ADDIS ABABA UNIVERSITY

ADDIS ABABA INSTITUTE OF TECHNOLOGY - AAiT

SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING – SiTE

Optimizing Intrusion Detection Systems with Ensemble Deep Learning: A Comparative Study of RNN and LSTM Architectures

BY

ADMASU AWASH

Advisor

HENOCK MULUGETA (PHD)

OCTOBER 2024

ACCEPTANCE

Optimizing Intrusion Detection Systems with Ensemble Deep Learning:
A Comparative Study of RNN and LSTM Architectures

BY

ADMASU AWASH

Accepted by the Addis Ababa University, Addis Ababa Institute of Technology - AAiT
School of Information Technology And Engineering –SiTE

Thesis Examination Committee:

Internal Examiner

External Examiner

Dean, Faculty of AAiT

OCTOBER 2024

DECLARATION

I, the undersigned, declare that this proposal work is my original work, has not been presented for a degree in this or any other universities, and all sources of materials used for the thesis work have been duly acknowledged.

Admasu AWash Wete

Name of Student

Signature

Addis Ababa

Ethiopia

This proposal has been submitted for examination with my approval as advisor.

Henock Mulugeta (PHD)

Advisor

Signature

Addis Ababa

Ethiopia

OCTOBER 2024

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest thanks to the Almighty God;

My advisor Henock Mulugeta (PHD), I would like to express my sincere gratitude to his valuable time, guidance and comments which enabled me to gain good research experience;

I am extremely grateful to my parents for their love, prayers, caring and sacrifices for educating and preparing me for my future. They have been my source of strength next to God. Last but not least, I would like to thank my friends and colleagues.

Abstract

Nowadays, due to the complexity and severity of security attacks on computer networks attackers can launch a variety of attacks against organizational networks using a variety of methods in order to access, modify, or delete crucial data. The rise in cyberattacks has made it necessary to create reliable and effective intrusion detection systems (IDS) that can instantly recognize malicious activity. IDS, which can automatically and quickly detect and categorize cyberattacks at host and network levels, has made substantial use of machine learning techniques. Although ML techniques like K Nearest Neighbor and Support Vector Machines have been used to building IDSs, those systems still have a high false alarm rate and poor accuracy. Many security researchers are integrating different machine learning approaches to protect the data and reputation of the organizations. Deep learning algorithms have emerged as a forceful instrument in this field and these can detect with better precision than conventional techniques. Recently, Deep learning has become more well-known in network-based intrusion detection systems, enhancing their efficiency in safeguarding hosts and computer networks. In the field of deep learning, ensemble learning has appeared as a potent method that improves the performance of single models by combining several of them. The present study employed two architectures of recurrent neural networks (RNNs), namely simple recurrent neural networks and long short-term memory (LSTM), in order to investigate the possible applicability of ensemble learning in intrusion detection systems (IDS). RNNs are suited for predicting sequential data in IDS by identifying temporal relations in network traffic. LSTMs, which are a kind of RNN, can deal with long-term dependencies well and help avoid vanishing gradient problem that is important in identifying complicated intrusion model.The performance of designed model and the IDS were evaluated using LITNET2020 publicly available dataset under performance evaluation metrics. In multiclass classification the ensemble model fared better than LSTM, yielding accuracy and precious 99.981% and 99.965%, respectively, whereas LSTM provided accuracy and precious of 99.638% and 99.451 %, respectively. Additionally, the suggested ensemble approach produced superior in multi-classification results for the various types of intrusions.

Keywords: Deep Learning, Intrusion Detection System, network based Intrusion Detection System ,Recurrent Neural Network

Table of Contents

List of Figures

List of Tables

List of Acronyms

AE	Auto Encoder
AI	Artificially Intelligence
ALOA	Ant Lion Optimization Algorithms
AMI	Advanced Metering Infrastructure
ANN	Artificial Neural Networks
AUC	Area under Curve
BCM	Ball vector machine
BM	Boltzman Machine
CIA	Confidentiality Availability and Integrity
CNN	Convolutional Neural Networks
CART	Classification and Regressing Tree
DBN	Deep Belief Networks
DL	Deep Learning
DNN	Deep Neural Network
DOS	Denial of Service
DT	Decision Trees
ELM	Extreme Learning Machine
FFNN	Feed Forward Neural Network
FLN	Fast Learning Network
FN	False Negative
FP	False Positive
GA	Genetic Algorithm
GP	Genetic Programming
GAN	Generative Adversarial Network
GPU	Graphical Processing Unit
GWO	Grey Wolf Optimizer
HIDS	Host-Based Intrusion Detection System
HS	Harmony Search
HTTP	Hyper Text Transfer Protocol
ICMP	Internet Control Message Protocol

IDS	Intrusion Detection System
IPS	Intrusion Prevention System
IOT	Internet of Things
IT	Information Technology
LITNET	Lithuanian Research and Education Networ
MLP	Multi Layer Perception
ML	Machine learning
NB	Naive Bayes
NIDS	Network-Based Intrusion Detection System
NLP	Natural language processing
KNN	k-nearest neighbor
PSO	Particle Swarm Optimization
RNN	Recurrent Neural Networks
RVM	Relevance Vector Machine
SCA	Sine Cosine
SMOTE	Synthetic Minority Oversampling Technique
SOM	Self Organizing Map
STL	Self Taught Learning
SVM	Support Vector Machines
TCP	Transmission Control Protocol
TP	True Positive
TN	True Negative
UDP	User Datagram Protocol
WMV	Weighted Majority Voting

CHAPTER 1

Introduction

1.1 Background of the study

In today's rapidly changing technological environment, application systems and computer networks are critical for efficient business processes. They allow for smooth transfer of resources such as data, processing capacity, storage and information. Individuals, firms, and governments are increasingly storing sensitive information about their everyday social interactions at Internet [1]. With the growth requirement of internet, information security field is facing more and more serious challenges from diversified threats. Therefore, it is vital to implement mitigation techniques[2]. Some of them are setting up firewalls, cryptography, IDS, authentication, and authorization systems, as well as antivirus software. IDS has been identified as one of the most promising methods for protecting against intricate and dynamic intrusion behaviors. [3].

Signature-based NIDS and anomaly detection systems are the two categories of IDS methods. By analyzing patterns across all of the retrieved data, NIDS uses signatures to find intrusions. Conversely, anomaly-based intrusion alerts detect any significant deviations from the usual traffic pattern in the monitored user behavior. To put it another way, signature-based NIDS is more successful at noticing known attack patterns, whereas anomaly detection NIDS performs significantly better at detecting new attack patterns. However, it often leads to false alerts because of variations in intruder behavior[4].

Hardware and software are used to IDS to monitor system and networks for intrusion. Furthermore, network packages, rootkit analyzer, and system logs are detected by IDS, which then reports its result to the main server or retain them in machine. It continuously monitors the system and network for various Malicious events that can intrude and crush the functioning system[5]. This is normally carried out by means of routinely collecting information from a whole lot of systems and network sources, and then reading the statistics for viable safety issues. There is no 100% guarantee regarding the security of any data on the network that is

connected to the Internet. Rather it is encouraged to use distinctive ways as an optional to mitigate any risk in line with IEEE x.805 eight safety dimensions map to the protection threats. As stated on the paper [6], The eight protection dimensions are availability, private, data integrity, confidentiality, authentication, non-repudiation, and communication security.

Researchers have proposed several machine learning(ML) and deep learning(DL) techniques over the past ten years to improve IDS's capacity to identify malicious attempts[7]. ML can derive relevant information automatically from large databases. With enough training data provided, detection levels achieved by ML based IDSs may be higher than those reached without them. Moreover, because they don't heavily rely on domain knowledge, machine learning-based IDSs are easy to create and implement[7]. One kind of ML that can yield more successful outcomes is DL. Jakhar and Kaur describe DL as a part of Artificial Intelligence that is used for solving computational issues through models and algorithms which imitate biological neural networks found in the human brain[8]. Fundamentally, DL is a group of ML methods that use artificial neural networks built on various hierarchies that correspond to different levels of abstraction to learn[9]. It is possible to learn feature representations through DL algorithms, It would subsequently be used to turn raw input into finished output. DL is characterized by a deep structure consisting of multiple hidden layers[7]. DL techniques exceed conventional methods of ML when they work with a lot of data [10]. Nevertheless, due to the significant rise in network traffic and the security threats it poses, there are many challenges that IDS systems need to overcome in order to detect malicious intrusions quickly. DL techniques for IDS are still being researched, and there are still ways to improve this technology's effectiveness in IDS.

Deep learning is gaining popular in study because it has benefits for addressing these problems. DL uses numerous layers of hierarchical information processing to accomplish tasks like feature learning, classification, and pattern recognition instead of relying solely on conventional machine learning techniques[11].The current study problem focused on improving IDS ability to detect sophisticated and dynamic cyber threats with better accuracy and efficiency. Previous studies on IDS have been hindered by several shortcomings, including the reliance on outdated datasets and a limited range of attack scenarios. Furthermore, these studies typically employed individual deep learning algorithms without implementing data balancing techniques to address class imbalances in the datasets.To overcome these limitations, the current study is utilizing the

LitNet2020 real-time network dataset, which provides a more up-to-date and comprehensive set of network activities for analysis. Additionally, the study is implementing Synthetic Minority Over-sampling Technique (SMOTE) data balancing techniques to mitigate class imbalances and improve the model's ability to learn from all classes equally.

Moreover, rather than relying on a single deep learning algorithm, the study is adopting a more sophisticated method by merging two RNN architectures: Simple RNN and LSTM. This combination allows for a more detailed analysis of sequential data patterns and improves the IDS's ability to detect and respond to evolving cyber threats effectively.This research intends to contribute to the field of cybersecurity by creating a stronger and precise IDS which can adapt to the dynamic nature of modern cyber threats, thereby improving network security and threat detection in order to incorporate these advancements used in the experimental work

1.2 Motivation of the study

The development of technology in the 21st century would determine how much the world economy grows. Key industries include big data, cloud computing, social media, and IoT , and artificial intelligence for leading countries changing into digital-based economies. Nonetheless, increased reliance on technology-driven computers and networks also leads to security flaws and attempts by a number of bad actors to infiltrate key institutions and infrastructures in order to either steal, destroy, or tamper with the key data. Because of the advancements in software and technology the sophistication of cyber attacks is rising, thus in order to protect our data and privacy, our current network systems need to adopt a cyber security culture[12].

As a result, to effectively prevent cyber security threats, a modern IDS must be used to monitor data flows between various networks such as WAN , LAN , identify intrusion attempts and malicious activity, and block them and their data source earlier than They have the ability to erode the infrastructure of the central networks. This study created a network-based approach to identify and detect intrusions using a DL technique to discover characteristics from past attacks in an attempt to decrease the system's vulnerability to fresh attacks.

1.3 Statement of the problem

Recently, Information Technology and Communication has gradually become quite significant in our day to day activities . Computer networks have become much more complex and expansive as a result of Internet usage.With advancement in information technology , more people are using network devices and services[13]. Consequently, the attack surface grown significantly, requiring the implementation of cyber security defense measure. To identify and manage dangerous activity on computer networks. Among the most significant and vital components of security infrastructures used to increase computer system security is IDSs. Over the last years, Numerous machine learning models have been created and evaluated. While some IDS employed feature selection, others used classification techniques including SVM, K-nearest, and decision trees. Almost all the intrusion detection systems based on machine learning (ML) are shallow learning or single feed forward networks that are built by manual feature engineering for their ML model features [14]. Moreover, shallow learning has demonstrated its incapacity to address current day environmental problems, due to the massive number of data entries. Hence, there has been a rise in deep learning models like recurrent neural network (RNN), variational autoencoder (VAE) and long short term memory (LSTM) in the recent years.

The purpose of this study is to address the limitations in previous studies that were depend outdated datasets and few attack scenarios that don't fairly represent current network activity, which restricts the relevance of their conclusions to existing real world situations. Previous studies typically employed individual deep learning algorithms without utilizing data balancing techniques. In order to enable intrusion detection systems to respond effectively as well as accurately to complex and changing cyber threats, this paper employs real time network dataset of LITNET2020 which employs SMOTE data balancing techniques , and combines two RNN architectures, namely Simple RNN and LSTM.

Deep learning methods are currently in high demanded across a number of industries because of the fact that continuous improvements in big data and computing capacity. How effective IDSs is determined by their capacity to identify attack related datasets that containing both regular and abnormal traffic[15]. Traditionally, several benchmark datasets like KDDCup'99 and NSL-KDD have been employed in evaluating IDS; however, these datasets are now outdated

as a result of the rapid advancement in network technologies and new cyber security threats. Many existing datasets were artificially generated or collected from controlled environments; thus they do not represent real world network traffic making them less effective in detecting actual attacks[16]. Furthermore, with increase in size of network traffic grows, manual data labeling also becomes impractical. Hence there is need for realistic and current network flow datasets that are capable of assessing performance of IDS effectively. This study provides an alternate benchmark dataset derived from real world network traffic depicting several types of attacks with normal behaviour over a longer time period addressing some shortcomings exhibited by previous ones[5].

Furthermore, current research frequently ignores the possibility of improving accuracy and stability by utilizing ensemble methods to bind the combined power of several algorithms[17]. Some of the shortcomings are, low detection rate, high training time, low processing speed, relatively high false alarm rate (FAR). This paper aims to enhance the accuracy and reliability of an IDS by integrating RNNs and LSTMs algorithms within a deep learning framework. Ensemble techniques offer a promising way to take advantage of enhance system efficiency by leveraging the harmonizing strengths of diverse models. RNNs and LSTMs are excellent tools for IDS because they can manage sequential data and take into account long term dependencies[18]. RNNs are good with sequential information, which is why it's crucial for evaluating network traffic; meanwhile, LSTMs recognize complex patterns and are resistant to noise that results in improved accuracy. Classification problems have been solved better by LSTM architectures than any other methods[19].

However, deep learning still faces difficulties when it comes to learning from datasets with skewed class distributions. A very unequal distribution of network traffic makes it extremely tough to further research on deep learning-based IDSs since most Internet users behave normally and there are very few harmful attempts[20]. Consequently, a huge quantity of traffic is generated in cyberspace. Yet, the spread of different kinds of malevolent traffic is also uneven with only a small portion being malicious. Deep learning networks require numerous training samples thus their performance becomes more stable with much data available for training. However when learning from unbalanced datasets outcomes are much poorer when using deep learning algorithms [14].In order to improve performance on classifying imbalanced data

This paper focuses on resampling strategies using SMOTE (Synthetic Minority Over-sampling Technique). SMOTE is used to address imbalanced datasets by generating synthetic samples for the minority class, thereby improving model performance in detecting rare instances and preventing bias towards the majority class[21].

Despite all these research studies on IDSs, new technologies must be advanced and existing intrusion detection systems must be improved in order to keep defenses against these attacks given the rising frequency of cyber-attacks. In distributed, dynamic, diverse and wireless computer environments, traditional information security technologies and procedures are frequently not able to address newly emerging cyber security problems. To combat cyber security threats, Novel methods and technology are required for network security. Therefore, deep learning could be a possible solution for the intrusion detection issue because it works so well under complex and large- scale data conditions.

1.4 Research questions

Various IDS problems have been brought up and resolved in various studies. We would respond to the following in this study:

- How do Simple RNN and LSTM deep learning techniques improve the accuracy and efficiency of IDS in detecting complex and evolving cyber threats ?
- how Does Applying techniques for handle imbalanced datasets to enhance the detection rates for minority class attacks in IDS?
- How can the integration of RNN and LSTM models enhance the performance of an IDS compared to using each model individually ?

1.5 Objective of the study

1.5.1 General Objective

The general objective of the thesis is to design and implement model for NIDS for network attacks using DL algorithm and assess the performance of the developed model.

1.5.2 Specific Objectives

The study would have the following specific objectives:

- To investigate and develop deep learning-based approaches that enhance the accuracy and efficiency of IDS in identifying complex and evolving cyber threats.
- To develop effective techniques for balancing imbalanced datasets to improve the performance and reliability of deep learning models, particularly in accurately detecting minority class IDS.
- To evaluate whether the integration of RNN and LSTM models can enhance the performance of IDS When considering the performance of employing RNN or LSTM models separately, in terms of accuracy, detection speed, and robustness
- Examine the impact of the hyperparameter tuning on the reliability and accuracy of deep learning IDS.

