Tribhuvan University

Institute of Science & Technology

A Deep Learning Approach for Intrusion Detection using Recurrent Neural Network

Dissertation

Submitted To:

Central Department of computer Science & information Technology

Tribhuvan University

Kirtipur, Kathmandu

Nepal

In partial Fulfillment of the requirements for the Degree of Master of Science in computer science and information technology

Submitted By:

Dipendra Rai

July, 2018

Supervisor

Mr. Bikash Balami

Kirtipur, CDCSIT

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

Student's Declaration

I hereby declare that I am the only author of this work and that no sources other than the listed here have been used in this work.

… … … … … … …….

Dipendra Rai

Date: July, 2018

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

Supervisor's Recommendation

I hereby recommend that the dissertation prepared under my supervision by **Mr. Dipendra Rai**entitled **"A Deep Learning Approach for Intrusion Detection using Recurrent Neural Network"** be accepted as in fulfilling partial requirement for completion of master Degree of science in computer science and information Technology.

… … … … … … … … … …

Mr. Bikash Balami (Lecturer)
Central Department of Computer Science and Information Technology
Kirtipur, Nepal
Date: July, 2018

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

LETTER OF APPROVAL

We certify that we have read this dissertation and in our opinion it is appreciable for the scope and quality as a dissertation in the partial fulfillment for the requirement of Master's Degree in

Computer Science and Information Technology.

Evaluation Committee

…….

Asst. Prof. Nawaraj Paudel
Head of Department
Central Department of Computer Science & Information Technology
Tribhuvan University
Kirtipur

…….

Mr. Bikash Balami
(Supervisor)
Central Department of Computer Science & Information Technology
Tribhuvan University
Kirtipur

…….
(Internal Examiner)

…….
(External Examiner)

Acknowledgement

I would like to express my sincere thanks to my supervisor **Mr. Bikash Balami**, CDCSIT for his support, motivation, suggestions and guidance. His advice was inevitable and with his help I was able to work on my own interested field and complete my thesis on time.

I am also thankful to **Mr. Nawaraj Poudel**, Head of Department, CDCSIT who has provided all the help and facilities, which I required, for the completion of my thesis.

Moreover, I would like to express my heartfelt gratitude to all my teachers at Central Department of Computer Science and Information Technology, Tribhuvan University who have imparted knowledge in various subjects.

Last but not the least; I would like to express my thanks to all my friends and my lovely parents for direct and indirect supports for the completion of this thesis.

Dipendra Rai

Abstract

Intrusion detection discover a critical part in guaranteeing data security and the key innovation is to precisely recognize different assaults in the system. In this dissertation, the intrusion detection model based on deep learning is investigated, and a deep learning approach for intrusion detection using recurrent neural networks (RNN-IDS) is used. The performance of the model is based on binary and multiclass classification, and the number of neurons and different learning rate impacts on the performance of the model has been studied. The performance of the model is compared with Naïve Bayes, Multilayer Perceptron and Support Vector Machine that has been analyzed by previous researchers on the benchmark data set. The test results demonstrate that RNN-IDS is remarkably appropriate for displaying high precision and its execution is better than that of machine learning techniques in both binary and multiclass classification. The RNN-IDS demonstrate enhances the precision of the intrusion identification and gives other examination strategy to intrusion detection discovery.

Keywords: *Recurrent neural networks, RNN-IDS, intrusion detection, deep learning, machine learning.*

Table of Contents

Acknowledgement i
Abstract ii
Table of Contents iii
List of Figures iv
List of Tables v
List of Abbreviations vi
CHAPTER 1 1
INTRODUCTION 1
1.1 Introduction 1
1.2 Problem Statement 4
1.3 Objectives 5
1.4 Limitation 5
1.5 Structure of Report 5
CHAPTER 2 6
LITERATURE REVIEW 6
2.1 Background and Literature Review 6
CHAPTER 3 9
RESEARCH METHODOLOGY 9
3.1 Data Description 9
3.2 Data Preprocessing 14
3.2.1 Numericalization 14
3.2.2 Normalization 15
3.3 Recurrent Neural Network 15
3.3.1 Algorithm 19
3.4 Evaluation Matrices 21
CHAPTER 4 24
RESULT, ANALYSIS AND COMPARISION 24
4.1 Result Analysis and Comparison 24
4.1.1 Binary Classification 24
4.1.2 Multiclass Classification 27
4.1.3 Comparison 30
CHAPTER 5 33
CONCLUSION 33
Reference 34
Appendix 37

List of Figures

Figure 1.1: Basic Intrusion Detection Architecture 2

Figure 1.2: A recurrent neural network and the unfolding in time of the computation involved in its forward computation. 4

Figure 3.1: Block diagram of proposed RNN-IDS 16

Figure 3.2: Training time (in epochs) and generalization accuracy for each learning rate, on a digit speech recognition task. 23

Figure 4.1: The accuracy and training time (second) of RNN-IDS with different layer and learning rate for binary classification 20

Figure 4.2: The accuracy and training time (second) of RNN-IDS with different layer and learning rate for multiclass classification 23

Figure 4.3: Performance of J48, Naïve Bayes, RF, MLP and SVM based on NSL-KDD dataset for binary classification 25

Figure 4.4: Performance of RNN-IDS and the other models in the binary classification. 26

Figure 4.5: Performance of J48, Naïve Bayes, RF, MLP and SVM based on NSL-KDD dataset in multiclass classification 26

Figure 4.6: Performance of RNN-IDS and the other models in the multiclass classification 27

List of Tables

Table 3.1: Different classifications in the NSL-KDD dataset 9

Table 3.2: Basic Features of Each Network Connection Vector 10

Table 3.3: Content Related Features of Each Network Connection Vector 11

Table 3.4: Time Related Traffic Features of Each Network Connection Vector 11

Table 3.5: Host Based Traffic Features of Each Network Connection Vector 11

Table 3.6: Attribute Value Type 14

Table 3.7: Mapping of Attack Class with Attack Type 14

Table 4.1: The accuracy and training time (second) of RNN-IDS with different layer and learning rate for binary classification 19

Table 4.2: Confusion matrix of 2-category classification on KDDTest+ with LR 0.01 20

