dataset. The primary goal is to develop a cloud-based IDS that can detect and mitigate DDoS attacks. The approach involves thoroughly analysing the current state-of-the-art in neural network classifiers, followed by building an IDS based on a neural network that can efficiently identify network intrusions in a cloud environment.. Using neural networks for intrusion detection in the cloud is an intriguing new direction. We have offered a few potential ways to boost Anomaly detection efficiency as part of our continuing study. The existence of irrelevant characteristics and outlier data in the dataset is exacerbated by many data and features employed in IDS. This is the root of the issue. The work employs a deep learning technique to identify and classify DDoS attacks on cloud computing systems. It has been proved that the implemented IDS is accurate and that the time required to complete it is appropriate.

The study employs a convolutional neural network model to classify attacks and demonstrates that CNNs trained on image representations exhibit better performance compared to other classifier models trained on the original tabular data. In addition, the model's structure is versatile enough to be expanded to incorporate different kinds of data. Furthermore, after conducting a comprehensive comparison of the models based on various aspects such as the duration of generating image representations, the performance of image representations, and the time required for training prediction models, our proposed CNN model tends to perform better. Furthermore, in terms of the preservation of feature structure, images would better represent the relationship between features as compared with raw data. As cloud computing becomes more prevalent, the need for effective intrusion detection techniques will only increase, and neural networks may offer a powerful solution.

Future scope

Another minor issue is that ANN needs to catch up in some areas, including further detection precision for low-frequency attacks and detection stability. As a result, it is critical to evaluate various approaches to solving this problem. The method for detecting anomalies might be enhanced further by including functionalities for detecting misuse. The goal is to build a database of informed people to achieve knowledge-based detection. There exists significant potential for additional experimentation and investigation of various underrepresented datasets for minority data within the anomaly detection domain in forthcoming times. Through the implementation of supplementary experiments on diverse under-sampling methodologies and their comparison with prior research, a more profound comprehension of the efficacy and applicability of these techniques can be obtained. The proposed investigation has the potential to enhance the overall comprehension of the efficacy of anomaly detection models when applied to under-sampled data. Furthermore, it may facilitate the creation of more resilient and precise detection techniques. Moreover, a thorough examination of the effects of diverse under-sampling techniques on distinct categories of datasets can yield significant knowledge regarding their suitability across diverse fields and data attributes.

Moreover, it is possible to transmit the results of the attack categorization model to a Security Information and Event Management (SIEM) platform for additional examination and response. Security Information and Event Management (SIEM) systems possess the capability to collect and analyse data from multiple sources such as intrusion detection and prevention systems, firewalls, and other security solutions. This enables them to offer instantaneous insight into security events and potential threats. Incorporating additional network datasets, such as SIEM, firewall logs, and web server logs, into the existing model through feature engineering methods presents a promising avenue for future exploration in this project. The task at hand necessitates the mapping of the data to the features employed in our model, and possibly adapting the model architecture to fit the new dataset. Enhancing the model's capability to identify a wider range of network intrusions could potentially augment its overall accuracy.

References:

[1] Zhuang, Fu & Lin, Guoyuan & He, Huanye & Zhang, Yifan & Li, Yonggang & Gu, Hao. (2021). Improved t-SNE in Anomaly Detection of Cloud Virtual Machine. Enterprise Information Systems. 1-26. 10.1080/17517575.2021.1995784.

[2] Orevi, H. Musafer, A. Alessa, M. Faezipour, and A. Abuzneid, "Features di mensionality reduction approaches for machine learning based network intrusion detection," Electronics, vol. 8, no. 3, p. 322, 2020.

[3] Li, Yun & Liu, Jie & Cao, Bin & Wang, Chonggang. (2018). Joint Optimization of Radio and Virtual Machine Resources with Uncertain User Demands in Mobile Cloud Computing. IEEE Transactions on Multimedia. PP. 1-1. 10.1109/TMM.2018.2796246.

[4] Alsmadi, Duha & Prybutok, V.R.. (2018). Sharing and storage behavior via cloud computing: Security and privacy in research and practice. Computers in Human Behavior. 85. 218-226. 10.1016/j.chb.2018.04.003.

[5] Shroff, Jugal & Walambe, Rahee & Singh, Sunil Kumar & Kotecha, Ketan. (2022). Enhanced Security Against Volumetric DDoS Attacks Using Adversarial Machine Learning. Wireless Communications and Mobile Computing. 2022. 1-10. 10.1155/2022/5757164.

[6] Cepheli, Özge & Buyukcorak, Saliha & Karabulut Kurt, Gunes. (2016). Hybrid Intrusion Detection System for DDoS Attacks. Journal of Electrical and Computer Engineering. 2016. 1-8. 10.1155/2016/1075648.