1.6 Expected contribution of the study

A RNN based detection system's architecture and implementation are covered in this paper. The recommended method does not require manual feature selection because it uses a DNN. It benefits the modern, dynamic network environment and significantly reduces the workload required of network experts. Additional advantages of a DL based NIDS using RNN include the following:

- **Enhancing an Intrusion Detection System (IDS):** System administrator and IPS uses the outcome as an input, it helps the prevention mechanism to be proactive rather than reactive. Depending on when the attack happens and what steps are taken in response. These are obviously more reactive approaches than proactive ones, as a model is designed to avert attacks in advance.
- **Enhancing Cyber security knowledge :** Insights provided by this paper are critical to enhance cyber security knowledge, empowering cyber-related organizations to improve their defenses against various forms of threats and protect their networks, systems and sensitive data. In the modern world, one of the most important things any person can

do is learn about cyber security because it has been recognized as a means of preventing and controlling constantly evolving risks. By staying well informed of the latest trends, technologies, and best practices in cybersecurity, organizations can harden their defenses against a wide range of cyber-attacks. The more you know about cyber threats, attack vectors and vulnerabilities empowers cybersecurity professionals to implementing strong security measures ,conduct detailed risk assessments, and develop proactive incident response strategies. However also ensures trust between the stakeholders hence enforcement of compliance regulations even when facing a rapidly evolving threat environment.

- This paper can be a key reference point for the future researchers in the field of IDS . By presenting detail methodologies, results, and discussions, if offers a clear outline for understanding key concepts and challenges in IDS. In addition, researchers can build on the experimental design, data analysis techniques, and performance evaluation matrices discuss in this paper .moreover, the practical implication and recommendations outlined assistance in translating research in to real world applications.

1.7 Scope

This thesis study developed and applied a model for local area network traffic into regular and attack-based NIDS approaches using deep learning techniques.Instead of trying to prevent potential network attacks, its main focus is on detecting possible network attacks, tracking model performance, and categorizing anomalous and normal profiles. The project's objective is to increase detection rates by modeling unknown threats. However, this study does not address HIDS. The dataset utilized in this research was obtained from LITNET, which implies that the trained model cannot be immediately applied to a particular organization network. This is a consequence of each organization's distinct network design and infrastructure. Additionally, grid search with cross validation requires a lengthy computational process.

1.8 Structure of the document

An outline of the thesis's structure is provided below. The first chapter summarizes introduction to this research giving motivation, problem statement, research question, objectives, and anticipated contribution. It also describes the organization of the paper and the scope of the project.

The second chapter that comes after this one offers theoretical Information regarding IDS and associated studies using various detection methods. Additionally covered is the categorization of IDS. The third chapter IDS methods, the dataset, algorithms, and research techniques that this research would make advantage of The study that was carried out, together with the preparation for the evaluation, standards, and performance analysis for the two chosen algorithms, are covered in the fourth chapter. Additionally, it describes the evaluation and study experiment. And the fifth chapter introduces the discussion and concluding remarks on our study, present ideas for improvements and recommendations for future research are forwarded.

CHAPTER 2

LITRATURE REVIEW

2.1 Overview of intrusion detection systems

Network attack are defined as an activities that aim to alter or destroy information and service in computer networks[7]. During a data transfer on particular networks, the attack takes place and thus seeks to affect the soundness or any other aspect of the system by altering its integrity, confidentiality and/or availability of computer network systems in general. For example, Trojan horse emails, Internet worms, unauthorized usage of a system, and DOS by abusing a feature of a system, or exploiting a bug in software to modify system data [22].

There are huge number of IDS mechanisms in service with aim at checking network data for any deviation from normal behavior of a system or of its user [7]. Apart from simple and sophisticated methods hackers possess to carry out their illegal activities, they have in some way exploited the weaknesses in hardware components and software systems of interconnected networks [23]. Some may also search for an already diagnosed behavior of an attack within the data. These systems, known as IDS, employ a variety of methods, including ML algorithms and statistical methodologies. IDS are a crucial tool for network system to detect security holes inside the network. Before further investigation, We can accomplish outline crucial and typically used terms associated with IDS from authors in [24][22].

IDS typically don't block network traffic by design, instead these systems just mark off questionable traffic and store it for future reference either by humans or systems. nevertheless, IPS block all highlighted traffic as suspicious. Most modern IDS and IPS do still produce a lot of false alarms leading to alerts that are as common in benign situations [7]. This may be a problem with IPSs as ordinary traffic gets blocked without any motive. Nevertheless, IDSs are a greater solution since they would absolutely forward or log the entries. In this argument, speed must be considered. IDSs quicker than IPSs Furthermore, the IDS could be more effective if it's being done to process the data while the operation is taking place.

2.1.1 Major types of IDSs

In present times, IDS is available in an extensive variety of designs, Each has a distinct method for observation and analysis. Based on detection techniques, IDS shares the same characteristics as detection-based IDS, signature-based IDS, and anomaly-based IDS. Security is enhanced by making IDS to fall under HIDS, NIDS and mix of the two defensive systems.

2.1.1.1 Signature based detection systems

Signature based IDS , this sort of detection may be very effective towards acknowledged attacks, Additionally, it is dependent on daily pattern changes and is unable to identify previously unknown risks or new one [25].Signature based IDSs have a significant disadvantage in each signature calls for an access within the database, and so a whole database may contain hundreds or lots of entries. Each packet to be in comparison with all the entries within the database. This may be very aid consuming and doing so will gradual down the throughput and making the IDS prone to DOS attacks. Using this weakness, certain IDS evasion tools flood the signature. An IDS that depends on signatures is overloaded with packets and cannot handle the volume of traffic. This leads to the IDS dropping packets and potentially allowing pass over attacks. Further, this sort of IDS remains susceptible against unknown attacks because it relies on the signatures presently within the database to come across attacks [25]

2.1.1.2 Anomaly based detection system

Anomaly-based IDS tries to determine the ”normal” activities of the system that require safety. It then raises an anomaly alarm whenever a given observation exceeds from a certain threshold[26]. These profiles consist of historical data that has been gathered over time from regular business operations. The detectors accumulate facts from the activities and use a lot of measures to determine whilst the monitored hobby deviates from ordinary interest [22]. However, because of the assumptions underlying anomaly detection mechanisms, their false alarm are very excessive. In particular, the primary causes of this restriction includes the consumer's everyday conduct version is based on data accumulated over a duration of ordinary operations, intrusive activities that have been ignored for the entirety of this age probably deserve to be accepted as normal.Also anomaly detection strategies can hardly ever hit upon stealthy attacks because of the reality those varieties of attacks are generally hidden in massive wide variety

of instances of normal behaviors. Moreover, the forms of parameters used as inputs of every-day fashions are normally decided by means of safety professionals. Any mistake happening in the course of the process of defining those parameters will booming the fake alarm charge and decrease the effectiveness of the ambiguity detection gadget. As a result, the layout of the detection strategies and the selection of the machine or network capabilities to be monitored are two of the primary open issues in anomaly detection[22].

2.1.1.3 Host based IDS (HIDS)

This kind is positioned on one device consisting of server or workstations, where the records is analyzed locally to the machine and are collecting this facts from one of a kind resources. In HIDS the anomaly and misuse detection systems collaborate. Agents of a HIDS are software programs running on workstations that required supervised according with established guide-lines. When an agent is operating on device, It can set off alarms and/or write data to log files. HIDSs are installed as simple security solutions for detecting any kind of unwanted behaviors that can compromise system integrity [25].

2.1.1.4 Network-based IDS

NIDS is an application that oversees all data passing through a network. NIDS thus helps in saving both time and money. Cross-referencing allows a comparison of current activities with those stored on other computers for potential risk indicators. For example, DoS attacks can be detected by monitoring network traffic. Instead of installing an IDS on all computers within an organization, one system can be placed to serve the same function [27]. NIDS commonly consists of a network equipment (or sensor) with a separate management interface and network interface card (NIC) that is operate in promiscuous mode. IDS monitors all traffic inside a network part and located next to a boundary.

2.1.1.5 Hybrid based IDS

Each Network-based IDS and Host-based IDS has their own advantages and disadvantages. Consequently, it is best to combine NIDS with HIDS to produce a more effective defense against attacks and flexibility[25].

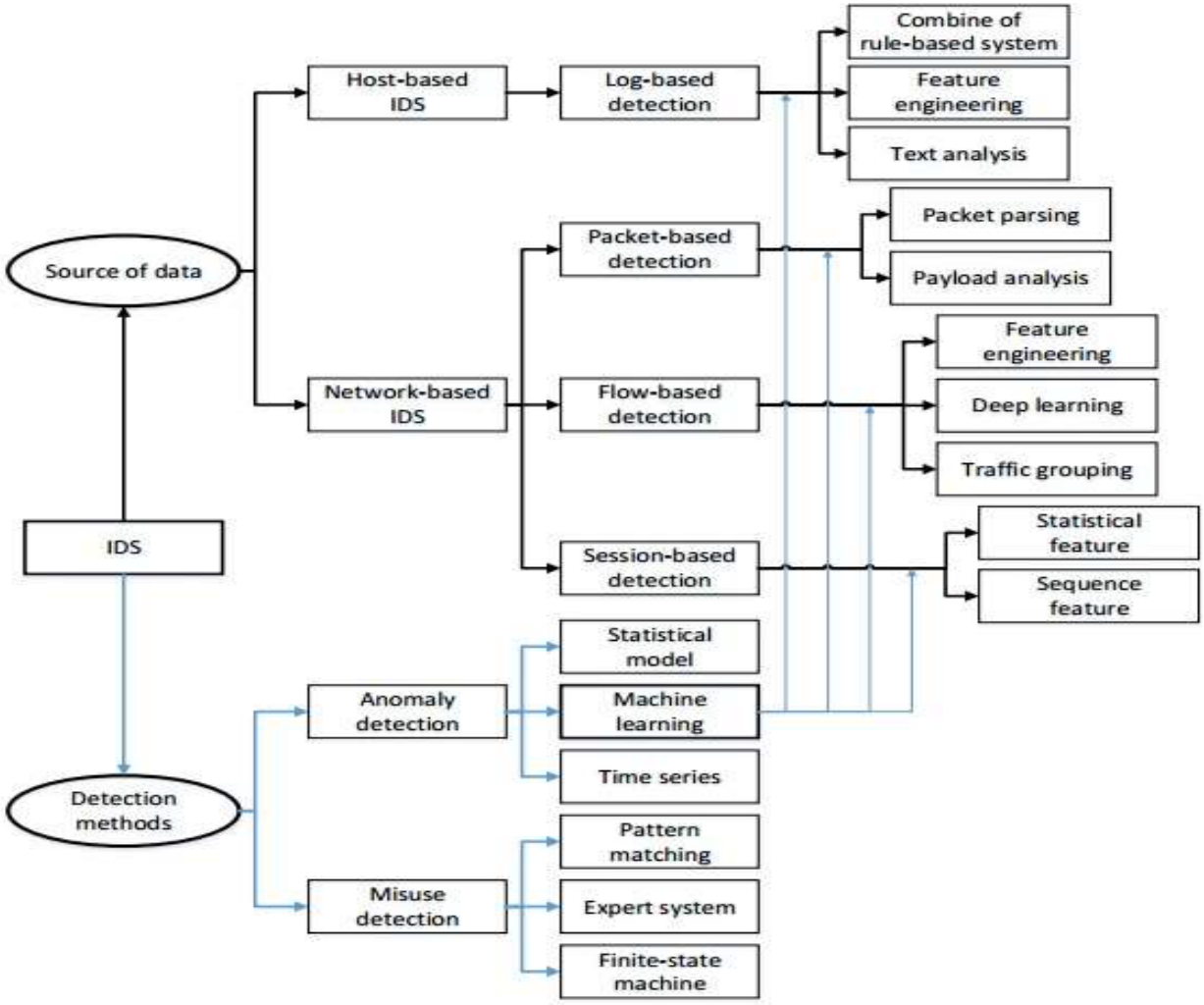

Figure 2.1: Classifications of IDS [7]

2.1.2 Machine Learning methods for IDS

Since Denning first posted the intrusion detection version, numerous IDS techniques have been utilized via research[28]. It is now viable to find particular patterns from network breaches the usage of techniques ranging from easy statistical methods to complex ML and data mining techniques. There are inherent boundaries to the usage of one ML algorithm. For higher performance, numerous getting to know algorithms have currently been mixed.

Ibrahim et al. [15] employ an unsupervised ANN to develop anomaly based IDS. This uses Self Organizing Map (SOM) ANNs for identifying and separating attack traffic from normal traffic. The underlying detection rate is 92.37% on the KDD 99 data set whereas 75.49% on NSL-

KDD dataset. Static networks are less efficient than SOMs. Due to the memory of dynamic networks' ability to learn to detect sequential or time-varying patterns. The use of resilience to attacks and adaptability led to the proposal of a classifier known as GPSVM, which is entirely based on SVM and GP to improve the unusual attack detection cost [24]. According to the experimental findings, GPSVM may generate type accuracy at the NSL-KDD dataset that is more balanced.

Aburomman and Reaz [7] proposed an ensemble production technique which combines critiques from numerous experts into one model. Weighted Majority Voting(WMV) became used to enhance accuracy and the excellent effects were acquired with Particle Swarm Optimization(PSO). However, such classifiers are based totally on binary class strategies which could distinguish among only two states. An ensemble approach primarily based on Bat algorithm (BA) turned into proposed in [29] which used a BA to optimize the real ensemble and an Extreme Learning Machine (ELM) as the foundation classifier, then making use of it to intrusion detection. Although the performance of the ELM is unstable, the approach combining different kind ELMs into ensembles completed better performance than using a single ELM.

Fang et al. [30] propose a method involving ML that enhances IDS in computer systems. They use RNN and support SVM to develop an IDS that ensures the safety of information systems and computing devices connected to a network. Fang et al. [31] assert that their neural network reduces missing text information by using a clustering algorithm and helps to further detect abnormal behavior between packets. Their SVM helps to improve IDS and minor the false alarm rate of the models.

According to the studies presented above, combining different learning algorithms has produced superior significances in terms of enhancing the the classifier's whole performance. Traditional classification algorithms, require manual preprocessing of the information and feature extraction Then they can't independently change the settings. The learning and classification process occasionally needs to involve experts. Since deep learning gives wonderful advantages in many fields, it has attracted a number of attention from both academia and business as a new hotspot in the study of neural networks. Some encouraging values in the place of IDS have been received the usage of DL.

2.2 Deep Learning(DL)

DL changed into first realized through Hinton, one of the creators of Genetic Programming (GP) set of rules, in 2006 [32], and this initiated the rapid growth of the phenomenon. It extends the DNN architecture into multiple hidden layers which try to learn at those levels by representing data as abstract concepts that correspond to nested nonlinear hierarchy within neural networks. This is simply a set of ML algorithms where the data is represented as abstract concepts, and the equal lower-stage principles able to be employed for define many higher-stage concepts [9].

The biggest difference between DL and shallow learning is that deep learning architecture has multiple hidden layers. Deep learning can use very simple algorithms to specify deep structure that can then learn appropriate superficial features that are important in learning high-level concepts, thus overcoming the drawbacks of ML algorithms. Although running the deep learning algorithm requires huge computing power, due to the progress of the graphics processing unit (GPU) and DL can optimize the million-level parameter model into small manageable chunks through independent training of different layers in the model [32], which greatly reduces the required computing resources. Figure 9 shows a three-layered neural network that is fully connected. The layer denoted by 'input' on the left comprises input neurons. There is exactly one output neuron in the 'output' layer situated on the right. The 'output' and 'input' layers are linked by the 'hidden' layer in the middle.