Table 4.3: The testing of RNN-IDS based on KDDTest+ with LR 0.01 20

Table 4.4: Confusion matrix of 2-category classification on KDDTest-21 with LR 0.01 21

Table 4.5: The testing of RNN-IDS based on KDDTest-21 with LR 0.01 21

Table 4.6: Confusion matrix of 2-category classification on KDDTest+ with LR 0.3 21

Table 4.7: The testing of RNN-IDS based on KDDTest+ with LR 0.3 21

Table 4.8: Confusion matrix of 2-category classification on KDDTest-21 with LR 0.3 22

Table 4.9: The testing of RNN-IDS based on KDDTest-21 with LR 0.3 22

Table 4.10: The accuracy and training time (second) of RNN-IDS with different layer and learning rate for multiclass classification 22

Table 4.11: Confusion matrix of multi-class classification on KDDTest+ with LR 0.01 23

Table 4.12: The testing of RNN-IDS based on KDDTest+ with LR 0.01 for multi-class 23

Table 4.13: Confusion matrix of multi-class classification on KDDTest-21 with LR 0.01 24

Table 4.14: The testing of RNN-IDS based on KDDTest-21 with LR 0.01 for multi-class 24

Table 4.15: Confusion matrix of multi-class classification on KDDTest+ with LR 0.3 24

Table 4.16: The testing of RNN-IDS based on KDDTest+ with LR 0.3 for multi-class 24

Table 4.17: Confusion matrix of multi-class classification on KDDTest-21 with LR 0.3 25

Table 4.18: The testing of RNN-IDS based on KDDTest-21 with LR 0.3 for multi-class 25

Table 4.19: Detection Rate in binary classification for different algorithm 26

Table 4.20: Detection Rate in multiclass classification for different algorithm 26

List of Abbreviations

CNN	Convolutional Neural Network
DBN	Deep Belief Network
DNN	Deep Neural Network
DOS	Denial of Service
GPU	Graphics processing unit
HIDS	Host Intrusion Detection System
IDS	Intrusion Detection System
IP	Internet Protocol
KDD	Knowledge Discovery and Data Mining
LSTM	Long Short Term Memory
MLP	Multilayer Perceptron
NIDS	Network Intrusion Detection System
NLP	Natural Language Processing
NSL-KDD	Network Socket Layer - Knowledge Discovery and Data Mining
Probe	Probing
R2L	Root to Local
RNN	Recurrent Neural network
RNN-IDS	Recurrent Neural network - Intrusion Detection System
SVM	Support vector Machine
U2R	User to Root

CHAPTER 1

INTRODUCTION

1.1 Introduction

With the undeniably profound mix of the Internet and society, the Internet is changing the manner by which individuals live, study and work, yet the different security dangers that we confront are ending up increasingly genuine. Distinguishing different system assaults, particularly unexpected assaults us an unavoidable key specialized issue.

A critical issue in the field of computer systems has been the failure to sufficiently shield internet-associated computer systems from security assaults. An Intrusion Detection System (IDS), a noteworthy research accomplishment in the data security field, can distinguish an attack, which could be a succeeding intrusion that has just happened.

An IDS is a system that monitors network traffic for suspicious activity and generate alerts when such activity is revealed. Intrusion detection takes a simple premise: every network resource and user develops and displays a pattern of normal usage that is specific and possibly unique. In spite of anomalies in network usage that they appear, they should be reasonable. Anything that cannot be readily explained should be considered a probable attack and investigated. Intrusion detection systems automate much of this process.

Intrusion recognition is the way toward observing the frequently happening in the system and breaking down them for indications of possible occurrences, intrusion, or inevitable threats to the security approaches. A typical business network has several access points to other networks, both public and private. The challenge is maintaining the security of these networks while keeping them open to their customers. Currently, attacks are so sophisticated that they can thwart the best security systems, especially those that still operate under the assumption that networks can be secured by encryption or firewalls. Unfortunately, those technologies alone are not sufficient to counter today’s attacks.

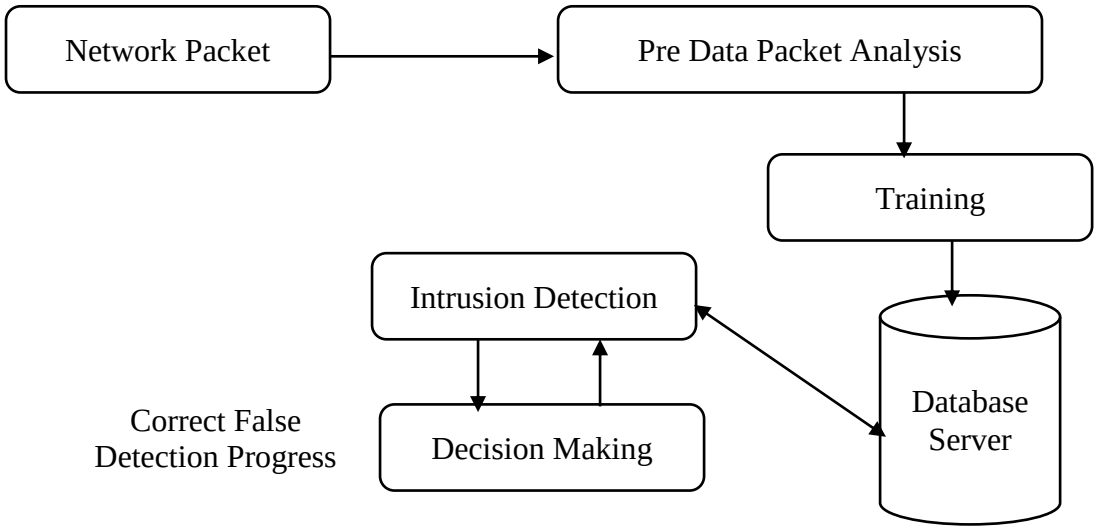

Figure 1.1: Basic Intrusion Detection Architecture

Intrusion detection systems used to detect suspicious activities using different methods, including the following:

A network intrusion detection system (NIDS) is deployed at a strategic point or points within the network, where it can monitor inbound and outbound traffic to and from all the devices on the network.

- Host intrusion detection systems (HIDS) run on all computers or devices in the network with direct access to both the internet and the enterprise internal network. HIDS is able to identify malicious traffic that originates from the host itself, as when the host has been infected with malware and is attempting to spread to other systems.