[7] Khalaf, Bashar & Mostafa, Salama & Mustapha, Aida & Mohammed, Mazin & Mahmoud, Moamin & Ali, Bander & Al-rimy, Bander & Razak, Shukor & Elhoseny, Mohamed & Marks, Adam. (2021). An Adaptive Protection of Flooding Attacks Model for Complex Network Environments. Security and Communication Networks. 2021. 5542919. 10.1155/2021/5542919.

[8] Omar, H. Musafer, A. Alessa, M. Faezipour, and A. Abuzneid, "Features di mensionality reduction approaches for machine learning based network intrusion detection," Electronics, vol. 8, no. 3, p. 322, 2018.

[9] Dehraj and sharma (2019). A Research of Virtual Machine Resource Scheduling Strategy Based on Cloud Computing and autonomous software. 294-304. 10.1007/978-981-10-0356-1_30.

[10] Fernandez, Xavier & Vega-Barbas, Mario & Villagra, Victor & Rivera, Diego & Alvarez-Campana, Manuel & Berrocal, Julio. (2020). Efficient Distributed Preprocessing Model for Machine Learning-Based Anomaly Detection over Large-Scale Cybersecurity Datasets. Applied Sciences. 10. 3430. 10.3390/app10103430.

[11] Ramachandra, A. Habibi Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," Proceedings of the 4th International Conference on Information Systems Security and Privacy, 2016.

[12] Prashanthi and kikoi, Santosa, R. Hartanto, and K. Sakurai, "Machine learning-based IoT-botnet attack detection with sequential architecture," Sensors, vol. 20, no. 16, p. 4372, 2019.

[13] Katsaros Machine Learning Based Anomaly Detection and Classification of Acoustic Emission Events for Wear Monitoring in Sliding Bearing Systems. Tribology International. 10.1016/j.triboint.2020.106811.

[14] Rahmanian AA, Ghobaei-Arani M, Tofighy S (2018) A learning automata-based ensemble resource usage prediction algorithm for cloud computing environment. Future Gener Comput Syst 79:54–71. https://doi. org/10.1016/j.future.2017.09.049

[15] Mehmood T, Latif S, Malik S (2018) Prediction of cloud computing resource utilization. In: 2018 15th international conference on smart cities: improving quality of life using ICT IoT (HONET-ICT), pp 38–42. https://doi.org/10.1109/HONET.2018.8551339

[16] Malik S, Tahir M, Sardaraz M, Alourani A (2022) A resource utilization prediction model for cloud data centers using evolutionary algorithms and machine learning techniques. Appl Sci. https://doi.org/ 10.3390/app12042160

[17] Cao, J. & Zeng, G.-S & Kuang, G.-J & Zhang, J.-W & Ma, Hong-Yan & Hu, K.-K & Niu, J.. (2017). On-demand physical resource allocation method for cloud virtual machine to support random service requests. Ruan Jian Xue Bao/Journal of Software. 28. 457-472. 10.13328/j.cnki.jos.005054.

[18] Shafi, Qaisar & Qaisar, Saad & Basit, Abdul. (2019). Software Defined Machine Learning Based Anomaly Detection in Fog Based IoT Network. 10.1007/978-3-030-24305-0_45.

[19] Cholakoska, Ana & Rakovic, Valentin & Gjoreski, Hristijan & Kalendar, Marija & Pfitzner, Bjarne & Arnrich, Bert. (2021). Machine Learning based Anomaly Detection in Ambient Assisted Living Environments.

[20] Sahu, Nilesh & Mukherjee, Indrajit. (2020). Machine Learning based anomaly detection for IoT Network :(Anomaly detection in IoT Network). 10.1109/ICOEI48184.2020.

[21] Sa, Sajidha & A., Vijayalakshmi & M., Nisha & Sisodia, Mayankita. (2022). A comparative study of machine learning based anomaly detection for IoT data using SPARK. AIP Conference Proceedings. 2516. 240002. 10.1063/5.0109693.

[22] Xu, B. & Zhao, C. & Zhu, Y.-J & Peng, Z.-P. (2014). Virtual machine resource scheduling multi-objective optimization in cloud computing. Xitong Fangzhen Xuebao / Journal of System

[23]Kaur, Harjinder & Singh, Gurpreet & Minhas, Jaspreet. (2013). A Review of Machine Learning based Anomaly Detection Techniques. International Journal of Computer Applications Technology and Research. 2. 10.7753/IJCATR0202.1020. Simulation. 26. 592-595.

[24] B. Padmaja, K. Sai Sravan, E. Krishna Rao Patro, and G. Chandra Sekhar, "A System to Automate the Development of Anomaly-based Network Intrusion Detection Model," in 1st International Conference on Applied Mathematics, Modeling and Simulation in Engineering (AMSE), 15-16 September 2021, India (Virtual). IOP Publishing Ltd., 2021, pp. 012015

[25] Mitra, Peetak & Akhiyarov, Denis & Araya-Polo, Mauricio & Byrd, Daniel. (2020). Machine Learning-based Anomaly Detection with Magnetic Data. 10.20944/preprints202012.0092.v1.