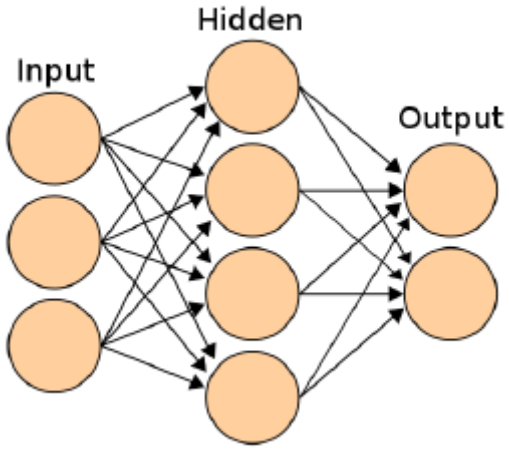

Figure 2.2: three-layered, fully coupled neural network [33]

2.3 Recurrent Neural Network

Neural networks with an internal loop are expanded or categorized as RNNs [34] as illustrated in Figure 2.4. The internal loop of the RNN is used to process sequential information (Scikit-learn, 2019). Additionally, the loop links each layer, stores and memorizes apply it to the current output using the prior input [35].

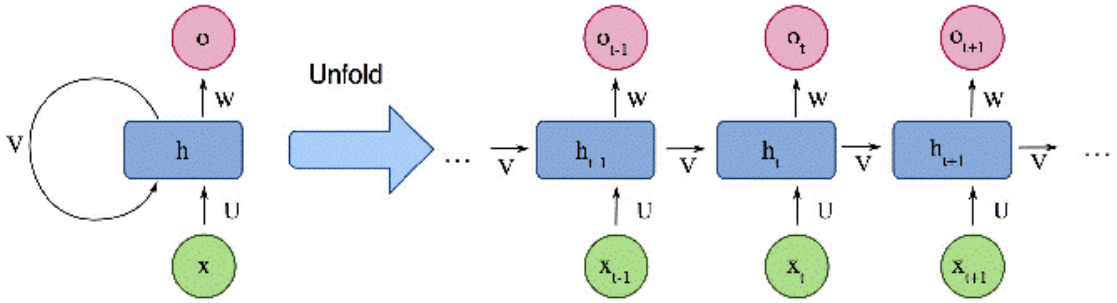

Figure 2.3: simple RNN architecture [36]

There are a variety of variations to the RNN such as the LSTM, and the GRU, which are both specific varieties of RNNs. The capacity of LSTM to learn and predict sequences makes it exceptionally special [37].Neural networks that can handle sequential data are known as RNNs [38]. Their looping back connection in the hidden layers, which allows it to reduce data from past times that expands the capability of traditional neural network

A major problem that hinders the learning process of RNNs is vanishing and exploding gradients. This led to developing LSTM [35]. Hochreiter et al.'s [35] proposed LSTM model to address the RNN's vanishing gradient issue. As seen in Figure 2.5, the LSTM unit is composed of the input gate, forget gate, output gate, and memory cell. Over time, the cell holds onto its values; the input gate establishes the input ratio; the forget gate transfers the previous memory; and the output gate decides whether or not the memory cell's output should be sent [35].

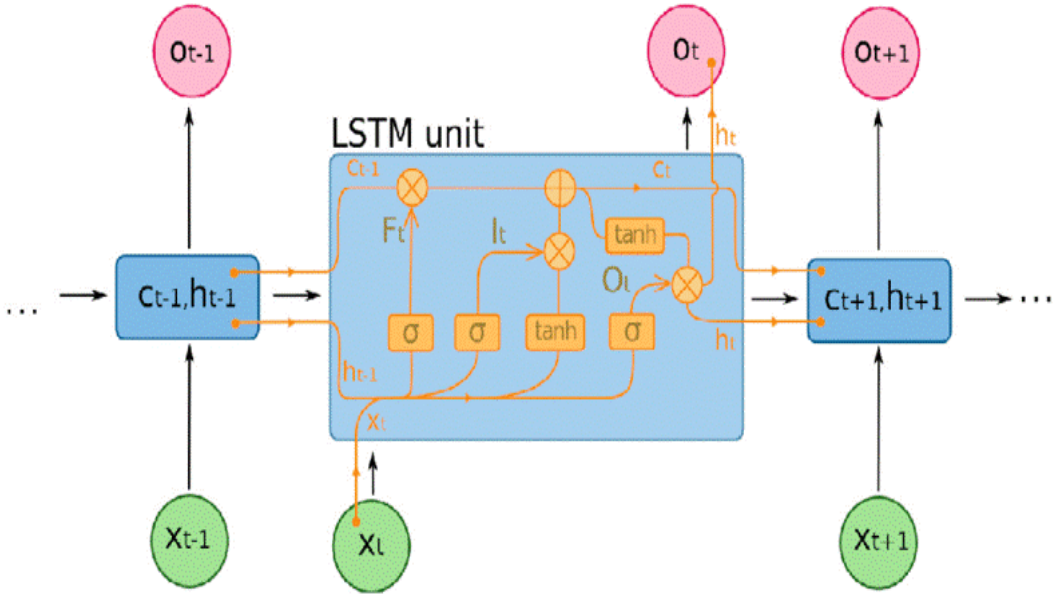

Figure 2.4: LSTM architecture [4]

The prim objective of is to obtain a disappearing gradient descent to prevent problems of long-term dependency[39] Furthermore, the LSTM is regarded by [36] Unlike the traditional RNN model, It can deal with the vanishing gradient problem and capable of long-term learning. Consequently, the LSTM is employed to identify potential network abnormalities exist on a network[39].Further RNN variation of RNN is the GRU architecture, which is considered as a simplification and enhancement of the LSTM [3]. The GRU architecture shown in Figure 2.6. integrates the forget and input gate, to have a single gate referred to as the update gate and it incorporates both the hidden state and the cell state[7]. Two gates are recognized to be present in a GRU: an update gate that establishes how much of the prior memory is to be retained and a reset gate that characterizes the combination of the new input and previous memory. Accordingly, the gated processes that control input, memory, and the information that is now available are employed by the GRU to create predictions [40].

RNNs particularly simple RNN and LSTM networks, are highly useful for enhancing cybersecurity using deep learning because of their proficiency in sequential data analysis and capture temporal dependencies. RNNs and LSTMs have several benefits for cybersecurity, including :-

- **Dealing with sequential Data:** Network traffic, user behavior, and system logs are inherently sequential in nature. This type of data may be processed by RNNs and LSTMs,

which makes them suitable for assessing data streams from cyber security sources[41][42].

- **Capturing Temporal Dependencies:** RNNs and LSTMs can capture the temporal dependencies and patterns in sequential data, which is crucial for detecting anomalies and identifying potential cyber threats. They can learn the normal behavior over time and flag deviations that may indicate malicious activities[41][42].
- **Improved Anomaly Detection:** By learning the normal patterns in sequential data, RNNs and LSTMs can effectively detect irregularities that may indicate cyber attacks, insider threats, or other malicious activities[42].
- **Handling Long-Term Dependencies:** LSTM networks are specifically designed to deal with the issue of vanishing gradients, enabling them to efficiently learn from sequences of data that span across long time periods. As a result, they have the capacity to identify intricate relationships and patterns that may exist over longer time periods.[41].

2.4 Overfitting and Under fitting

For any sample in the training data, models built with DL or other ML approaches are unable to produce reliable predictions, in contrast to unobserved data outside of the training samples. A model is considered correct when it demonstrates the utmost amount of accuracy on both test and training data. If the projections we received were not what we had expected, the model's performance would suffer. Consider that we desire to assess the degree to which our ML model adapts and processes new data. For that we have Over fitting and under fitting, which are majorly responsible for the poor performances of the machine learning algorithms.

2.4.1 Over fitting

When a model covers extreme detail, noise, and It is said to be over fit owed to flaws in the training dataset. which negatively impacts the model's performance on fresh, untested data. Once a model grasped with much data,it starts picking on the noise as well as incorrect data records. Due to noise and a large number of details, this model is unable to classify information correctly.When the model starts categorizing data wrongly, it is often due to having more information in comparison to actual signals. Non parametric and non-linear methods cause over fitting since they have less constraints than parametric approaches, leading to unrealistic mod-

els that don't reflect any sense reality but just mirror training data. There are various methods used for preventing over fitting like cross-validation, feature selection, reducing network size in neural network, dropout and early stopping [43].

2.4.2 Under fitting

Under fitting occurs when prediction model's build process is over simplistic and unable to identify complex pattern based on the practice data Under those circumstances, accuracy on both seen training data and unseen test data will be lower. Under fitting destroys the accuracy of our machine learning model. If this happens, it simply indicates that the algorithm or our model does not well suit the data. Sometimes, when you attempt to create a linear model with nonlinear data and do not have enough information to build an appropriate model .When this happens, the machine learning model's rules are applied too loosely and easily to such few records, which increases the probability that the model is going to generate several incorrect predictions. Under fitting One can prevent underfitting by aid of the use of extra facts and additionally decreasing the features by means of characteristic choice.

2.5 Tuning of Hyperparameters

Hyperparameters are essential to machine learning algorithms because they govern the way the algorithms learn during training and the performance of the final models[44]. A hyperparameter is a setting that is changed during model testing and training to adjust the architecture of a neural network. They are values that a data scientist chooses before beginning to train an algorithm and help in controlling its implementation.For example, for DL algorithms, the subsequent are initialized hyperparameters network weight initialization, activation future, quantity of devices and hidden layers, learning rate, momentum, number of epochs and batch size. Models may have diverse hyperparameters and locating the best combination of parameters may be treated as search problem. There are three methods for adjusting hyperparameters: Bayesian optimization, random search, and grid search[44].

2.6 Related Works

Security experts may be quite concerned about deep learning due to its potential applications in a wide range of fields. Enhancing their network and system security is one use for IDS.

Anomaly IDS prefer using neural networks because of their flexibility and resilience. It has also enabled the use of this approach to anticipate the next command based on user profiles and behaviors [45].Some research studies optimize the weight of the LSTM using a mathematical technique. for example, [46] Trains LSTM to maximize classification accuracy by using four different optimizers such as Sine Cosine Algorithm (SCA) , Grey Wolf Optimizer (GWO) , Ant Lion Optimizer Algorithm (ALOA) , and Harmony Search (HS).

Yin, Zhu, Fei, and He [47], state that one of the best methods for creating a high accuracy classification model is to use an RNN-based intrusion detection system (RNN-IDS). Using the NSL-KDD datasets, they conducted their analysis.When it comes to binary and multi-class classification, RNN-IDS outperforms traditional machine learning methods.Additionally, researchers have compared RNN-IDS's performance with that of a few conventional machine learning methods, including J48, SVM, RF, and ANN. In [48], the authors used the CNN-LSTM model on the KDD Cup'99 datasets and showed that CNN-LSTM performed better than CNN-GRU and CNN-RNN. In [49], the authors used the LSTM-RNN model on the KDD Cup'99 datasets. To ascertain the optimal hyper-parameter values, they conducted a number of tests. They also examined the effects of learning rate and hidden layer size on the classifier's performance.According to Staudemeyer[50] the LSTM-RNN model can identify each type of attacks hidden within KDD Cup'99 training data. They used both extracted minimum feature sets and full features in their experiment for testing purposes. In order to derive minimum input sets from source data, DT and a backward elimination method were employed. Then they assessed how well was their model performing through ROC curve calculation and AUC-value related outcomes.

Ashwaq et al., [51] discuss the growing threats and vulnerabilities associated with IoT landscape. an increase in the quantity of linked devices, which is expected to exceed 20 billion by 2024, underscores the necessity for security in this interconnected environment. In response, these researchers suggest a different technique that utilizes RNN deep learning algorithms to detect intrusions in IoT settings.The work uses the NSL-KDD dataset for training and testing. The suggested RNN model shows remarkable success, identifying intrusions with an accuracy of 87%. The study highlights how crucial it is to leverage machine learning and deep learning techniques to enhance security in light of the evolving IOT. In addition, the authors initiate to

continue working on this project and intend to look into optimization strategies to improve the suggested mode's detection accuracy even more.

Muhuri et al in [19] Provide an RNN-based network intrusion detection solution. The suggested model combines RNN and LSTM for classification, and For feature selection and optimization, it also employs a genetic technique. Using the NSL-KDD dataset, this model seeks to guarantee greater detection rates in comparison to earlier models that have been developed. Moreover, GA optimal feature selection seeks to eliminate bias in LSTM-RNN classification processes. This minimizes training time, lowers the misclassification rate, and increases the model's accuracy. The suggested model contains the 122 original features are extracted from the dataset which have been optimized through Genetic Algorithm (GA) into a dataset of 99 features. The suggested model performed better in binary and multiclass classification tests than RF and SVM. Consequently, the LSTM-RNN with GA greatly improves the accuracy and detection rates of NIDs in comparison to traditional machine learning techniques.

S Amutha et al. [52] proposed an innovative method called NID-RNN to improve Secure NIDS performance. The study explores deep learning by emphasizing the integration of LSTM with NID to prevent malicious activities or cyber-attacks because they are crucial.Taking into consideration binary and multi-class classifications, the researchers conduct thorough analysis on the NSL-KDD dataset, assessing the accuracy and precision of each model. Notably, the suggested NID-RNN approach outperforms other deep learning algorithms with an 8% enhancement in accuracy, according to the data. Network security is strengthened by the RNN model, which successfully classifies attack types with an astounding 99.4% accuracy rate.

Sunil et al. In [53] discussed IDS in which they proposed a new idea by creating a composite framework that combines Bidirectional RNN with LSTM and GRU. They simulated and evaluated their system using the CICIDS2017 dataset. The model showed remarkable 99.13% accuracy in classifying network attacks.In general, this again shows that their model works effectively Regarding both accuracy and false positives as compared to Naıve Bayes. Another point worth raising is how else efficient the model With just 58% of the dataset's attributes, it was able to produce such remarkable results. Thus confirming its resource efficiency as well as practicality.

In the article by Sydney Mambwe Kasongo [54], it was noted that as data transmission volumes continue to grow in communication systems, It is necessary to create an IDS framework using cutting-edge machine learning (ML) technologies. This framework used different types of Recurrent Neural Networks (RNN's) such as LSTM, GRU and Simple RNNs to ensure that network systems are more securely protected. The system was designed with an XGBoost based feature selection algorithm to address reduced test accuracy in identifying new attacks whenever features increase in dimensions. The evaluations on benchmarks like NSL-KDD and UNSW-NB15 showed promising results.In NSL-KDD binary classification tasks, the XGBoost-LSTM model performed better with the test accuracy was 88.13%, the training duration was 225.46 seconds and a validation accuracy of 99.49%. XGBoost-Simple-RNN, with test accuracy of 87.07%, was the most effective model on UNSW-NB15. XGBoost-LSTM and XGBoost-GRU produced different test accuracy results for multiclass classification. On NSL-KDD, the former obtained 86.93% and the latter, 78.40%. In light of the constantly changing nature of cyber threats, these results highlight how effective the suggested IDS is when compared to current techniques.

Paper Ref	**Methodology**	**Dataset**	**Key Findings**	**Research Gap**
[54]	RNN, LSTM and GRU	NSL-KDD and UNSW-NB15	In binary classification XGBoost-LSTM achieved the best performance with a TAC of 88.13% (NSL-KDD) and the XGBoost-Simple-RNN obtained a TAC of 87.07% (UNSW-NB15). For the multiclass configuration, the XGBoost-LSTM obtained a TAC of 86.93% (NSL-KDD) and the XGBoost-GRU got a TAC of 78.40%.	The author suggests an outline for enhancing the model's performance for minority classes as mentioned in the section about future work.Additionally, hybrid approaches which combine many RNN models will be examined and applied.

[53]	LSTM and GRU	CICIDS 2017	The model achieved a classification accuracy of 99.13% with very low false positive rate. The proposed design was compared with others, which are Random Forest, Naïve Bayes, k-Nearest Neighbor (KNN) and Ensemble method.	While the suggested method has demonstrated positive outcomes. However, cannot effectively classify a few attacks. That's why in future research a good classifier and all attack types shall be looked for together with the best training data.
[52]	LSTM and RNN	NSL-KDD	The suggested NID-RNN method achieves an 8% improvement in precision over all existing deep learning techniques. 99.4% accuracy in classifying attack kinds is achieved by RNN.	It does not extensively explore ensemble methods that combine multiple algorithms for improved performance.
[19]	LSTM and RNN	NSL-KDD	provides a fresh perspective on addressing network security challenges and proposed model produced the highest accuracy rate of 96.51% and 99.91% for binary classification using 122 features and an optimal set of 99 features, respectively.	In their recommendations for future work, the authors suggest testing the model with a more recent dataset like UNSW-NB15 in order to verify the performance of LSTM-RNN model and establishing a framework of collecting current network information
[51]	RNN	NSL-KDD	95.4% and 95.6% total accuracy was achieved by the models for the prepartitioned and user-defined multiclass categorization, respectively.	NSL-KDD dataset not represent real world network traffic and diversity of intrusion scenarios..