- Signature-based intrusion detection systems monitor all the packets traversing the network and compares them against a database of signatures or attributes of known malicious threats, much like antivirus software.

- Anomaly-based intrusion detection systems monitor network traffic and compare it against an established baseline, to determine what is considered normal for the network with respect to bandwidth, protocols, ports and other devices.

Machine learning techniques have been for the most part used in recognizing various types of faults, and can help maintaining those faults. Geoffrey Hinton a pioneer in the field of artificial neural networks and co-published the first paper on the back propagation algorithm for training multilayer perceptron networks have introduced the phrase "deep" to describe the

development of large artificial neural networks. According to Yoshua Bengio "Deep learning algorithms seek to exploit the unknown structure in the input distribution in order to discover good representations, often at multiple levels, with higher-level learned features defined in terms of lower-level features"

Deep learning based strategies have been effectively used in sound, picture and image processing applications. These strategies expect to take in a decent element representation from extensive measure of unlabeled information and in this manner apply these learntfeatures on a constrained measure of named information in a managed grouping. The marked and unlabeled information originate from various transmissions, however they should be remarkable to one another [3].

Because of developing computational assets, recurrent neural systems (RNNs) (which have been around for a considerable length of time however their maximum capacity has recently turned out to be broadly perceived. Convolution Neural Networks (CNNs) have been established as a powerful class of models for image recognition problem [4]. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being dependent on the previous computations.

In recent years, RNNs have played an important role in the fields of computer vision, natural language processing (NLP), semantic understanding, speech recognition, language modeling, translation, picture description, and human action recognition [5], among others.

Recurrent neural networks include input units, output units and hidden units, and the hidden unit completes the most important work. The RNN model essentially has a one-way flow of information from the input unit to the hidden units, and the synthesis of the one-way information flow from the previous temporal concealment unit to the current timing hiding unit. A RNN approach can be used for supervised classification learning.

The below diagram shows a RNN being unfolded into a full network. By unrolling we can write out the network for the complete sequence.

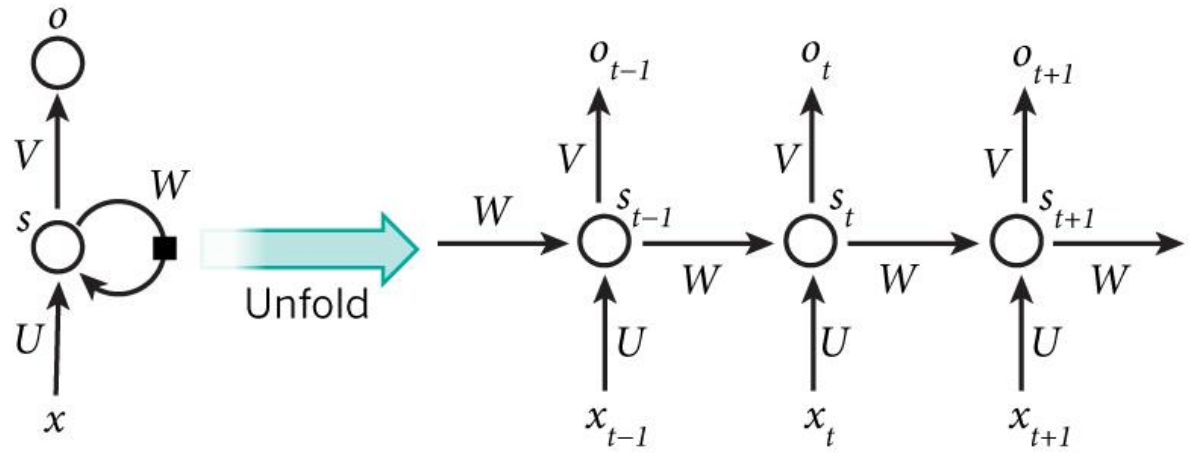

Figure 1.2: A recurrent neural network and the unfolding in time of the computation involved in its forward computation. *Source: Nature*

1.2 Problem Statement

The drastic growth in the volume of network data, which is set to continue. This growth can be predominantly attributed to increasing levels of connectivity, the popularity of the Internet of Things and the extensive adoption of cloud based services. Dealing with these volumes requires techniques that can analyses data in an increasingly rapid, efficient and effective manner. The in-depth monitoring and granularity required to improve effectiveness and accuracy. NIDS analysis needs to be more detailed and contextually-aware, which means shifting away from abstract and high-level observations. For example, behavioral changes need to be easily attributable to specific elements of a network, e.g. individual users, operating system versions or protocols. Also the number of different protocols and the diversity of data traversing through modern networks. This is possibly the most significant challenge and introduces high-levels of difficulty and complexity when attempting to differentiate between normal and abnormal behavior. It increases the difficulty in establishing an accurate norm and widens the scope for potential exploitation or zero-day attacks.

1.3 Objectives

The main objective of this dissertation is:

- To design and implementation of the intrusion detection system based on recurrent neural networks.
- To study the performance of the model in binary classification and multiclass classification and compare with the performance of Naïve Bayes, Support Vector Machine (SVM) and Multilayer Perceptron.

1.4 Limitation

Limitations of this research were:

- The study and analysis in this dissertation is based on NSL-KDD dataset and comparison is with the performance of previous researcher's analysis on the dataset.
- The research is focused on Accuracy rate, True positive rate and False positive rate

1.5 Structure of Report

This report is organized is organized in six chapters including the following chapters.

- Chapter 1 of this dissertation work is introduction part, which is organized into subsequent four chapters.
    - First chapter is focused on introduction and overview of intrusion detection and Deep Learning Approach.
    - Second chapter is about problem analysis of existing or previous works which demands further study to get better solutions.
    - Third chapter describe the main objective of this dissertation works.
    - Fourth chapter is about limitation of this dissertation works.
- Chapter 2 contains explanation of all previous studies related to this topic in detail under literature review.
- Chapter 3 contains all the details of data which is applied for analysis purpose and comparative performance measure and also the algorithms used for the research
- Chapter 4 includes the result analysis and comparison of implemented algorithm i.e. RNN and other algorithm based on NSL-KDD dataset. The result of the study is shown in tabular form as well as in graph.
- Chapter 5 provides final conclusion and future works of the study.