[26] Cheng, Y. Tan, and X. Gu, "Pal: P ropagationaware anomaly localization for cloud hosted distributed applications," in *Managing Large-scale Systems via the Analysis of System Logs and the Application of Machine Learning Techniques*, 2021, pp. 1-8.

[27] Gadal, Saad & Mokhtar, Rania & Abdelhaq, Maha & Alsaqour, Raed & Sayed Ali Ahmed, Elmustafa & Saeed, Rashid. (2022). Machine Learning-Based Anomaly Detection Using K-Mean Array and Sequential Minimal Optimization. Electronics. 11. 2158. 10.3390/electronics11142158.

[28] Abdelli, Khouloud & Cho, Joo & Tropschug, Carsten & Pachnicke, Stephan. (2022). Machine Learning-based Anomaly Detection in Optical Fiber Monitoring. Journal of Optical Communications and Networking. 14. 10.1364/JOCN.451289.

[29]Miao, Xuna & Wu, Xiaobo. (2018). Virtual machine anomaly detection strategy based on cloud platform operating environment perception: Cloud platform-based virtual machine anomaly detection strategy. Concurrency and Computation: Practice and Experience. 30. e4656. 10.1002/cpe.4656.

[30]Lin, Mingwei & Yao, Zhiqiang & Gao, Fei & Li, Yang. (2016). A Virtual Machine Instance Anomaly Detection System for IaaS Cloud Computing. International Journal of Future Generation Communication and Networking. 9. 255-268. 10.14257/ijfgcn.2016.9.3.23.

[31]Shi, Feng & Lin, Jingna. (2022). Virtual Machine Resource Allocation Optimization in Cloud Computing Based on Multi objective Genetic Algorithm. Computational Intelligence and Neuroscience. 2022. 1-10. 10.1155/2022/7873131.

[32]Yan, L. & Chenxian, G.. (2015). The virtual machine resource allocation based on service features in cloud computing environment. 9. 639-647.

[33] Nie, Jun. (2016). A Research of Virtual Machine Resource Scheduling Strategy Based on Cloud Computing. 294-304. 10.1007/978-981-10-0356-1_30.

[34] Gondalia, Archana & Shah, Apurva. (2022). A Survey of Advancement in Anomaly Intrusion Detection System. 10.21203/rs.3.rs-2284207/v1.

[35] Ingre, Bhupendra & Yadav, Anamika & Soni, Atul. (2017). Decision Tree Based Intrusion Detection System for NSL-KDD Dataset. 10.1007/978-3-319-63645-0_23.

[36] Jivani, Anjali & Shah, Karishma & Koul, Shireen & Naik, Vidhi. (2016). The Adept K-Nearest Neighbour Algorithm - An optimization to the Conventional K-Nearest Neighbour Algorithm. Transactions on Machine Learning and Artificial Intelligence. 4. 10.14738/tmlai.41.1876.

[37] Sharma, Ankit. (2022). Intrusion Detection System using Multi-Layer Perceptron with Grid Search CV. International Journal for Modern Trends in Science and Technology. 8. 98-101. 10.46501/IJMTST0807016.

[38] Wang, X., & Li, K. (2019). Anomaly detection algorithm of intrusion detection system based on machine learning. Journal of Ambient Intelligence and Humanized Computing, 10(1), 253-260.

[39] Zhang, Shuai, Y.-L. S. A. (2017), 'Deep learning-based recommender system: a survey and new perspectives', Journal of ACM Computing Surveys 1(1), 1–35.

[40] W.-L. Chu, C.-J. Lin, and K.-N. Chang, ''Detection and classification of advanced persistent threats and attacks using the support vector machine,'' Appl. Sci., vol. 9, no. 21, p. 4579, Oct. 2019.

[41] M. A. Salama, H. F. Eid, R. A. Ramadan, A. Darwish, and A. E. Hassanien, ''Hybrid intelligent intrusion detection scheme,'' in Soft Computing in Industrial Applications. New York, NY, USA: Springer, 2011, pp. 293–303

[42] Russo S, Besmer MD, Blumensaat F, Bouffard D, Disch A, Hammes F, Hess A, Lürig M, Matthews B, Minaudo C, Morgenroth E, Tran-Khac V, Villez K. The value of human data annotation for machine learning based anomaly detection in environmental systems. Water Res. 2021 Nov 1;206:117695. doi: 10.1016/j.watres.2021.117695. Epub 2021 Sep 27. PMID: 34626884.

[43] M. R. G. Raman, N. Somu, K. Kirthivasan, R. Liscano, and V. S. S. Sriram, ''An efficient intrusion detection system based on hypergraphGenetic algorithm for parameter optimization and feature selection in support vector machine,'' Knowl.-Based Syst., vol. 134, pp. 1–12, Oct. 2017.