[47]	RNN	NSL-KDD	Compared to conventional classification techniques like J48, naïve bayesian, and random forest, it achieves a higher accuracy rate, detection rate, and low false positive rate especially when multi-class classification.	The paper evaluates individual RNN algorithms, it does not extensively explore ensemble methods that combine multiple algorithms for improved performance.

Table 2.1: Summary of Literature Review

In the related work section, a review of previous studies in the field of IDS using deep learning approaches exposing various challenges shared by many previous scholars. Many of the papers summarized in Table 2.1 rely on outdated datasets that do not accurately reflect modern network traffic, limiting the applicability of their findings to current real-world scenarios. Deep learning IDS, particularly a RNN, may perform poorly if it has insufficient data. For IDSs, these models are trained to recognize particular network behavioral patterns that can point to an upcoming attack. The RNN may have trouble producing accurate representations of the various forms of network traffic if there isn't much training data available. Therefore, inaccurate outcomes could contain false positives or false negatives. It might also be difficult to get a representative sample of the different kinds of network traffic that the RNN might require to categorize with the data at hand.

Besides, these researches frequently do not investigate the possibilities of improving capturing accuracy and stability by using multiple algorithms combined together via ensemble methods. The purpose of this paper is to improve the precision and dependability of an IDS through the incorporation of ensemble algorithms into a deep learning environment. Ensemble approaches can boost system efficiency by means of fusing strengths from several models.

Another research gap is the capacity to choose only a few attack scenarios for studies. Most early studies on NIDSs using RNN focused on identifying specific attacks like DoS and intrusion attempts. A dataset containing a variety of attack types is used to train an RNN-based

deep learning IDS. Thus, the few attack scenarios provide a research need for a NIDS using DL. The system might not correctly detect such attacks, which could lead to serious security issues and breaches. Another critical issue identified in the reviewed papers is the handling of imbalanced datasets. Many studies do not adequately address the skewed distribution of attack types in their datasets, which can lead to models that are biased towards the majority class and less effective at detecting less common but potentially more dangerous attacks.

Deep learning methods are currently in high demanded across a number of industries because of the fact that continuous improvements in big data and computing capacity.In keeping with this line of reasoning, this paper suggests a DL method for intrusion detection utilizing RNN-IDS. The RNN-IDSs created in the previous studies often employed the KDD Cup 1999 or NSL-KDD benchmark dataset to exclude any chance of a biased IDS. This illustrates the need for impartial RNN-IDS modeling and testing utilizing information from real production networks. Since the effectiveness of NIDS depends on its capacity to recognize attacks,They require access to a dataset of network traffic that includes examples of both typical and unusual network activity. This has been tested a lot for being able to spot network attacks using old benchmark datasets like KDDCup'99 and NSL-KDD. Nevertheless, Because of the rapid pace at which new Kinds of network attacks emerge and evolve alongside advancements in networking technology, these datasets are now no longer useful. The reference datasets for network traffic frequently contain information that has been fabricated on purpose or Those were unreliable of real network flows since they were gathered in a very controlled situation.

The availability of current and realistic benchmark datasets for network flow is necessary To handle the problem discussed above. Therefore, the LITNET-2020 benchmark dataset is used in this study. This data set came from an actually used network and It was employed to assess how effectively does NIDS do when spotting attacks within a network. In this thesis we would like to provide a collection of data that is recent, accurately annotated, publicly available, and has real internet packets representing all major protocols instances of common network activity in addition to a range of network attacks traffic, covers a significant amount of time, and conforms to the observation made by Ring et al. [55]. The dataset was produced at the KTU LITNET network between March 6 and January 31, 2020.

By performing additional research and development, By doing so, we would be bridging the knowledge gap about particular attack scenarios detected by NIDS through enhancing by means of innovative techniques the accurate detection of some attacks by the artificial intelligence technique used in the RNN model, either by collecting more data about such attacks or through improving on the general accuracy of this IDS.

CHAPTER 3
METHODOLOGY

3.1 Overview

In this study, research approach design science is followed and a proposed IDS is presented. Various components of the proposed IDS are described, along with their relevance and the methods employed in creating these components. This includes study design, data types and sources, tools and methods, algorithms, data analysis and presentation, and evaluation metrics. And we summarize the components of RNN based IDS. Data preprocessing, IDS structural models, detection methods, and metrics. Data Preprocessing describes the manner LITNET-2020 dataset are numerically transformed and normalized into fully numerically transformed vectors that can be input to the IDS. The Python programming language is applied to generate an IDS based on LSTMs and RNNs using Keras as a DL framework and Tensor Flow as a backend.

The capacity of RNNs and LSTMs to represent temporal and sequential dependencies in network traffic data makes them the chosen models for NIDS. They are particularly good at managing inputs of varying length, enabling real-time detection, and collecting long-term patterns. Their contextual learning capabilities they are useful for recognizing abnormalities and minor intrusion patterns. Though other algorithms such as CNNs can be effective, RNNs and LSTMs offer certain benefits over other algorithms like CNNs, given that network traffic is flow-based and requires precise and timely detection. In addition, while standard RNNs struggle with remembering previous information over long intervals of time, LSTMs like RNNs can adapt and learn patterns according to sequence length which is an important factor when it comes to analyzing network traffic data that may have variable lengths of sequences This makes them more powerful than other deep learning algorithms, which may not handle serial data as effectively.

3.2 Research Approach

A research approach involves the steps regarding the methods of collecting, analyzing and interpreting data [56]. This study used a quantitative research approach which emphasizes on numbers and figures, therefore, regarded as being scientific [56]. Studies about ANNs are highly based on numbers and figures; therefore, a quantitative research approach is employed in this investigation.

3.3 Research Design

A research design is a structure, plan or strategy of investigation, outlining steps to be followed when collecting and interpreting data in a study, to meet research objectives or answers to research questions [57]. This study employs an experimental research design, which is common in the ANN.

In this thesis, we followed a well-organized method of study made up of numerous stages to ensure extensive analysis. The initial stage was for data preparation in which data was collected, arranged into datasets and checked for completeness and accuracy. This was subsequently followed by pre-processing methods aimed at removing inconsistencies, imputing unknown values and standardizing or normalizing variables depending on situations. Considering the difficulties presented by imbalanced datasets, oversampling, under sampling, synthetic data generation among others were some of our solutions to balancing the data's dispersion. Furthermore, feature selection for better prediction performance was also carried out so as to determine which features are most relevant and informative with regard to target variable. Also we employed ensemble techniques. We wanted to learn diverse patterns through combining several RNNs and LSTMs models thus trying to improve overall model accuracy. The ensemble models were built with the intention that individual RNN models predictions could be aggregated leading to more robust and dependable predictions. We used a variety of metrics to evaluate model performance, including Accuracy, Precision, Recall, F1 Score and AUC. We further tested models' robustness with cross-validation techniques.

3.3.1 Dataset Preparation

When training and testing AI models for reliability, it's important to pay close attention to the type of information being used . In NIDS, the frequently used datasets encompass KDD98, KDDCUP99 and NSLKDD. However, latest studies have shown that these data sets no longer represent the modern network traffic of both normal and attack vectors. Hence we select out few samples from latest data recently enacted by scientists for open use by everyone interested. In fact, we are giving away a brand new benchmark data set gathered from a live network to test how well NIDS identifies network attacks. By considering the study of Ring et al. [58], Our goal is to present data set that is accurate, up to date, freely accessible, and contains instances of usual network activity as well as real-world network traffic containing a variety of network attacks across a substantial time span. From June 3, 2019, until January 31, 2020, the data set was created at the KTU LITNET network.

3.3.1.1 Dataset Description

The dataset LITNET-2020 is an actual network flow dataset taken from LITNET network environment in Lithuania and used for NIDS mainly. The traffic of networks was captured within a time frame of ten months starting from 6th March 2019 to 31st January 2020. It offers real world examples of regular and attack network traffic, aiming to offer a complete dataset for researchers to expand and test intrusion detection models. The dataset includes 85 aspects of network traffic that are described and analyzed in element. These key features are play a crucial role in for both recognizing and classifying various forms of network intrusions [58]. The information collected lasted for ten months hence LITNET-2020 dataset contains an extensive variation of network traffic modes and irregularities to turn into one of the rare ones [58].

3.3.1.2 Attack Types in LITNET-2020 Dataset

Malicious network traffic is here defined as any traffic produced from an attack with the intent of doing harm or intrusion to a computer system or network. The datasets selected for this research contain the following classes of malicious traffic.

Description of Network Attacks The following attack kinds are defined in the proposed dataset.

1. **Smurf attack :-** Sends ICMP broadcast requests from the goal node to the network to

overload it with traffic to motive the centered node to slow down.

2. **An ICMP flood attack :-**We can refer to this particular attack as a ping flood, which requires transmitting an overwhelming number of ICMP destine-unreachable requests (pings).

3. **UDP-flood attack:-** DOS attack that uses the UDP. Data packets on the network using the UDP protocol along with 53 port number are transmitted on each IP address for DNS.

4. **TCP SYN-flood attack:-** DDoS attack that utilizes a flaw within the popular TCP three handshake to overload the victim node's sources and make it unresponsive. S flags are available in the packages, however AFRPU flags are absent.

5. **HTTP-flood attack:-** DDoS attack directed at a web server or program with the aid of taking benefit of requests that appear like valid for the HTTP both GET and POST.HTTP floods in complex Layer 7 attack need much less bandwidth than different type of attacks and might disable the target server or internet site without the use of spoofing, reflection, or malformed packets. Only port 80 is targeted by the attack packets.

6. **LAND attack :-** A Layer 4 DoS is when the attacker node sets both the source and destination TCP segment data. TCP is the protocol used by the attacker packets and have the S flags .Repeated processing of the same packet within the TCP stack results in a suspended state of an attacked node.

7. **W32 Blaster Worm attack:-** takes advantage of buffer overrun vulnerability in Interface for Microsoft Windows DCOM RPC .These attacks target only port 4444 which is the Kerberos and port 135, 69 (TFTP).

8. **Code Red Worm attack:-** Attempts to trigger a target node's buffer overflow problem such that it starts overwriting the adjacent memory. The HTTP GET approach applies to the packets being addressed to the supply IP and most effective to 80 (no SSL) ports.

9. **Spam bot's attack :-**Sends out junk mail or uploads spam to forums or social media web sites. Only port 25 is targeted by using the packets (no SSL). The distinction of the attack is that there will be a higher than usual. multiple SMTP connections made from a same IP address.

10. **Reaper Worm attack :-** After the IP is passed to the exploit process, it starts the final stage of scanning. The ports 81, 82, 83, 84, 88, 1080, 3000, 3749, 8001, 8060, 8080, 8081, 8090, 8443, 8880, and 10,000 are the targets of the Reaper attacks. If a package just has a TCP stream and no ICMP, UDP, or ICMP6 protocols, then it is considered an attack.

11. **Port Scanning/Spread attack :-** looks for a port that is active to exploit a known security flaw by sending client requests to a few server port addresses. A host and one or more other hosts established an unusually high number of connections as a result of the following: One port, several addresses; one port, many addresses. Breaking up packets

12. **Packet fragmentation attack :-** DOS attack wherein the attacker uses datagram fragmentation to overload a network.

Attack Type	**NO. of Flows**	**NO. of Attacks (Flows)**
Smurf	3,994,426	59,479
ICMP flood	3,863,655	11,628
UDP flood	606,814	59,479
TCP SYN flood	14,608,678	3,725,838
HTTP flood	3,963,168	22,959
LAND attack	3,569,838	52,417
Blaster worm	2,858,573	24,291
Code red worm	5,082,952	1,255,702
Spam bot's detection	1,153,020	747
Reaper worms	4,377,656	1176
Scanning/spread	6687	6232
Packet fragmentation attack	1,244,866	477
TOTAL	**45,330,333**	**5,328,934**

Table 3.1: Distribution of attacks in the data set.

	Dataset label	Number of Records selected
1	None	1,285,338
2	tcp_syc_f	205811
3	tcp_red_w	162753
4	icmp_smf	15411
5	udp_f	12064
6	tcp_land	6820
7	tcp_w32_w	3194
8	icmp_f	2994
9	http_f	2941
10	tcp_udp_win_p	799
11	udp_reaper_w	170
12	smtp_b	108
13	udp_0	75
14	**Total record**	**1,698,478**

Table 3.2: Distribution of selected dataset for this study.

3.3.2 Data Processing

The fundamental stages cleansing of data and preparation include the identification of approaches for dealing with missing data fields, acquisition of necessary information to model , elimination of noise or outliers where necessary, and considering temporal information and known alterations.To guarantee that a good resolution is reached, preprocessing entails transforming the raw dataset into a format appropriate for analysis and assessment.

Upon analyzing the LITNET-2020 dataset, It was disclosed that some features within it had an only unique value that included ”fwd, opkt, and obyt,” while others like the timestamp feature were not used. Specifically, Inside this investigation, the timestamp feature which represented the time of recording network traffic information was somewhat ignored. Additionally, address related information such as destination and source IP and port numbers which are distinct features and could not be used in attack detection. And also Categorical features were then processed further by encoding labels to to eliminate them. The important features were selected using K-best .the result of k_best feature selection illustrated in figure 3.2

3.3.2.1 Feature selection method

The K_Best approach for selecting features is an important element that serves to optimize our model performance because it systematically detects the k-best features within the data set given. To elevate interpret ability and reduce over fitting With this method,the data space's dimension count is deliberately reduced. With Select K-Best, functions are carefully evaluated thru the F_Classif scoring feature, which is an utility of the ANOVA F_value between label/outputs and features/inputs. Using F_Classif because the scoring characteristic. Afterwards applying the fit transform method makes the feature selector truly matching with the data where it considers just the best features by using the ANOVA F_value scores that they have. This detailed approach greatly enhances prediction accuracy [31].

NetFlow datasets often contain inconsistent feature attributes classified into six categories: flow, basic, content, time, additional generated and label features. Nevertheless, packets capture also gathers irrelevant and repetitive details. Eliminating unnecessary data yields more unbiased detection methods which are better on average. Consequently, unneeded elements that contain a single unique value together with the non-used Timestamp column have been dropped out while leaving out The 28 best features and further checked for feature collinearity.

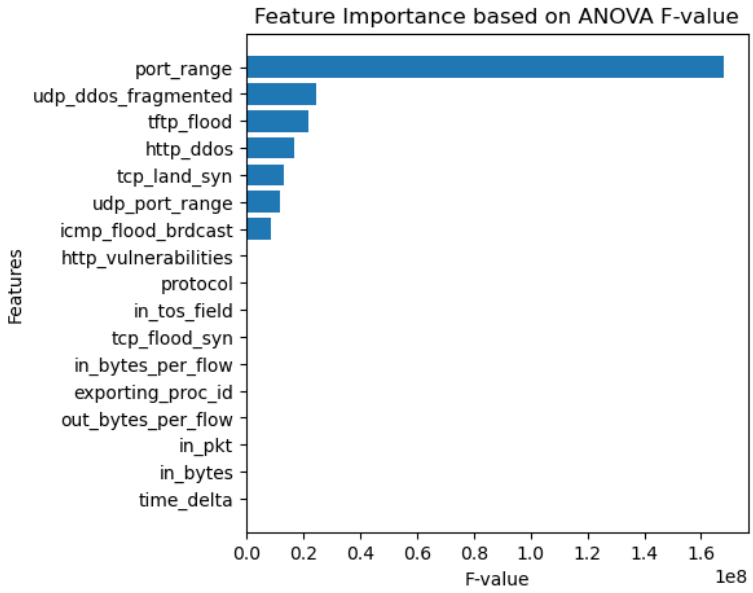

Figure 3.1: K_best Feaatures for LITNET202 Dataset

3.3.2.2 Handling the imbalanced-labelled data

Class imbalance is a ML issue where the classes are not evenly represented in the data. While training machine learning models, this might lead to problems since these models might be biased towards the class which is more common. It is more likely for the model to detect and predict the majority class if there are more samples of one class than the other. Consequently, when applying the model to analyze well-distributed datasets it may come up with wrong conclusions [21]. LITNET-2020 dataset suffers from imbalance problem in class distribution, where the wide variety of normal (benign) flow reaches 3/4 of the dataset's size. Figure 3.3 illustrates the quantity of benign and malicious traffic inside the classification class earlier than handling the imbalanced labelled data the use of the oversampling technique.