CHAPTER 2

LITERATURE REVIEW

2.1 Background and Literature Review

H. Soliman et.al [6] studies a number of approaches based on traditional machine learning, which includes Fuzzy C-means Clustering, Back Propagation Neural Network, Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Multi-agent and Refined Clustering, Random Forest (RF). For each IDS, the architecture and the related functionality are briefly introduced, discussed, and compared, focusing on both the operational strengths and weakness. Also, a comparison of the studied IDSs is carried out using a set of critical evaluation metrics that are divided into two groups; the first one related to performance and the second related to security.

Dong and Wang [15] discusses different methods which were used to classify network traffic. The authors concluded that deep learning has gained prominence due to the potential it portends for machine learning. For this reason, deep learning techniques have been applied in many fields, such as recognizing some kinds of patterns or classification. Intrusion detection analyses got data from monitoring security events to get situation assessment of network. Lots of traditional machine learning method has been put forward to intrusion detection, but it is necessary to improvement the detection performance and accuracy. The author use different methods on open data set and did experiment with these methods to find out the best to intrusion detection.

Javaid et al. [7], a deep learning based approach for developing such an efficient and flexible NIDS. We use Self-taught Learning (STL), a deep learning based technique, on NSL-KDD - a benchmark dataset for network intrusion. Also they observed that the proposed NIDS performed very well compared to previously implemented NIDSs (NB-Tree, Random Tree, or J48) for the normal/anomaly detection when evaluated on the test data.

Tang et al. [8], apply a deep learning approach for flow-based anomaly detection in an SDN (Software Defined Network) environment. They build a Deep Neural Network (DNN) model for an intrusion detection system and train the model with the NSLKDD Dataset. For this, they use six basic features (duration, protocol_type, src_bytes, dst_bytes, cout, srv_count) taken from the forty one features of NSL-KDD Dataset. Through experiments, they

concluded that the deep learning approach shows strong potential to be used for flow-based anomaly detection in SDN environments

Shekahn et al. [9], proposed three-layer Recurrent Neural Network (RNN) architecture with categorized features as inputs and attack types as outputs of RNN as misuse based IDS. The input features are categorized to basic features, content features, time-based traffic features, and host-based traffic features. The attack types are classified to Denial-of-Service (DoS), Probe, Remote-to-Local (R2L), and User-to-Root (U2R). Their experimental result shows that the proposed model is able to improve classification rate, particularly in R2L attacks and it also show better detection rate when compared to similar other algorithms.

Zhao et al. [10review and summarize the emerging research work of deep learning on machine health monitoring. with brief introduction of deep learning techniques and its applications in machine health monitoring systems are reviewed mainly with the following aspects: Autoencoder (AE) and its variants, Restricted Boltzmann Machines and its variants including Deep Belief Network (DBN) and Deep Boltzmann Machines (DBM), Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN).

Alrawashdeh and Purdy [11], proposed a deep learning approach for anomaly detection using a Restricted Boltzmann Machine (RBM) and a deep belief network are implemented. They uses a one-hidden layer RBM to perform unsupervised feature reduction. The resultant weights from this RBM are passed to another RBM producing a deep belief network. The pre-trained weights are passed into a fine tuning layer consisting of a Logistic Regression (LR) classifier with multi-class soft-max. They use the DARPA KDDCUP'99 dataset to evaluate the performance. They achieve a detection rate of 97.9% on the total 10% KDDCUP'99 test dataset. By improving the training process of the simulation, they are also able to produce a low false negative rate of 2.47%. Although the deficiencies in the KDDCUP'99 dataset are well identified, it still presents machine learning approaches for predicting attacks with a reasonable challenge.

Kim et al. [12], an artificial intelligence (AI) intrusion detection system using a deep neural network (DNN) investigated and tested with the KDD Cup 99 dataset in response to ever-evolving network attacks. The authors claimed an average accuracy rate of 99%, and summarized that both RNN and Long Short-Term Memory (LSTM) models are needed for improving future defenses.

Lee et al. [16] proposed a deep neural network based approach for alleviating the problems of self-training by combining schemes: pre-training, dropout and error forgetting. By applying combinations of these schemes to various dataset, a trained classifier using the authors approach shows improved performance than trained classifier using common self-training. However, the author also shows that the combination of the error forgetting and example re-evaluation shows the performance degradation in the experiments.

You et al. [13] proposed an automatic security auditing tool for short messages (SMS). Their method is based upon the RNN model. The feature of short messages was extracted by word2vec which captures word order information, and each sentence is mapped to a feature vector. In particular, words with similar meaning are mapped to a similar position in the vector space, and then classified by RNNs. First preprocess short messages, extract typical features from the existing security and non-security messages via word2vec, and classify short messages through RNN which accept a fixed –sized vector as input and produce a fixed sized vector as output. Their experimental results show that the RNN model achieves an average of 92.7 percent accuracy which is higher than SVM.

The finding from the literature review has shown that despite the high detection accuracies being achieved, there is still possibility for the improvement. The area is still in infantile stage, with most researchers still experimenting on combining various algorithms (e.g.training, optimization, activation and classification) and layering approaches to produce the most accurate and efficient solutions for a specific dataset. Hence the model and work presented in this paper will be able to make a valid contribution to the current pool of knowledge.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Data Description

The NSL-KDD dataset [14] produced in 2009 is broadly utilized in intrusion detection tests. The dataset covers the KDDTrain+ dataset as the training set and KDDTest+ and KDDTest-21 datasets as the testing set, which has diverse ordinary records and four kinds of attack records, as appeared in table below. The KDDTest-21 dataset is a subset of KDDTest+ and is harder for grouping.

NSL-KDD data set which is used for training and testing of intrusion detection [1] is shown below.

Table 3.1: Different classifications in the NSL-KDD dataset

	Total	Normal	DOS	Probe	R2L	U2R
KDDTrain+	125973	67343	45827	11456	995	49
KDDTest+	25192	13449	9234	2289	209	11
KDDTest-21	22542	12709	7749	1867	175	42

The advantages of choosing NSL-KDD dataset over KDD CUP 99 are:

1. Redundant records are removed to enable the classifiers to produce an un-biased result.
2. Sufficient number of records is available in the train and test data sets, which is reasonably rational and enables to execute experiments on the complete set.
3. The number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD data set. As a result, the classification rates of distinct machine learning methods vary in a wider range, which makes it more efficient to have an accurate evaluation of different learning techniques.
4. The number of records in the train and test sets are reasonable, which makes it affordable to run the experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research works will be consistent and comparable.