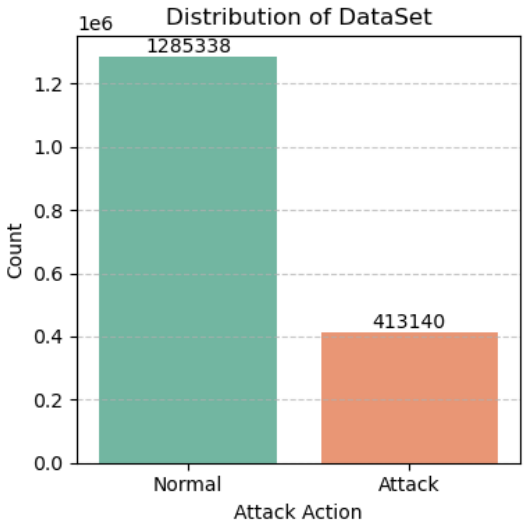

Figure 3.2: Imbalanced dataset.

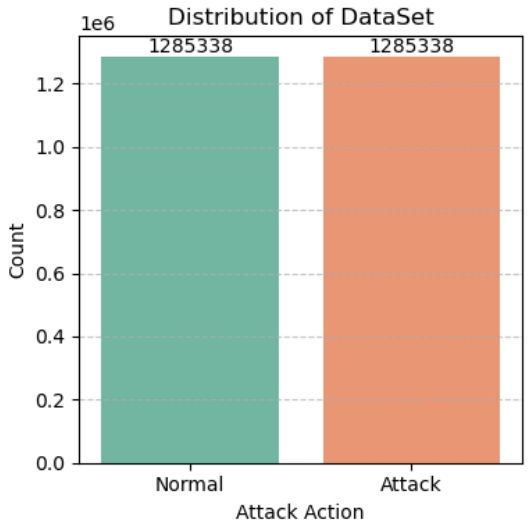

Figure 3.3: Balanced dataset

Figure 3.4: Graph for imbalanced and balanced Dataset

To deal with this issue, SMOTE [59] over-sampling methods was implemented. SMOTE is a way of oversampling, in which artificial samples are produced for the class of minorities. It avoids overfitting due to random oversampling. By interpolating close by high quality examples, we targeted on the function space to create new examples. Figure 3.4 displays the quantity of benign objects and their malicious counterparts in a classification class after processing imbalanced labeled dataset.

3.3.3 Model architecture

The model architecture is discussed in this chapter, Besides the implementation of the proposed Ensemble deep learning IDS. Ensemble DL, incorporating both RNN and LSTM architectures, presents a formidable solution to enhance predictive capabilities in sequential data analysis [60]. RNNs excel in capturing sequential dependencies by retaining data from earlier inputs through recurrent connections [61]. Nevertheless, the vanishing gradient problem frequently causes them to struggle when managing long-range dependencies [62]. Conversely, LSTMs, a variant of RNNs, mitigate this issue by integrating memory cells and gating mechanisms, allowing for the selective retention or forgetting of information over extended sequences [63]. By combining RNNs and LSTMs in an ensemble framework, the strengths of both architectures can be harnessed to optimize learning representations, reduce overfitting, and enhance model robustness [64].

The use of various ML models to solve problems and make data-driven decisions has become the most significant topic. Three main approaches are commonly employed that are widely utilized to create ML models. The first is that only one ML method (supervised or unsupervised learning) is applied. The other approach is hybrid, utilizing learning algorithms that are both method . The intention behind this tactic is to have both algorithms complement each other in order to increase the model's performance on a certain job. The third strategy is known as ensemble learning, and Several ML algorithms are used in it to form an ML model; in our work, we employed ensemble learning algorithms[65] .

RNN have recollections of the past and make decisions based totally on statistics They have gained knowledge from the prior. RNNs get their call from the truth that they do the same mission for every element in a sequence. An RNN's output depends on the calculations performed prior to it, also demonstrating that it holds "memory" that stores information about the calculations made up to that point. RNN models are frequently utilized in fields like NLP and speech recognition. RNN are more intriguing because we can work with vector sequences, which include both input and output sequences.

LSTM regarded as the updated form of a typical RNN model developed for the simpler capture

of long-term dependencies that exist within sequences.A typical RNN operates by activating the hidden state using short-term memory. which is influenced by other local activation's closest to it,in that it is influenced more by the nearest local activation's Long-term memory. Nevertheless, implies that the calculations that take place over entire long sequences influence the network weights. It was therefore necessary to redesign the RNN such that it would have an activation nation that could serve as weights and retain data over long distances, for this reason the name "Long Short-Term Memory" [46]

Ensemble learning leverages diverse representations learned by individual models, potentially leading to superior generalization and performance on unseen data [66]. Moreover, ensemble models are inherently more resilient to noise and outliers, as they integrate predictions from multiple sources to make a choice in the end [67]. To employ this approach, multiple RNN and LSTM models are independently employed the same dataset for training, and their predictions are combined using techniques such as averaging or weighted aggregation [7]. Alternatively, the outputs of individual models can be fed into a meta-learner, such as another neural network, to learn an optimal combination of predictions [15]. This ensemble strategy demonstrates promising potential to advance predictive analytics in various domains, including NLP, time series forecasting, and medical diagnostics.

Our aim is to establish a DL model that is a combined of individual models, which are more accurate, have less false alarms, and capable of sniffing and identifying unexpected attacks. The body of literature suggests that no single algorithm can capture all possible anomalies with high efficiency. A certain algorithm may be good at detecting one type of anomaly but fail in identifying other sorts of abnormalities. This is why the ensemble method turned into proposed. Since our number one goal from the start has been to boost detection capabilities while decreasing false rate.

3.3.3.1 Parameter setting

We examine the LSTM and RNN network's performance on the LITNET2020 dataset in order to calculate the values of the parameters of the chosen methods. The weights of the both network were then optimized using the hybrid RNN algorithms, and lastly, we assessed how well the suggested algorithm Ensemble learning performed in binary classification (normal, anoma-

lous) and 12-category classification.

The selection of parameters in this study was driven by extensive experimentation to increase the performance of the RNN model for intrusion detection.Numerous configurations were employed, tuning parameters such as the learning rate, batch size, hidden units as well as the number of epochs. The aim was to identify the configuration that provided the most consistent correct identification of attacks in both the majority and minority classes.After conducting multiple experiments with several parameter combinations, the chosen settings offered the maximum accuracy rates across several datasets. The last configuration illustrated in Table 3.3

Table 3.3 show the parameter setting for the model.

Algorithm	**LSTM**		**Simple RNN**		**Ensemble Model**	
Parameter	Binary	Multi	Binary	Multi	Binary	Multi
Optimizer	Adam	Adam	Adam	Adam	Adam	Adam
Learning Rate	0.001	0.001	0.001	0.001	0.001	0.001
Input Layer	40	56	32	48	32	64
Hidden Node 1	32	48	24	32	24	48
Hidden Node 2	16	32	16	28	16	32
Output Layer	1	12	1	12	1	12
L1_L2 Regularization	0.001	0.001	0.011	0.011	0.011	0.011
Drop Out	0.2	0.2	0.5	0.5	0.5	0.2
Batch Size	286	196	128	196	128	128
Epochs	20	25	20	25	20	25
Loss Function	Cross Entropy		Cross Entropy		Cross Entropy	
Output Layer Activation	sigmoid	Softmax	sigmoid	Softmax	sigmoid	Softmax

Table 3.3: Parameter Setting for the Model

These specific parameter settings were selected because they were the most effective in improving detection rate, especially for the minority class attacks which are important in intrusion detection. The conclusion of the work is that the ultimate parameters proved to be the most efficient for the performance of the IDS.

3.3.3.2 Training and testing RNN models

The several RNN models (simple RNN, LSTM) were trained and tested using the LITNET-2020 dataset. Recurrent neural networks are composed of input units, output units, and hidden units; the hidden unit performs the majority of the work.The RNN model basically consists of a one-way information flow from the input units to the concealed units as well as a synthesis of the one-way information flow from the previous temporal concealment unit to the present timing concealment unit.Hidden units could be viewed as the network's data store, tracking the information from start to finish.

The LITNET-2020 dataset was divided into an 80% training set and a 20% testing and validation, respectively. This particular method was decided upon after a small experiment aiming to establish the most appropriate way to go about splitting data. The results from this survey are given in a summary form using Figure 3.4 shows the division of LITNET-2020 dataset into four different training and testing subsets; all were then evaluated which yielded that set with ratios at 80:20 being the top choice.

Traffic Type	**Traffic Type**	**Training 80%**	**Testing 20%**
Normal	1,285,338	1,028,270.4	257,067.6
Attack	413,140	330,512	82,628
Total	1,698,478	1,358,782.4	339,695.6

Table 3.4: STATISTICS OF THE LITNET-2020 DATASET

3.3.4 Measurements and Evaluation

The efficiency of a model can be measured using different metrics. The metrics include the accuracy, precision and recall, which can be calculated using a confusion matrix [68]. A confusion matrix is known as a techniques used for measuring the effectiveness of Machine Leaming classification and it is also a comparison summary of the predicted and actual results in classification problems. The confusion matrix is in the form of a table with four different variations of the expected values and actual values as shown in Table 3.5.

		Actual Values	
		Negative	Positive
predicted values	Negative	True Negative (TN)	False Positive (TP)
	Positive	False Negative (FN)	True Positive (TP)

Table 3.5: Confusion matrix

Additionally, the confusion matrix is comprised of True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). TP relates to the quantity of malicious records which are correctly classified, TN relates to the quantity of non malicious records which are correctly classified, FP relates to the quantity of non malicious records which are incorrectly classified and FN relates to the quantity of malicious records which are incorrectly classified as not malicious [24]. The comparison summary provided by the confusion matrix is greatly important as it helps to determine the model's performance after it is trained. The confusion matrix values are utilised to determine the Accuracy, Precision,Recall and F-measure metrics, that are employed in order to assess model performance. The evaluation metrics are further explained in sub-sections 3.3.4.1

3.3.4.1 Accuracy

The model's accuracy is determined by dividing the number of correctly classified dataset samples by the total number of samples in the dataset [37]. The accuracy metric provides the percentage of true detection over the total traffic records, calculated using equation (3.1).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{3.1}$$

Accuracy is regarded to be critical in classification and the aim should be to achieve the highest accuracy **liu2016learning**. In addition,[35] also assert that accuracy is the significant performance indicator of an RNN model.

3.3.4.2 Precision

Precision is a metric which shows how many records, predicted by a model as intrusions, are actual intrusions. The precision is the ratio of true TP over the sum of TP and FP records [37] and it is calculated using equation (3.2).

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

3.3.4.3 Recall

Recall is the number of attack detection which are correct and it is also referred to as the True Positive Rate. It is the proportion of TP records to the total of FN and TP records. [37]. The recall is calculated using equation (3.3).

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

3.3.4.4 F-Measure

F-Measure is the mean of the Precision and Recall [37] which combines the properties of the recall and precision metrics. F-measure can be calculated using equation (3.4).

$$F1 - score = 2 * \frac{Recall * Precision}{Recall + Precision} \tag{3.4}$$

When both the Precision and Recall reaches 100 %, the F-Measure is knowm to be at maximum, which means that the classifier did not get any false alarms and detected all of the attacks.

Accuracy is the most common model evaluation metric, but it can be deceptive if the data are imbalanced. In such instances, it is needed to consider using other evaluation metrics, in addition to the accuracy metric. Even with the use of other matrices, a study by [47] suggests that the imbalanced data can cause the evaluation metrics to disclose more about the class the distribution instead of the actual output of models when the data are imbalanced. Therefore, when selecting a model, a combination of measures should be considered instead of only depending on one measure.

3.4 Data Analysis

Applying statistical methods to data allows for its methodical organization, examination, and description[69]. For this study, quantitative data analysis was carried out. The quantitative analysis involves numbers and can use mathematical operations for the properties of the data [27]. Quantitative data analysis was carried out using the F-measure, Precision, Accuracy and Recall metrics.

CHAPTER 4

Experiment and Analysis

4.1 EXPERIMENT

4.1.1 overview

We described the design of the suggested ensemble IDS in this Chapter. Different components of the planned ensemble IDS are outlined, along with their significance and the strategies to be used in their construction. The architecture is presented in this chapter, along with the algorithms that have been proposed.

4.1.2 Experimental Setup

This thesis used a recurrent neural network (RNN) algorithm LITNET2020 dataset as the experimental setup. The experiment was carried out through key steps for use. Firstly, the LITNET2020 dataset would be preprocessed which involves data cleaning, normalization and feature engineering to make sure it is suitable for RNN model input. Then, to assess the RNN algorithm's effectiveness, this dataset could be divided into training set, and testing set. The number of layers, units, activation functions and other hyper parameters was used to specify the RNN model's architecture. Next, the training of the RNN model would begin using the training set.In order to prevent overfitting, its performances would also be tracked on the validation set. Since various experiments were conducted in optimizing performance of the model's performance, including tuning hyper-parameters adjusting architecture, there are also regularization techniques and dropout methods that were employed during this process. Finally, the trained RNN model can be assessed by its ability of generalization on testing set while accuracy, precision recall, F1 –score are all calculated to judge how effective suggested approach works.

4.1.2.1 Tools Used

Python was utilized for the experimental tools since it is an adaptable and simple language to employ for implementing machine learning algorithms. Python modules utilized in this thesis are as follows Pandas and NumPy , for data preprocessing and manipulation, Scikit-

learn libraries for dataset splitting and evaluation metrics, TensorFlow or Keras for building and training the RNN model and Matplotlib for data visualization. Another indispensable tool was the environment for coding all these tools that was Jupyter Notebook, which provides a link between all these tools as an interactive editorial system. As for the hardware, the test was carried out on a laptop with an Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz 3.19 GHz, 8GB RAM and a Windows 10 Pro operating system.

1. **TensorFlow2.0**

   Google published Tensor Flow, an open-source deep learning library, in November 2015. It has Python and C++ APIs. Although users may also be working with computational primitive wrappers like matrix operations, element-wise math operators, and looping control, it still offers a lot of abstraction power. Given data flow processing and dependencies, networks are viewed by Tensor Flow as a directed graph of nodes. With the intention of creating deep neural network classifiers, the Keras package uses Tensor Flow as its back end.

2. **Keras**

   Tensor Flow or Theano can be utilized as a back end with the Python deep learning tool Keras. Its main objective is to advance and simplify the DL model implementation for research and development. The code in this experiment, which is developed in Python 3, can run on both GPUs and CPUs. Deep neural network classifiers for the suggested GAN and WGAN IDS models are developed using Keras

3. **Scikit-Learn**

   The Python machine learning toolkit Scikit-learn includes advanced ML techniques for both supervised and unsupervised tasks. Providing a high-level, general-purpose language that is understandable by non-machine learning experts is its main objective. It's easy to use, highly efficient, and has comprehensive information. Performance metrics are computed using Scikit-learn to assess the discriminator network's classification capabilities.

4. **Activation Function** An ANN can be equipped with an activation function to aid in the network's ability to recognize intricate patterns in the data. In contrast, the activation function determines what should be fired to the subsequent neuron at the conclusion of

a neuron-based model, just like in our own brains. Additionally, Each neuron's output is normalized to fall between -1 and +1 via activation functions. [70].

(a) **Leaky ReLU:** permits a tiny gradient signal to pass for negative values. It strengthens the gradients flowing into the Generator from the Discriminator as a result. Instead of passing a gradient, it passes a small negative gradient (slope) of 0. In this experiment, we used the generator and discriminator network at every layer except the output layer.

(b) **Sigmoid:** The Discriminator's output is a single integer in the range of 0 to 1, that is, the likelihood that the input data is fraudulent, is many ways similar to previous binary classification models i.e. generated. In most cases, this approach is applied to binary classification issues. . So we have used it at output layer of Discriminator network to classify normal and attack.

(c) **Softmax** Softmax is a activation function that is applied to multi-class classification issues using neural networks to get probabilities from score (raw predication). The purpose of exponentiating the scores and fractionalizing them by the total of all other is to make those probabilities range between 0 an 1, with their sum adding up to 1 After acquiring our calculated traits, the activation function enables us to (usually soft maximum), these become normalized in order that be understood as a probability output. A value closer to 1 means it is more certain that element is of this category. Moreover, one of the key properties of softmax function is that it's differentiable and as derivative can be quite easily used for training neural predictors by employing gradient optimization algorithm called back propagation[34].

5. **Optimizer**

Adam:- Owing to its ability to self-tune and produce good results in numerous applications, Adaptive Moment Estimation is a popular variation of gradient descent. For the purpose of minimizing overfitting, weight decay is set at 0.005, and Adam is utilized as an optimizer to expedite convergence. When training, it performs better, reaching a global minimum more quickly and consistently[71].

4.1.2.2 Sequence of Steps

The procedures involved in loading and preprocessing are explained in full in the techniques section on LITNET2020 dataset, including data cleaning, imputation, and feature scaling or normalization are performed which is explained in detail in the methodology section. After that, we finally train the ensemble model by using the selected features so far obtained via the selection methods. Finally, the model's performance is evaluated Regarding precision and correctness among others including F-measure.

4.2 Analysis

The investigations' findings and their analysis of the selected datasets are presented in this section. We carried out two primary studies to Assess the practicality of the suggested approach. The first experiment investigates the proposed algorithm in binary classification (normal or malicious traffic), while the latter is assessed using multi-class classification.

4.2.1 Performance Evaluation Result for binary classification

The aim of this examination is to verify how good is the new RNN at detecting intrusions for the two categories of network data which are normal and abnormal (binary classification) and multi class classification . LITNET202 Datasets were employed in this endeavor.

In the study's findings, The models' functionality was assessed based on their accuracy, precision, detection rate, and true positive rate (TP), which is a representation of misses or poor predictions, recall as well as false alarm rate or false positive rate (FP) for an individual attack type within the dataset. The models need to record these evaluation criteria during ranking since they determine the most performing models in a given area [72].The Table provides the experiment findings 5.1 and 5.2 This show how well the model works with both balanced and unbalanced datasets.Simple RNN and LSTM are the two RNN algorithms that are employed to evaluate how well the proposed Ensemble model performs, four (4) measurement metrics are used, They are: f-score Precision, Recall and Accuracy.

The evaluation result, As indicated in Table 5.1 and 5.2 , All algorithms demonstrate superior detection performance both imbalanced and balanced dataset. Simple RNN model obtained an

Method	Accuracy	Precision	F1-Score	Recall
Simple RNN	97.894 %	97.943 %	97.863 %	97.894 %
LSTM	98.0 %	98.1 %	98.0 %	98.0 %
Ensemble Model	97.951 %	98.005 %	97.919 %	97.951 %

Table 4.1: Performance Comparison of the Imbalanced dataset

Method	Accuracy	Precision	F1-Score	Recall
Simple RNN	97.831 %	97.921 %	97.83 %	97.831 %
LSTM	99.0 %	99.0 %	99.0 %	99.0 %
Ensemble Model	97.994 %	98.023 %	97.994 %	97.994 %

Table 4.2: Performance Comparison of the balanced dataset

accuracy score of 97.894%, with a recall score of 97.894%, precision 97.943% and F-score 97.863 % using Imbalance dataset, 97.831% accuracy score with 97.831% recall, precision 97.921% and F-score 97.83 % score for balanced dataset. The accuracy score of the LSTM model was 98.0%, with a recall score of 98.0%, precision 98.1% and F-score 90.0 % using Imbalance dataset, 99.0% accuracy score with 99.0% recall, precision 99.0% and F-score 99.0 % score for balanced dataset. and The accuracy score of the ensemble model was 97.951%, with a recall score of 97.951%, precision 98.001% and F-score 97.919 % using Imbalance dataset, 97.994% accuracy score with 97.994% recall, precision 98.023% and F-score 97.994 % score for balanced dataset.

We can observe that every algorithm worked well on both balanced and unbalanced datasets.despite this,Data balancing strategies enhance each of the three models' overall performance across all performance metrics.Among those, the LSTM score accuracy was 98.0% in the balanced dataset and 99.0% in the imbalanced dataset.

The confusion matrix for all implemented algorithms in this experiment is shown in Figure 5.4 There has been a growth in true positives (TP) and true negative values, while false positives and false negatives have decreased from figure 5.1 - 5.3 . False Positives (FP) and False Negatives (FN) for Ensemble model 6962 and 0 in imbalanced dataset Its size is less than both of them, which is a crucial factor for IDSs.

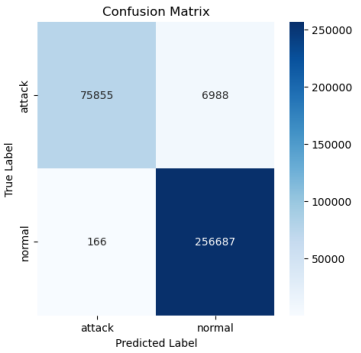

Figure 4.1: Confusion matrices for Simple-RNN

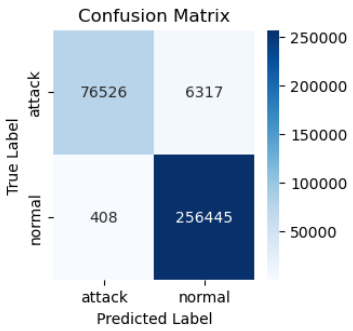

Figure 4.2: Confusion matrices for LSTM

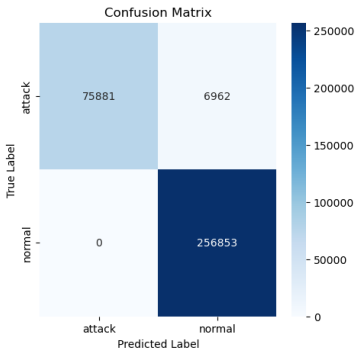

Figure 4.3: Confusion matrices for Ensemble Model

Figure 4.4: Confusion matrices for Imbalanced Dataset

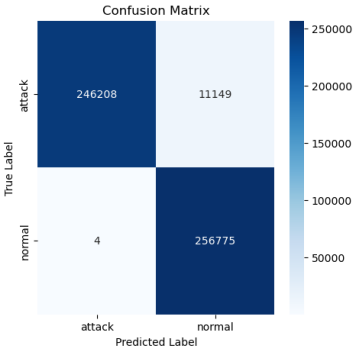

Figure 4.5: Confusion matrices for Simple-RNN

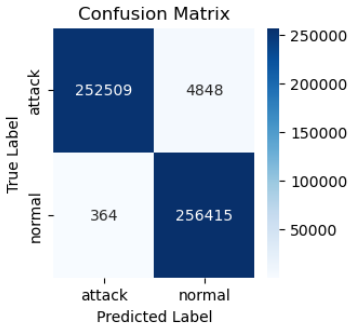

Figure 4.6: Confusion matrices for LSTM

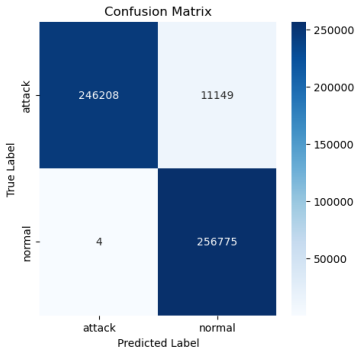

Figure 4.7: Confusion matrices for Ensemble model

Figure 4.8: Confusion matrices for Balanced Dataset

As shown in the resulting confusion matrix for balanced Data set exhibited in figure 5.8 , the three model have better result comparing with imbalanced dataset .that indicate that when the dataset are balanced using SMOTE data balancing techniques improved IDS performance. and the Ensemble model has classified well for binary IDS according to the confusion matrices.

An additional factor is considered for both validation loss and training loss, this graph are important tools For monitoring a ML model's overall performance during training, the training loss needs to be on the decrease to show that learning is taking place and performance is getting better validation loss must show the same trend, though it is not supposed to reduce by

the same rate as train loss. Additionally, it help to detecting over fitting over-fitting detection you might determine that some over-fitting towards training data starts occurring by comparing the training and the validation loss graphics. If the training loss keeps decreasing but validation loss rises, then it's a sign that your model has grown into specialized on one type of information or another not being able to create any knowledge for unseen datasets.

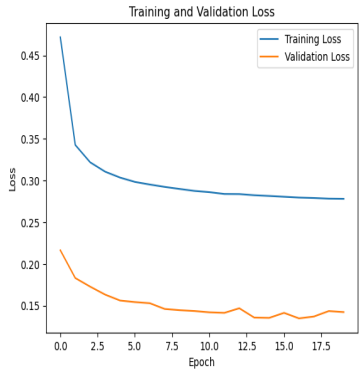

Figure 4.9: Training and value loss Graph for Simple_RNN

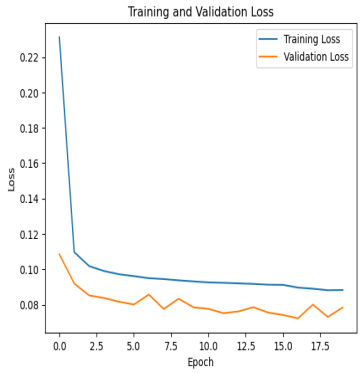

Figure 4.10: Training and value loss Graph for LSTM

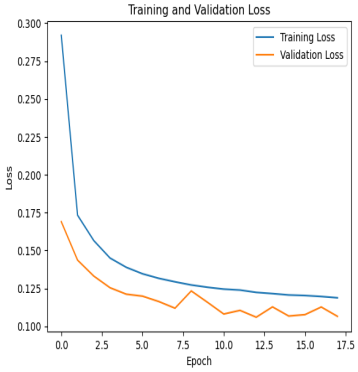

Figure 4.11: Training and value loss Graph for Ensemble model

Figure 4.12: Training and value loss Graph for Imbalanced dataset

The results of this experiment were plotted in Figure 5.9 - 5.11. training loss graph of the three model result shows a progression in model performance, with each subsequent model achieving better training loss reduction. but have graph 2 in figure 5.10 indicates an improved model that continuously achieves decrease training loss compared to graph 1 in figure 5.9. the last graph in figure 5.11 displays the best-performing model, with the lowest training loss among the three, highlighting significant improvements over both Graph 1 and Graph 2 for imbalanced dataset.

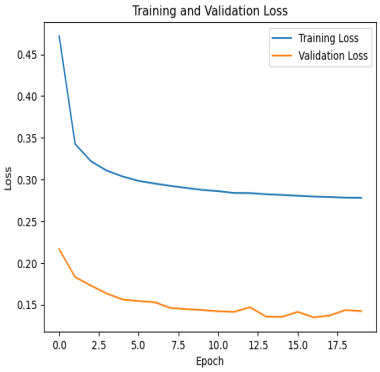

Figure 4.13: Training and value loss Graph for Simple-RNN

Figure 4.14: Training and value loss Graph for LSTM

Figure 4.15: Training and value loss Graph for Ensemble Model

Figure 4.16: Training and value loss Graph for Balanced Dataset

The training loss graph shows that in an imbalanced dataset, the three models exhibit variations in performance. However, when applied to a balanced dataset, the models demonstrate significantly improved training loss curves as illustrated in figure 5.13 - 5.15. Overall, the clear conclusion from this evaluation is that balancing the dataset can notably improve the general effectiveness of RNN.

In general, when comparing the ensemble method with single RNN and LSTM algorithms , ensemble method enhance was able to reduce the FP rate and provide good training loss graph performance on both balanced and imbalanced data found in LITNET2020 Dataset even if it still needs improving . The proposed ensemble method combines the advantages both RNN and LSTM detection approaches which, able to control over fitting problem.Furthermore, all of the suggested models' effects from data resampling are noted, and their respective performances are then contrasted. When it comes to accuracy, precision, recall, and F-score, the LSTM model performs better than the other recommended binary models. Important features are extracted from the inputs by the LSTM model, aiding in the detection of attacks.

4.2.2 Performance Evaluation Result for multi- class classification

This study aims to assess how well the advised RNN model for IDS implements when network traffic is categorized into different categories of attacks (i.e., multiclassification) where there are 13 classes of data Normal and 12 types of attacks. As shown in figure 5.17

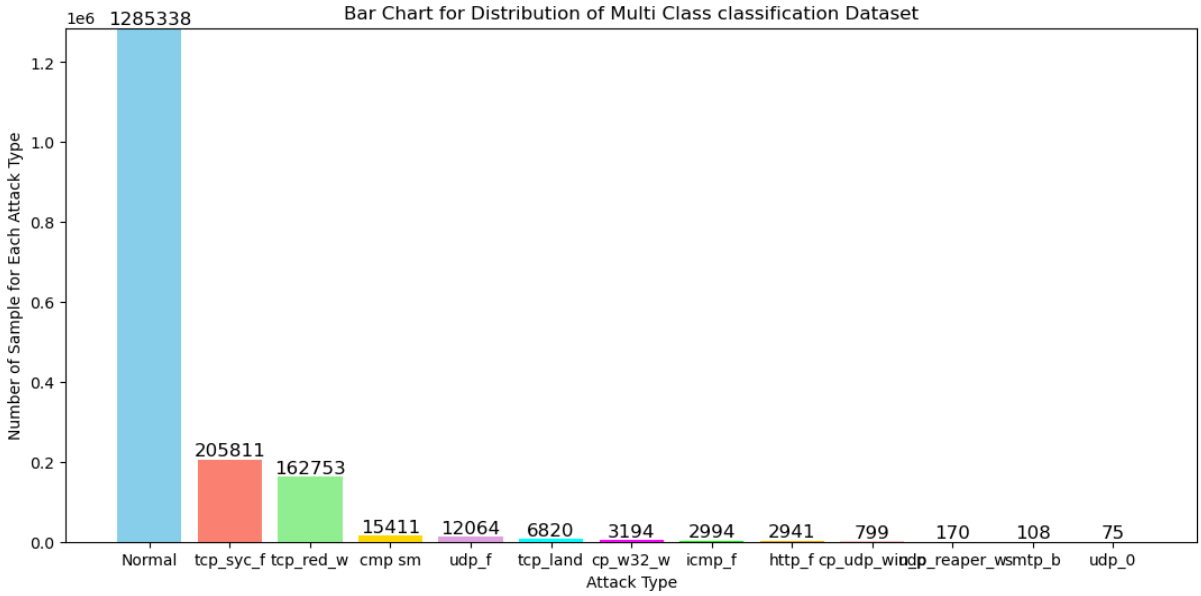

Figure 4.17: Multi Class Distribution Graph

Using the LITNET-2020, We additionally examined and compared the enseble model's accuracy using an LSTM and a simple RNN, and the amount of iterations needed. The results of this experiment were plotted in Table 5.3 This figure indicates that the suggested ensemble moded algorithm achieved a higher accuracy rate of 99.981%) In comparison with the simple RNN's accuracy 98.29%. LSTM results were achieved 99.638% accuracy.