Data Files:

- **KDDTrain+.ARFF**: The full NSL-KDD train set with binary labels in ARFF format
- **KDDTest+.ARFF**: The full NSL-KDD test set with binary labels in ARFF format
- **KDDTest21.ARFF**: A subset of the KDDTest+.arff file which does not include records with difficulty level of 21 out of 21

The NSL-KDD contains all the essential records of the complete KDD data set. In each record there are 41 attributes unfolding different features of the flow and a label assigned to each either as an attack type or as normal. The details of the attributes namely the attribute name, their description and sample data are listed in the Tables 3.3, 3.4, 3.5, 3.6. The Table 3.7 contains type information of all the 41 attributes available in the NSL-KDD data set. The 42nd attribute contains data about the various 5 classesof network connection vectors and they are categorized asone normal class and four attack class. The 4 attack classesare further grouped as DoS, Probe, R2L and U2R.

Table 3.2: Basic Features of Each Network Connection Vector

Attribute No.	Attribute Name	Description	Sample Data
1	Duration	Length of time duration of the connection	0
2	Protocol_type	Protocol used in the connection	TCP
3	Service	Destination network service used	ftp_data
4	Flag	Status of the connection – Normal or Error	SF
5	Src_bytes	Number of data bytes transferred from source to destination in single connection	491
6	Dst_bytes	Number of data bytes transferred from destination to source in single connection	0
7	Land	f source and destination IP addresses and port numbers are equal then, this variable takes value 1 else 0	0
8	Wrong_fragment	Total number of wrong fragments in this connection	0
9	Urgent	Number of urgent packets in this connection. Urgent packets are packets with the urgent bit activated	0

Table 3.3: Content Related Features of Each Network Connection Vector

Attribute No.	Attribute Name	Description	Sample Data
10	Hot	Number of "hot‟ indicators in the content such as: entering a system directory, creating programs and executing programs	0
11	Num_failed_logins	Count of failed login attempts	0
12	Logged_in	Login Status : 1 if successfully logged in; 0 otherwise	0
13	Num_compromised	Number of ``compromised' ' conditions	0
14	Root_shell	1 if root shell is obtained; 0 otherwise	0
15	Su_attempted	1 if ``su root" command attempted or used; 0 otherwise	0
16	Num_root	Number of ``root" accesses or number of operations performed as a root in the connection	0
17	Num_file_creations	Number of file creation operations in the connection	0
18	Num_shells	Number of shell prompts	0
19	Num_access_files	Number of operations on access control files	0
20	Num_outbound_cmds	Number of outbound commands in an ftp session	0
21	Is_hot_login	1 if the login belongs to the ``hot" list i.e., root or admin; else 0	0
22	Is_guest_login	1 if the login is a ``guest" login; 0 otherwise	0

Table 3.4: Time Related Traffic Features of Each Network Connection Vector

Attribute No.	Attribute Name	Description	Sample Data
23	Count	Number of connections to the same destination host as the current connection in the past two seconds	2
24	Srv_count	Number of connections to the same service (port number) as the current connection in the past two seconds	2
25	Serror_rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in count (23)	0
26	Srv_serror_rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in srv_count (24)	0

27	Rerror_rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in count (23)	0
28	Srv_rerror_rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in srv_count (24)	0
29	Same_srv_rate	The percentage of connections that were to the same service, among the connections aggregated in count (23)	1
30	Diff_srv_rate	The percentage of connections that were to different services, among the connections aggregated in count (23)	0
31	Srv_diff_host_rate	The percentage of connections that were to different destination machines among the connections aggregated in srv_count (24)	0

Table 3.5: Host Based Traffic Features of Each Network Connection Vector

Attribute No.	Attribute Name	Description	Sample Data
32	Dst_host_count	Number of connections having the same destinationhost IP address	150
33	Dst_host_srv_count	Number of connections having the same port number	25
34	Dst_host_same_srv_rate	The percentage of connections that were to the same service, among the connections aggregated in dst_host_count(32)	0.17
35	Dst_host_diff_srv_rate	The percentage of connections that were to different services, among the connections aggregated in dst_host_count(32)	0.03
36	Dst_host_same_src_port_rate	The percentage of connections that were to thesame source port, among the connections aggregated in dst_host_srv_count (33)	0.17
37	Dst_host_srv_diff_host_rate	The percentage of connections that were to different destination machines, among the connections aggregated in dst_host_srv_count(33)	0
38	Dst_host_serror_rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_count (32)	0
39	Dst_host_srv_serror_rate	The percent of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_srv_count (33	0
40	Dst_host_rerror_rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in dst_host_count (32)	0.05
41	Dst_host_srv_rerror_rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in dst_host_srv_count (33)	0

The attack classes present in the NSL-KDD data set are grouped into four categories [17]:

1. DOS: Denial of service is an attack category, which reduces the victim's resources thus making it unable to handle genuine requests – e.g. syn flooding. the features for DOS attack are: “source bytes” and “percentage of packets with errors”
2. Probing: Observation and other probing attack's objective is to getting information about the remote target e.g. port scanning. the features for Probing attack are: “duration of connection” and “source bytes”
3. U2R: illegal access to local super user (root) privileges is an attack type, by which an attacker uses a usual account to login into a target system and tries to gain root/administrator privileges by misusing some weakness in the victim e.g. buffer overflow attacks. the features for U2R attack are: “number of file creations” and “number of shell prompts invoked,”
4. R2L: illegal access from a remote machine, the attacker intrudes into a remote machine and gains local access of the target machine. E.g. password guessing the features for R2L attack are: “duration of connection” and “service requested” and host level features - “number of failed login attempts”

Table 3.6: Attribute Value Type

Types	Features
Nominal	Protocol_type(2), Service(3), Flag(4)
Binary	Land(7), logged_in(12), root_shell(14), su_attempted(15), is_host_login(21), is_guest_login(22)
Numeric	Duration(1), src_bytes(5), dst_bytes(6), wrong_fragment(8),urgent(9), hot(10),num_failed_logins(11),num_compromised(13),num_root(16),num_file_creations(17),num_shells(18),num_access_files(19), num_outbound_cmds(20), count(23), srv_count(24), serror_rate(25),srv_serror_rate(26), rerror_rate(27),srv_rerror_rate(28), same_srv_rate(29), diff_srv_rate(30), srv_diff_host_rate(31), dst_host_count(32), dst_host_srv_count(33), dst_host_same_srv_rate(34), dst_host_diff_srv_rate(35), dst_host_same_src_port_rate(36), dst_host_srv_diff_host_rate(37), dst_host_serror_rate(38), dst_host_srv_serror_rate(39), dst_host_rerror_rate(40), dst_host_srv_rerror_rate(41)

The specific types of attacks are classified into four major categories. The table 3.8 shows this detail.