Method	**Accuracy**	**Precision**	**F1-Score**	**Recall**
Simple RNN	98.29 %	96.966 %	97.56 %	98.29 %
LSTM	99.638 %	99.451 %	99.507 %	99.638 %
Ensemble Model	99.981 %	99.965 %	99.973 %	99.981 %

Table 4.3: Multi Class Classification Performance

Additionally, the assessment metrics for the ensemble, LSTM, and simple RNN models in the multi-class classification scenario are displayed in Tables 5.18, 5.19 and 5.20 and , respectively. Overall, the three models fared well in most circumstances across all accuracy, precision, recall, and F-score criteria.But among the multi-class models, the ensemble model is the most .

confusion matrix is plotted in Figure 5.18 it is shown that the simple RNN model successfully detects icmp_smf , tcp_land and tcp_syc_f attacks with good accuracy, precision and recall it is

	precision	recall	f1-score	support
0	0.000	0.000	0.000	57400.00
1	0.000	0.000	0.000	58700.00
2	53.269	100.000	69.511	311200.00
3	98.814	100.000	99.403	25654700.00
4	0.000	0.000	0.000	1400.00
5	0.000	0.000	0.000	135700.00
6	100.000	100.000	100.000	3280900.00
7	100.000	100.000	100.000	4141900.00
8	0.000	0.000	0.000	14900.00
9	0.000	0.000	0.000	61500.00
10	0.000	0.000	0.000	2000.00
11	0.000	0.000	0.000	246400.00
12	0.000	0.000	0.000	2900.00
accuracy	98.290	98.290	98.290	98.29
macro avg	27.083	30.769	28.378	33969600.00
weighted avg	96.966	98.290	97.560	33969600.00

Attack Type	Encoded Value
http_f	0
icmp_f	1
icmp_smf	2
Normal	3
tcp_udp_win_p	4
tcp_land	6
tcp_syc_f	7
tcp_red	8
tcp_w32_w	9
udp_0	10
udp_f	11
udep_reaper_w	12

Figure 4.18: Confusion Matrices of Simple-RNN for multi classification

evaluated against current multi-class NIDS in traditional machine learning .

	precision	recall	f1-score	support
0	0.000	0.000	0.000	5.740000e+04
1	72.589	24.361	36.480	5.870000e+04
2	76.349	100.000	86.589	3.112000e+05
3	100.000	100.000	100.000	2.565470e+07
4	0.000	0.000	0.000	1.400000e+03
5	86.488	100.000	92.755	1.357000e+05
6	100.000	100.000	100.000	3.280900e+06
7	100.000	100.000	100.000	4.141900e+06
8	0.000	0.000	0.000	1.490000e+04
9	100.000	100.000	100.000	6.150000e+04
10	0.000	0.000	0.000	2.000000e+03
11	100.000	100.000	100.000	2.464000e+05
12	0.000	0.000	0.000	2.900000e+03
accuracy	99.638	99.638	99.638	9.963800e+01
macro avg	56.571	55.720	55.063	3.396960e+07
weighted avg	99.451	99.638	99.507	3.396960e+07

Attack Type	Encoded Value
http_f	0
icmp_f	1
icmp_smf	2
Normal	3
tcp_udp_win_p	4
tcp_land	6
tcp_syc_f	7
tcp_red	8
tcp_w32_w	9
udp_0	10
udp_f	11
udep_reaper_w	12

Figure 4.19: Confusion Matrices of LSTM for multi classification

confusion matrix is plotted in Figure 5.19 it is shown that the LSTM model successfully detects http_f , icmp_smf , tcp_land , tcp_syc_f ,tcp_w32_w and udp_f attacks with good accuracy, precision and recall .Its performance is compared to recent multi-class NIDS in traditional machine learning . confusion matrix is plotted in Figure 5.20 it is shown that the LSTM model successfully detects http_f , icmp_smf ,icmp_f, tcp_land , tcp_syc_f ,tcp_w32_w and udp_f attacks with good accuracy, precision and recall . Its performance is compared to recent multi-class NIDS in traditional machine learning and single LSTM and RNN. multi-class LSTM and ensemble method able to detect most of the attack type successfully.

	precision	recall	f1-score	support
0	90.110	100.000	94.798	5.740000e+04
1	100.000	100.000	100.000	5.870000e+04
2	100.000	100.000	100.000	3.112000e+05
3	100.000	100.000	100.000	2.565470e+07
4	0.000	0.000	0.000	1.400000e+03
5	100.000	100.000	100.000	1.357000e+05
6	100.000	100.000	100.000	3.280900e+06
7	100.000	100.000	100.000	4.141900e+06
8	100.000	100.000	100.000	1.490000e+04
9	100.000	100.000	100.000	6.150000e+04
10	0.000	0.000	0.000	2.000000e+03
11	100.000	100.000	100.000	2.464000e+05
12	0.000	0.000	0.000	2.900000e+03
accuracy	99.981	99.981	99.981	9.998100e+01
macro avg	76.162	76.923	76.523	3.396960e+07
weighted avg	99.965	99.981	99.973	3.396960e+07

Attack Type	Encoded Value
http_f	0
icmp_f	1
icmp_smf	2
Normal	3
tcp_udp_win_p	4
tcp_land	6
tcp_syc_f	7
tcp_red	8
tcp_w32_w	9
udp_0	10
udp_f	11
udep_reaper_w	12

Figure 4.20: Confusion Matrices of ensemble model for multi classification

Finally we analysis the result of multi class classification using training loss graph as plotted on figure 5.21 , 5.22 and 5.23 . the graph result indicate that train and test loss continuously drop over each epoch but the ensemble model graph curve demonstrate that better since training and test loss are both steadily declining, without the test loss increasing while the training loss keeps decreasing .

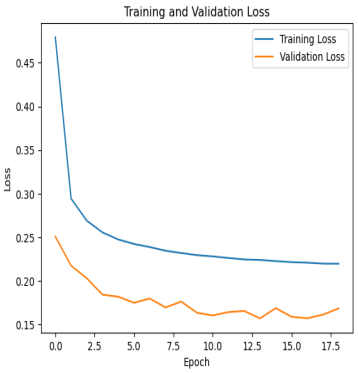

Figure 4.21: Training and value loss Graph for Simple-RNN

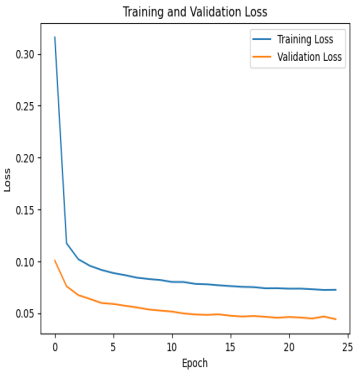

Figure 4.22: Training and value loss Graph for LSTM

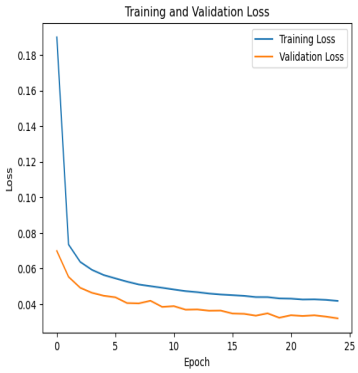

Figure 4.23: Training and value loss Graph for Ensemble model

Figure 4.24: Training and value loss Graph for Multi Class Classification dataset

CHAPTER 5

CONCLUSIONS AND FUTURE WORKS

5.1 CONCLUSIONS

Today's world of communication is extremely complicated and beyond human knowledge due to the increase in the quantity of linked devices and services. Computer networks provide human communication and the integration of systems and services while being dynamic, expanding, and always changing. Hackers have been harming this inter-connected environment by destroying or stealing personal information. As complexity grows, it becomes harder to successfully communicate to human decision making the results of techniques and metrics for tracking networks, classifying traffic, and figuring out malicious or abnormal events. Security specialists require equipment that aid them understand the cause for, and make decisions approximately the facts their analytic systems produce.

In today's data driven world, using DL algorithms as back-end engine is of great help as it automatically detects malicious and normal network traffics hence more support to security professionals.We have examined several RNN based DL algorithms in this paper, and we provide an ensemble model that is built on ensembles NIDS using RNN using Jupiter note book on LITNET-2020 dataset and we demonstrated that it outperforms then others in terms of recall, accuracy, precision, F-measure, and model building time. Experiment results show that feature k-best selection algorithms should be utilized in order to decrease data size dimension.

Out of binary models suggested, the LSTM model has produced the highest accuracy rates, precision, recall values, and F-scores during classification into either normal or attacks, Resampling techniques also show promise in addressing gap that exists within IDS, owing to an uneven distribution of classes. In multiclass classification, the ensemble achieves optimal performance as measured by F-score, recall, accuracy rates, and precision. Furthermore, as the test and training loss graphs demonstrate, the simple RNN model simply experiences overfitting as its complexity is raised. Yet, the overfitting issue is resolved by the ensemble approach.

5.2 Future Works

- The data used for this thesis obtained from publicly available data source, We suggest for further research It is better to assess the model. utilizing real-time traffic data in real environment to categorize and forecast cyber-attacks
- This paper offers an IDS that yields promising results. Nonetheless, The minority class detection rates could be improved, which will permit this work to be extended. You can raise the detection rate even more by utilizing various deep learning architectures, various arrangements of techniques that combine undersampling and oversampling, and various data resampling strategies
- Study on additional features and dataset included and selecting relatively high performance models.
- personal computer used to conduct using this research, but it would be much better if the same research were conducted on a more powerful computer called a having GPU. Especially when working with an enormous quantity of data, such as one has during large-scale feature engineering, it is very important that You fully exploit hardware-accelerated performance that comes with GPU since it enables fast iterations and enough experiments for the development of such strong and signification attributes that may enhance quality of the entire deep learning model.

Bibliography

[1] T.-Y. Kim and S.-B. Cho, "Web traffic anomaly detection using c-lstm neural networks," *Expert Systems with Applications*, vol. 106, pp. 66–76, 2018.

[2] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks," *IEEE communications surveys & tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.

[3] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Ids: A comprehensive review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, 2013.

[4] H. N. Aludhilu, "An ids usingrnn with a real-world dataset," Ph.D. dissertation, University of Namibia, 2020.

[5] A. P. Singh and M. D. Singh, "Analysis of host-based and network-based ids," *IJ Computer Network and Information Security*, vol. 8, pp. 41–47, 2014.

[6] S. Mohammadi, M. H. Sherkat, and M. Jamporazmey, "A taxonomy framework based on itu-tx-805 security architecture for quantitative determination of computer network vulnerabilities," *Security and Communication Networks*, vol. 6, no. 7, pp. 864–880, 2013.

[7] H. Liu and B. Lang, "Ml and dl methods for ids: A survey," *applied sciences*, vol. 9, no. 20, p. 4396, 2019.

[8] D. Jakhar and I. Kaur, "Artificial intelligence, machine learning and deep learning: Definitions and differences," *Clinical and experimental dermatology*, vol. 45, no. 1, pp. 131–132, 2020.

[9] L. Deng, D. Yu, *et al.*, "Dl: Methods and applications," *Foundations and trends® in signal processing*, vol. 7, no. 3–4, pp. 197–387, 2014.

[10] J. Lansky, S. Ali, M. Mohammadi, *et al.*, "Deep learning-based intrusion detection systems: A systematic review," *IEEE Access*, vol. 9, pp. 101 574–101 599, 2021.

[11] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep rnn for ins in sdn-based networks," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, IEEE, 2018, pp. 202–206.

[12] M. A. Trabelsi, "The impact of artificial intelligence on economic development," *Journal of Electronic Business & Digital Economics*, 2024.

[13] L. Haddon, *Information and communication technologies in everyday life: A concise introduction and research guide*. Berg Oxford, 2004.

[14] J. Wang, D. He, A. Castiglione, B. B. Gupta, M. Karuppiah, and L. Wu, "Pcnn cec: Efficient and privacy-preserving convolutional neural network inference based on cloud-edge-client collaboration," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 5, pp. 2906–2923, 2022.

[15] A. Divekar, M. Parekh, V. Savla, R. Mishra, and M. Shirole, "Benchmarking datasets for anomaly-based nids: Kdd cup 99 alternatives," in *2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS)*, IEEE, 2018, pp. 1–8.

[16] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1–22, 2019.

[17] I. S. Atawodi, "A machine learning approach to network intrusion detection system using k nearest neighbor and random forest," 2019.

[18] F. Laghrissi, S. Douzi, K. Douzi, and B. Hssina, "Intrusion detection systems using long short-term memory (lstm)," *Journal of Big Data*, vol. 8, no. 1, p. 65, 2021.

[19] P. S. Muhuri, P. Chatterjee, X. Yuan, K. Roy, and A. Esterline, "Using a lstm-rnn to classify network attacks," *Information*, vol. 11, no. 5, p. 243, 2020.

[20] P. Bedi, N. Gupta, and V. Jindal, "I-siamids: An improved siam-ids for handling class imbalance in network-based intrusion detection systems," *Applied Intelligence*, vol. 51, no. 2, pp. 1133–1151, 2021.

[21] H. M. Elmasry, A. E. Khedr, and H. M. Abdelkader, "Enhancing the ids efficiency using a partitioning-based recursive feature elimination in big cloud environment," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 1, 2023.

[22] K. AYELE, "Anomaly-based ids using gans," Ph.D. dissertation, St. Mary's University, 2021.

[23] A. Adeyemo, "Design of an ids and an ips for the eiu cybersecurity laboratory," 2016.

[24] N. Sainis, D. Srivastava, and R. Singh, "Classification of various dataset for ids," *International Journal of Emerging Technology and Advanced Engineering*, vol. 8, 2018.

[25] K. Rajasekaran and K. Nirmala, "Classification and importance of ids," *International Journal of Computer Science and Information Security*, vol. 10, no. 8, p. 44, 2012.

[26] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based nids: Techniques, systems and challenges," *computers & security*, vol. 28, no. 1-2, pp. 18–28, 2009.

[27] A. Patil, A. Laturkar, S. Athawale, R. Takale, and P. Tathawade, "A multilevel system to mitigate ddos, brute force and sql injection attack for cloud security," in *2017 International Conference on Information, Communication, Instrumentation and Control (ICICIC)*, IEEE, 2017, pp. 1–7.

[28] S. Al-Eidi, O. Darwish, Y. Chen, and G. Husari, "Snapcatch: Automatic detection of covert timing channels using image processing and machine learning," *IEEE Access*, vol. 9, pp. 177–191, 2020.

[29] Z. Cai, Z. Wang, K. Zheng, and J. Cao, "A distributed tcam coprocessor architecture for integrated longest prefix matching, policy filtering, and content filtering," *IEEE Transactions on Computers*, vol. 62, no. 3, pp. 417–427, 2011.

[30] W. Fang, X. Tan, and D. Wilbur, "Application of ids in network safety based on machine learning," *Safety Science*, vol. 124, p. 104 604, 2020.

[31] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[32] R. Wason, "Dl: Evolution and expansion," *Cognitive Systems Research*, vol. 52, pp. 701–708, 2018.

[33] P. Graff, F. Feroz, M. P. Hobson, and A. Lasenby, "Skynet: An efficient and robust neural network training tool for ml in astronomy," *Monthly Notices of the Royal Astronomical Society*, vol. 441, no. 2, pp. 1741–1759, 2014.

[34] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for cnn," *arXiv preprint arXiv:1612.02295*, 2016.

[35] M. Ponkarthika and V. Saraswathy, "Nids using dnn," *Asian Journal of Science and Technology*, vol. 2, no. 2, pp. 665–673, 2018.

[36] K. Pawar, R. S. Jalem, and V. Tiwari, "Stock market price prediction using lstm rnn," in *Emerging trends in expert applications and security*, Springer, 2019, pp. 493–503.

[37] M. Elsherif, M. U. Hassan, A. K. Yetisen, and H. Butt, "Wearable contact lens biosensors for continuous glucose monitoring using smartphones," *ACS nano*, vol. 12, no. 6, pp. 5452–5462, 2018.

[38] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Dl for cyber security ids approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, p. 102 419, 2020.

[39] S. Althubiti, W. Nick, J. Mason, X. Yuan, and A. Esterline, "Applying lstm rnn for ids," in *SouthEastCon 2018*, IEEE, 2018, pp. 1–5.

[40] A. Liu, Y. Su, W. Nie, and M. S. Kankanhalli, "Hierarchical clustering multi-task learning for joint human action grouping and recognition.," 2017.

[41] P. K. Tomar, K. S. Kumar, G. Krishna, K. Soumya, R. K. Ibrahim, and M. B. Alazzam, "Improved detection of cyber-attacks using a bi-directional rnn with lstm deep learning model," in *2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, IEEE, 2023, pp. 2660–2664.

[42] M. K. Hooshmand and D. Hosahalli, "Network anomaly detection using deep learning techniques," *CAAI Transactions on Intelligence Technology*, vol. 7, no. 2, pp. 228–243, 2022.

[43] R. Pramanik, S. Khare, G. Harshvardhan, and M. K. Gourisaria, "A comparative study for depression prediction using ml classification models," in *Advances in Data and Information Sciences*, Springer, 2022, pp. 233–246.

[44] J. Jordan, "Hyperparameter tuning for ml models," *Retrieved from: Jeremy Jordan: https://www. jeremyjordan. me/hyperparameter-tuning*, 2017.

[45] S. ZAVRAK and M. İskefiyeli, "Flow-based ids on software-defined networks: A multivariate time series anomaly detection approach," 2022.

[46] A. A. Awad, A. F. Ali, and T. Gaber, "An improved lstm for ids," *Plos one*, vol. 18, no. 8, e0284795, 2023.

[47] C. Yin, Y. Zhu, J. Fei, and X. He, "A dl approach for ids using rnns," *Ieee Access*, vol. 5, pp. 21 954–21 961, 2017.

[48] R. Vinayakumar, K. Soman, and P. Poornachandran, "Applying convolutional neural network for network intrusion detection," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, IEEE, 2017, pp. 1222–1228.

[49] R. C. Staudemeyer and C. W. Omlin, "Evaluating performance of long short-term memory recurrent neural networks on intrusion detection data," in *Proceedings of the South African institute for computer scientists and information technologists conference*, 2013, pp. 218–224.

[50] R. C. Staudemeyer, "Applying long short-term memory recurrent neural networks to intrusion detection," *South African Computer Journal*, vol. 56, no. 1, pp. 136–154, 2015.

[51] L. Ashiku and C. Dagli, "Network intrusion detection system using deep learning," *Procedia Computer Science*, vol. 185, pp. 239–247, 2021.

[52] S. Amutha, R. Kavitha, R. Srinivasan, and M. Kavitha, "Secure network intrusion detection system using nid-rnn based deep learning," in *2022 International conference on advances in computing, communication and applied informatics (ACCAI)*, IEEE, 2022, pp. 1–5.

[53] S. Gautam, A. Henry, M. Zuhair, M. Rashid, A. R. Javed, and P. K. R. Maddikunta, "A composite approach of intrusion detection systems: Hybrid rnn and correlation-based feature optimization," *Electronics*, vol. 11, no. 21, p. 3529, 2022.

[54] M.-J. Kang and J.-W. Kang, "Ids using dnn for in-vehicle network security," *PloS one*, vol. 11, no. 6, e0155781, 2016.

[55] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and A. V. Vasilakos, "An effective network traffic classification method with unknown flow detection," *IEEE Transactions on Network and Service Management*, vol. 10, no. 2, pp. 133–147, 2013.

[56] P. Chetty, "Importance of research approach in a research," *Research design strategy*, 2016.

[57] R. Kumar, A. Ray, *et al.*, "Selection of material for optimal design using multi-criteria decision making," *Procedia materials science*, vol. 6, pp. 590–596, 2014.

[58] R. Damasevicius, A. Venckauskas, S. Grigaliunas, *et al.*, "Litnet-2020: An annotated real-world network flow dataset for nids," *Electronics*, vol. 9, no. 5, p. 800, 2020.

[59] S. Bagui and K. Li, "Resampling imbalanced data for nids datasets," *Journal of Big Data*, vol. 8, no. 1, p. 6, 2021.

[60] S. S. Ramalingam, J. Vansteenkiste, D. Planchard, *et al.*, "Overall survival with osimertinib in untreated, egfr-mutated advanced nsclc," *New England Journal of Medicine*, vol. 382, no. 1, pp. 41–50, 2020.

[61] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[62] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[63] S. Hochreiter and J. Schmidhuber, "Lstm," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[64] A. Karpathy, "The unreasonable effectiveness of rnns," *Andrej Karpathy blog*, vol. 21, p. 23, 2015.

[65] K. WORKU, "A predictive model of nidss using machine learning approach," Ph.D. dissertation, St. Mary's University, 2023.

[66] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE transactions on pattern analysis and machine intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[67] D. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *Journal of artificial intelligence research*, vol. 11, pp. 169–198, 1999.

[68] B. Ingre and A. Yadav, "Performance analysis of nsl-kdd dataset using ann," in *2015 international conference on signal processing and communication engineering systems*, IEEE, 2018, p. 92.

[69] M. Riva, M. Panzeri, A. Guadagnini, and S. P. Neuman, "Role of model selection criteria in geostatistical inverse estimation of statistical data-and model-parameters," *Water Resources Research*, vol. 47, no. 7, 2011.

[70] A. Kubaji *et al.*, "A new deep learning based object detection system for increasing salesman performance= satış elemanı verimliliği için yeni bir derin öğrenme tabanlı nesne tespit sistemi," 2022.

[71] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[72] A. Karthik, J. Shetty, G. Shobha, and R. Dev, "Implementation of gans in hpcc systems using gnn bundle," *IAES International Journal of Artificial Intelligence*, vol. 10, no. 2, p. 374, 2021.

CHAPTER 6

Appendix

Appendix A: PYTHON CODE USED

```python
# import impotatant library
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from keras.models import Sequential
from keras.layers import SimpleRNN, Dense, Dropout
from keras.regularizers import l1_l2
from keras.callbacks import EarlyStopping
from keras.activations import relu
import matplotlib.pyplot as plt

# Step 1: Load the dataset
dataset = pd.read_csv('corr_dataset.csv')
seed = 124
np.random.seed(seed)

# Step 2: Preprocess the data
X = dataset.drop(columns=['attack_action'])
y = dataset['attack_action']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Reshape the data for RNN input (samples, timesteps, features)
X_train_reshaped = X_train_scaled.reshape((X_train_scaled.shape[0], 1, X_train_scaled.shape[1]))
X_test_reshaped = X_test_scaled.reshape((X_test_scaled.shape[0], 1, X_test_scaled.shape[1]))

# Step 3: Define the Simple RNN model with Dropout and L2 regularization
model = Sequential([
    SimpleRNN(36, return_sequences=True, input_shape=(X_train_reshaped.shape[1], X_train_reshaped.shape[2]),
              activation='relu',
              kernel_regularizer=l2(l2=0.01)),  # L1 and L2 regularization
    Dropout(0.3),  # Add dropout after the first SimpleRNN layer
    SimpleRNN(32, return_sequences=True, activation='relu', kernel_regularizer=l2(l2=0.01)),
    Dropout(0.3),  # Add dropout after the second SimpleRNN layer
    SimpleRNN(28, return_sequences=True, activation='relu', kernel_regularizer=l2(l2=0.01)),
    Dropout(0.3),  # Add dropout after the third SimpleRNN layer
    #SimpleRNN(32, activation='relu', kernel_regularizer=l1_l2(l1=0.01, l2=0.01)),  # Add fourth SimpleRNN layer
    #Dropout(0.5),  # Add dropout after the fourth SimpleRNN layer
    Dense(1, activation='sigmoid')
])

# Step 4: Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Define early stopping to prevent overfitting
early_stopping = EarlyStopping(monitor='val_loss', patience=15, verbose=1)

# Train your model and capture the history
history = model.fit(X_train_reshaped, y_train, epochs=3, batch_size=128 , validation_data=(X_test_reshaped, y_test),
                    callbacks=[early_stopping], verbose=2)

# Step 6: Evaluate the model
loss, accuracy = model.evaluate(X_test_reshaped, y_test)
print("Test Accuracy:", accuracy)
```

Figure 6.1: code used for RNN model

Appendix B: CORRELATION OF INPUT FEATURES

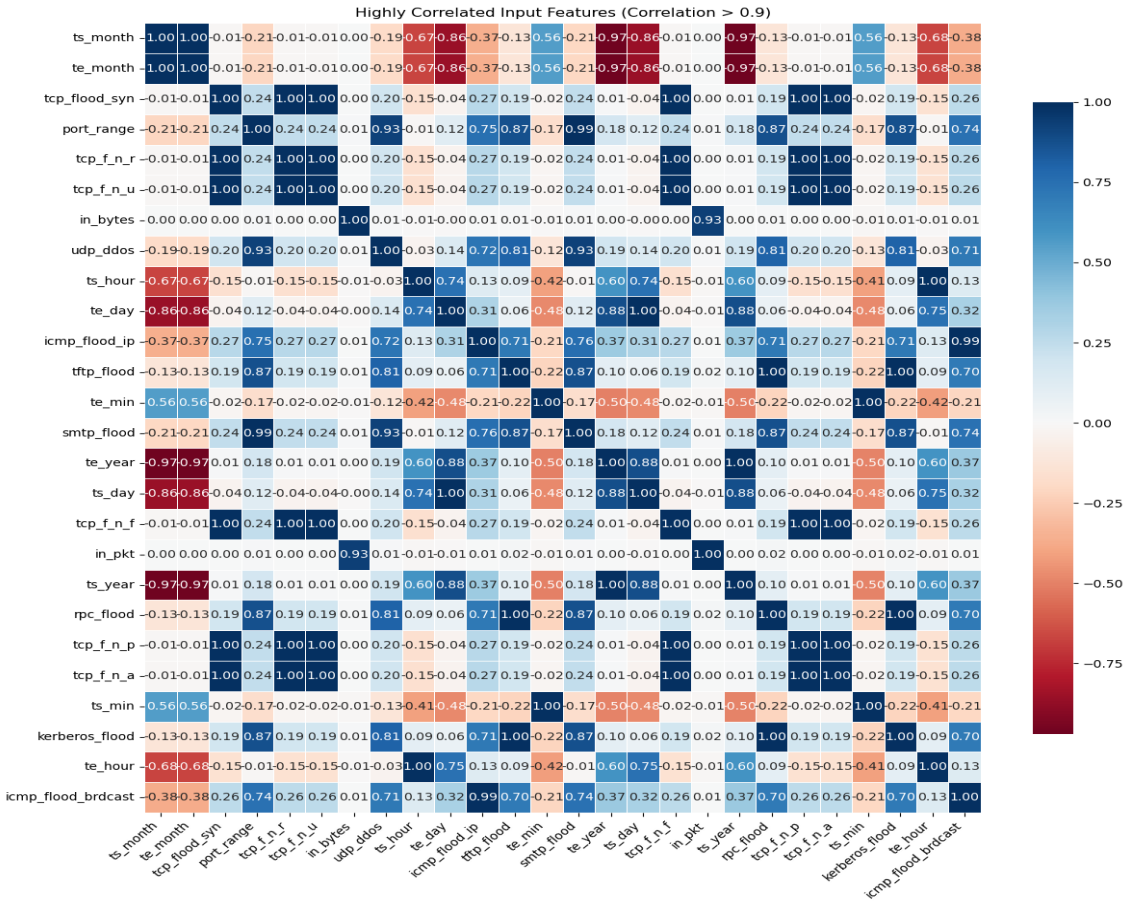

Figure 6.2: input feature correlation result

Appendix C: SAMPLE OUTPUT FOR THE MODEL OVER EPOCH

```
Epoch 18/25
6933/6933 - 42s - loss: 0.0440 - accuracy: 0.9968 - val_loss: 0.0335 - val_accuracy: 0.9998 - 42s/epoch - 6ms/step
Epoch 19/25
6933/6933 - 42s - loss: 0.0440 - accuracy: 0.9967 - val_loss: 0.0348 - val_accuracy: 0.9992 - 42s/epoch - 6ms/step
Epoch 20/25
6933/6933 - 42s - loss: 0.0432 - accuracy: 0.9968 - val_loss: 0.0324 - val_accuracy: 0.9998 - 42s/epoch - 6ms/step
Epoch 21/25
6933/6933 - 42s - loss: 0.0431 - accuracy: 0.9968 - val_loss: 0.0338 - val_accuracy: 0.9996 - 42s/epoch - 6ms/step
Epoch 22/25
6933/6933 - 42s - loss: 0.0426 - accuracy: 0.9969 - val_loss: 0.0334 - val_accuracy: 0.9997 - 42s/epoch - 6ms/step
Epoch 23/25
6933/6933 - 42s - loss: 0.0427 - accuracy: 0.9968 - val_loss: 0.0337 - val_accuracy: 0.9994 - 42s/epoch - 6ms/step
Epoch 24/25
6933/6933 - 43s - loss: 0.0424 - accuracy: 0.9968 - val_loss: 0.0330 - val_accuracy: 0.9998 - 43s/epoch - 6ms/step
Epoch 25/25
6933/6933 - 42s - loss: 0.0418 - accuracy: 0.9969 - val_loss: 0.0320 - val_accuracy: 0.9998 - 42s/epoch - 6ms/step
10616/10616 [==============================] - 21s 2ms/step
10616/10616 [==============================] - 23s 2ms/step - loss: 0.0320 - accuracy: 0.9998
Test Accuracy: 0.999814510345459
Confusion Matrix:
```

Figure 6.3: sample result of the model

Appendix D: Multi Class Confusion matrices for ensemble method

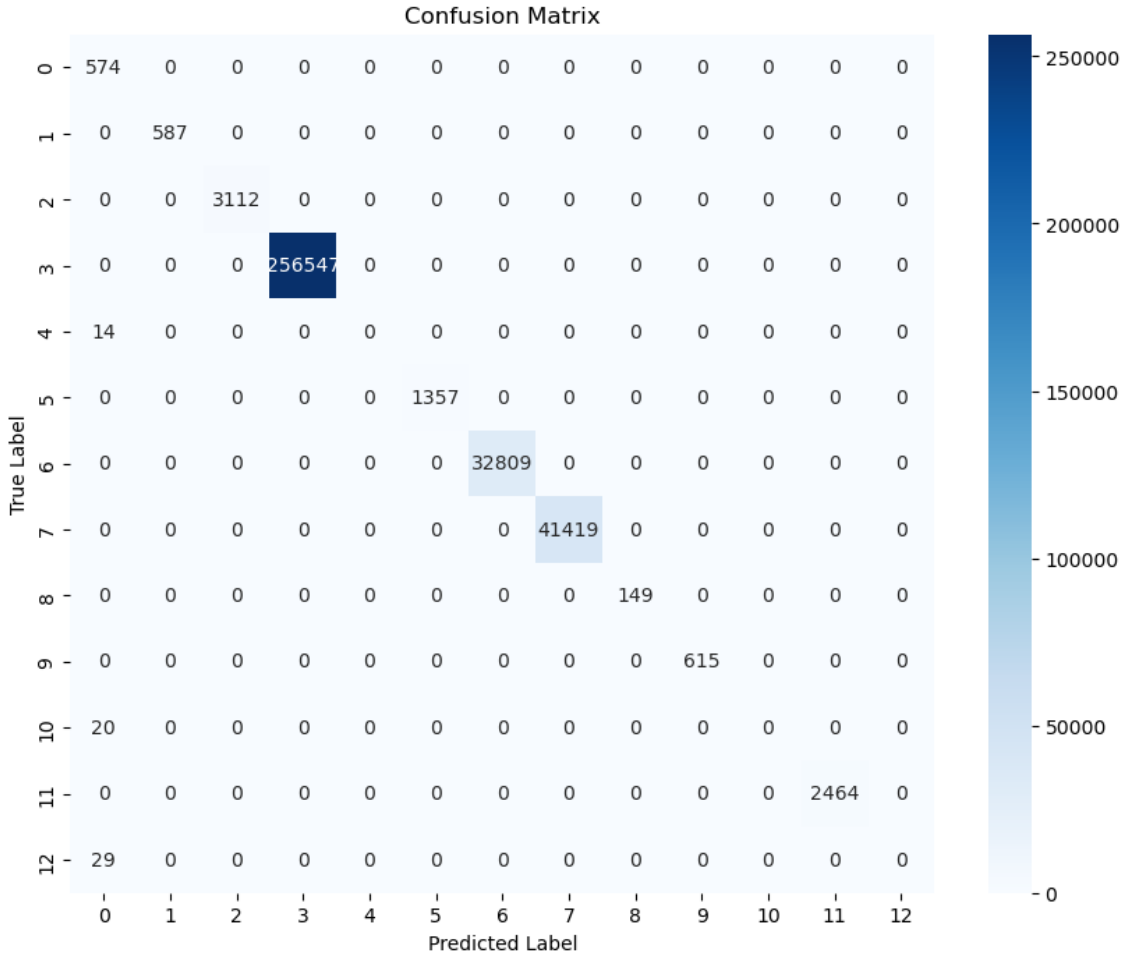

Figure 6.4: Class Confusion matrices for ensemble Method