Table 3.7: Mapping of Attack Class with Attack Type

Attack Class	Attack Type
DoS	Back, Land, Neptune, Pod, Smurf,Teardrop,Apache2, Udpstorm, Processtable, Worm
Probe	Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint
R2L	Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendmail, Named
U2R	Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps

3.2 Data Preprocessing

3.2.1 Numericalization

In NSL-KDD dataset there are 38 numeric features and 3 nonnumeric features. Since the RNN-IDS should have input to be a numeric matrix, so we have to convert the nonnumeric features entitled protocol_type, service and flag to numeric value. The feature 'protocol' has 3 types of attributes, feature 'service' have 70 types of attributes and feature 'flag' have 11 types of attributes.After numericalization of the NSL-KDD data set the 41- dimensional features map into 122-dimensional features after transformation.

3.2.2 Normalization

Some of the features (duration, src_byte, flag, and service) in NSL-KDD dataset have very large scope i.e. the minimum and maximum value for the feature value greatly differs. So the logarithmic scaling method is used to obtain the value in the range. Since, the value of every feature is mapped to the [0, 1] range linearly according to (1), where Max denotes the maximum value and Min denotes minimum value for each feature.

$$x_i = \frac{x_i - Min}{Max - Min} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 1$$

For duration value 2074 i.e. present on NSL-KDD dataset, first logarithmic scaling method is used by which we get 3.32, Max value and Min value for duration is 0, 4.77 respectively now to map the feature to [0, 1] range then:

$$x_{duration} = \frac{x_{duration} - Min}{Max - Min}$$

$$\text{Normalized value for duration 2074 is } = \frac{3.22 - 0}{4.77 - 0} = 0.68$$

Also to normalize the non-numeric features that has been given numeric value has been calculated as, for example, the non-numeric feature protocol type is divided into 3 types' tcp, udp and icmp and through numericalization process given 1, 2, and 3 numeric value respectively. Then for normalized value for udp, the value for udp is 1, Min value is 1 and Max value is 3

$$x_{udp} = \frac{x_{udp} - Min}{Max - Min}$$

$$\text{Normalized value for udp is} = \frac{2 - 1}{3 - 1} = 0.5$$

3.3 Naïve Bayes

The naïve Bayes model is a straightforward Bayesian probability model which operates on a strong independence assumption also the probability of one attribute does not affect the probability of the other. Given n series of attributes, the naïve Bayes classifier makes 2n! Independent assumptions. The error of naïve Bayes in output is the result of three factors: training data noise, bias, and variance. Training data noise can only be minimized by selecting worthy training data. The training data must be separated into various sets by the

machine learning algorithm. Bias is the error due to groupings in the training data being very big. Variance is the error due to those groupings being too small. The algorithm for naïve Bayes for intrusion detection using NSL-KDD dataset is shown below.

Input:

F=Full set of 41 features of NSL-KDD dataset

ac= classifiers accuracy

err= RMSE

avg_tpr= average TPR

// ac, err and avg-tpr resulted from invocation of NBC on full dataset, these values used as threshold values for feature selection

Algorithm: [21]

Begin

Initialize: S={F}

For each feature {f} form

(1) T=S-{f}

(2) Invoke Naïve Bayes classifier on dataset with T features

(3) If CA>= ac And RMSE<=err And A_TPR>= avg_tpr then

S=S-{f} F=S

// Set F with reduced features

End

3.4 Support Vector Machine (SVM)

The SVM, originally a type of pattern classifier based on a statistical learning technique for classification and regression with a variety of kernel functions, has been successfully applied to a number of pattern recognition applications. Due to good generalization nature and the ability to overcome the curse of dimensionality, SVM is the poplar technique for anomaly intrusion detection. The SVM select appropriate setup parameters because it does not depend on traditional empirical risk such as neural networks. One of the main advantage of using SVM for IDS is its speed, as the capability of detecting intrusions in real-time is very important. SVMs can learn a larger set of patterns and be able to scale better, because the classification complexity does not depend on the dimensionality of the feature space [22]. The algorithm for intrusion detection for Support Vector Machine (SVM) is as follow:

Input:

F – Full feature set

IGR: Information Gain Ratio Measure

C: K-means classifier

T: Gained Accuracy Threshold For each feature f compute IGR(f)

Output:

S – Best feature subset

Algorithm [22]

Initialize:

S={}, ac=0

Repeat

- Assign acp= ac
- Evaluate f=getNext(F)
- Calculate S=SU{f}
- Calculate F=F-{f}
- Evaluate ac = accuracy(C,S)
- Continue the above steps until (ac-acp) < T Or ac <acp

3.5 Multilayer Perceptron

A multi-layer perceptron is a feed-forward neural network, consisting of a number of units (neurons) which are connected by weighted links. The units are organized in several layers, namely an input layer, one or more hidden layers, and an output layer. The input layer receives an external activation vector, and passes it via weighted connections to the units in the first hidden layer. These compute their activations and pass them to neurons in succeeding layers.

The algorithm for Multilayer Perceptron for intrusion detection is as follows [20]

Input: Training data set

Output: Best parameters

InitPopulation (Pop): Initializes the population;

EvalPopulation (Pop): Evaluate the fitness of each individual in the population;

1: while not termination do

2: Select best-ranking individuals from Pop and delete others;

//crossover

3: for i=1 to (Pop.Length)/2 do

4: Randomly select two solutions Xa and Xb from population;

5: Generate Xc and Xd by crossover to Xa and Xb;

6: Sace Xc and Xd in Pop;

7: end for

//Mutation

8: for i=1 to Pop.Length do

9: Select Solution Xi from Pop;

10: Mutate a set of bits randomly and generate Xi';

11: Update Xi by Xi' in Pop;

12: endfor

//Evaluation

13: for i=1 tp Pop.Length do

14: Train MLP with using soltion Xi from Pop;

15: Evaluate MLP model and calculate accuracy;

16: endfor

3.6 Recurrent Neural Network

Recurrent neural networks have introduced a directional loop that can memorizes the previous information and apply it to the current output, which is the essential difference from traditional Feed-Forward Neural Networks (FNNs). The preceding output is also related to the current output of sequence, and the nodes between the hidden layers are no longer connectionless: instead they have connections. Not only the output of the input layer but also the output of the last hidden layer acts on the input of the hidden layer. The steps involved in RNN-IDS is shown below.

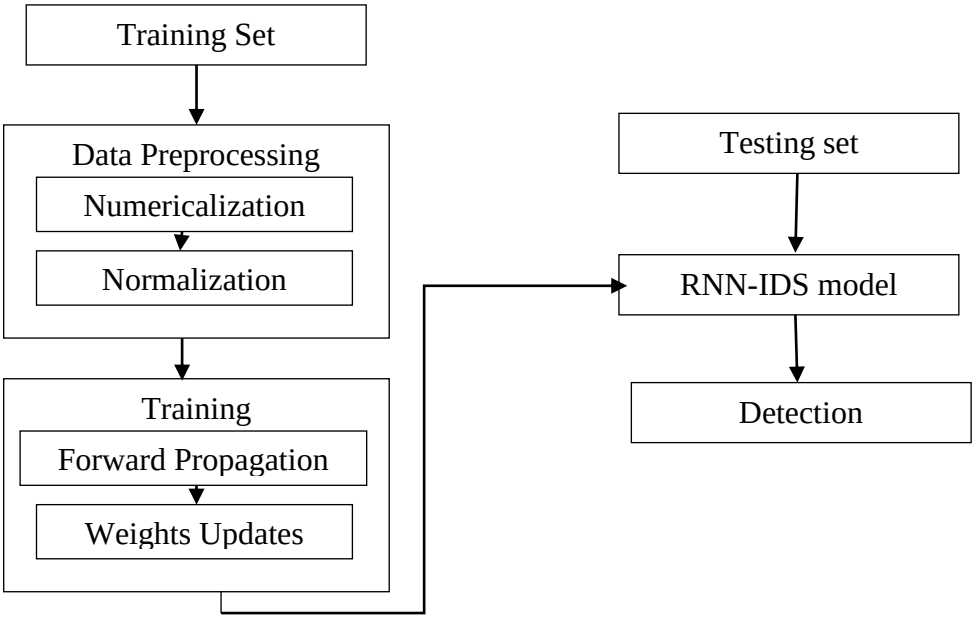

Figure 3.1: Block diagram of RNN-IDS [5]

3.6.1 Recurrent Neural Network

The training of RNN_IDS model consists of two parts – Forward Propagation and Back Propagation. Forward Propagation is responsible for calculating the output values and Back Propagation is responsible for passing the residuals that were accumulated to update the weights.

The standard RNN is formalized as follows:

Given training samples $x_i = (i = 1,2, \dots m)$ a sequence of hidden states $h_i = (i = 1,2, \dots m)$ and a sequence of prediction$\hat{y}_i = (i = 1,2, \dots m)$. $W_{hx}$is the input-to-hidden weight matrix, W_{hh} is hidden-to-hidden weight matrix, W_{yh} is the hidden-to-output weight matrix, and the vectors b_h and b_y are the biases. The activation function Θ is a sigmoid and the classification function g engages the SoftMax function.

The objective function associated with RNNs for a single training pair (x_i, y_i) is defined as $f(\theta) = L_{(y_i : \hat{y}_i)}$where L is the distance function which measures the derivation of the predictions $\hat{y}_i$ from the actual label y_i . Let η be the learning rate and k be the number if current iterations.

Algorithm 1: Forward Propagation Algorithm [8]

Input: $x_i = (i = 1,2, \dots m)$

Output: $\hat{y}_i$

1: for i from 1 to m do

2: $t_i = W_{hxxi} + W_{hhhi-1+bh}$

3: $h_i = sigmoid(t_i)$

4: $s_i = W_{yhhi+by}$

5: $\hat{y}_i = SoftMax\ (s_i)$

6: end for

Algorithm 2: Weight Update Algorithm [8]

Input: $(y_i\ :\ \hat{y}_i)(i = 1,2, \dots m)$

Initialization: $\theta = W_{hx}, W_{hh}, W_{yh}, b_h, b_y$

Output: $\theta = W_{hx}, W_{hh}, W_{yh}, b_h, b_y$

1: for i from k downto 1 do

2: calculate the cross entropy between the output value and the label value: $L_{(y_i : \hat{y}_i)} \longleftarrow \sum_i \sum_j y_{ij} \log(\hat{y}_{ij}) + (1 - y_{ij}) \log(1 - \hat{y}_{ij})$

3: compute the partial derivatives with respect to θ_i: $\delta_i \longleftarrow dL/d\theta_i$

4: weight update: $\theta_i \leftarrow \theta_i \eta + \delta_i$

5: end for

Why RNN?

Recurrent Neural networks are the state of the art algorithm for the sequential data. It is the algorithm that remembers the inputs they received, which enables them to be very precise in predicting what comes next, due to its internal memory, which makes its perfectly suitable for machine learning problems that involve sequential data.

Recurrent Neural network has two inputs, the present and the recent past. This is important because the sequence of data contains crucial information about what is coming next, so RNN can do more precisely than other algorithm.

Since in this dissertation NSL-KDD dataset have been used for training and testing purpose which is a time series data. These testing and training data are in sequential format. After training of the algorithm, the algorithm have to precisely detect the intrusion, so RNN with recent past information makes intrusion detection better. The data set used and for better detection rate RNN is better option.

3.7 Evaluation Matrices

The Accuracy, AC of intrusion detection is used to measure the performance of the RNN-IDS model. The True Positive (TP) is equal to those correctly excluded, and it denotes the number of anomaly records that are identified as anomaly. The False Positive (FP) is the equal of wrongly excluded, and it denotes the number of normal records that are identified as anomaly. The True Negative (TN) is equal to those correctly admitted, and it denotes the number of normal records that are identified as normal. The False Negative (FN) is equal to those incorrectly admitted, and it denotes the number of anomaly records that are identified as normal.

Accuracy: the percentage of the number of records classified correctly versus total the records shown in (2).

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative} \quad (2)$$

True Positive Rate (TPR): as the equivalent of the Detection Rate (DR), it shows the percentage of the number of records identified correctly over the total number of anomaly records, as shown in (3).

$$True\ Positive\ Rate = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (3)$$

False Positive Rate (FPR): the percentage of the number of records rejected incorrectly is divided by the total number of normal records, as shown in (4).

$$False\ Positive\ Rate = \frac{False\ Positive}{False\ Positive + True\ Negative} \quad (4)$$

Hence, the motivation for the IDS is to obtain a higher accuracy and detection rate with a lower false positive rate.

3.8 Learning Rate

Learning rate that is too large often moves too far in the "correct" direction, resulting in overshooting a valley or minimum in the error surface, thus hurting accuracy. Because of this effect, a learning rate that is too large takes longer to train, because it is continually overshooting its objective and "unlearning" what it has learned, thus requiring expensive backtracking or causing unproductive oscillations. This instability often causes poor generalization accuracy as well, since the weights can never settle down enough to move all the way into a minimum before bouncing back out again.

D. Randall et.al [19] perform a test to illustrate the effect of learning rates on training speed and generalization accuracy on digit speech recognition. A multilayer perceptron with 130 inputs (plus one bias), 100 hidden nodes, and 178 outputs were trained on 21,085 training instances. The output class of each instance corresponded to one of 178 context-dependent phoneme categories from a digit vocabulary. Each context-dependent phoneme belonged to one of 23 base phoneme classes. For each instance, one of the 178 outputs had a target of 1, and all other outputs had a target of 0. The targets of each instance were derived from hand-labeled phoneme representations of a set of training utterances.

The neural network was trained using 15 different learning rates from 100 down to 0.00001. Each neural network began with random initial weights, and the training instances were presented in a different random order each training iteration (or epoch). For all learning rates, the same random number seeds were used, so the initial weights and order of presentation of the training instances were identical.

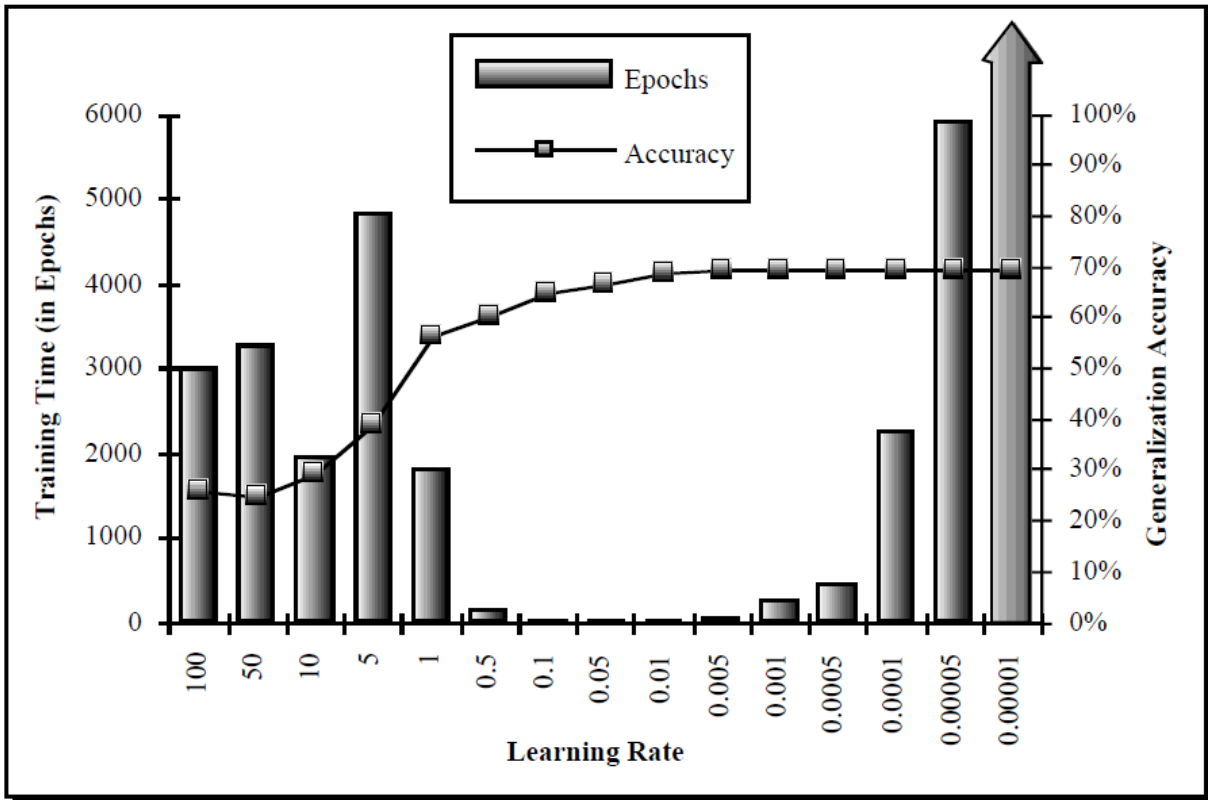

Figure 3.2: Training time (in epochs) and generalization accuracy for each learning rate, on a digit speech recognition task [19].

Based on the digit speech recognition task experiment result, learning rate 0.01 and 0.3 will produce high accuracy when minimum epoch is trained and tested.

CHAPTER 4

RESULT, ANALYSIS AND COMPARISION

4.1 Result Analysis and Comparison

4.1.1 Binary Classification

For the binary classification of the NSL-KDD dataset for detecting the intrusion we have two output node and 122 input node because the 41 features of NSL-KDD dataset have been converted to 122 dimensional features through numericalization and normalization process. The number of epochs for each of the training is 800. For each of the training 4, 8, 16 and 32 layer neural network with 1 hidden layer and learning rate 0.01 and 0.3. The observed value on the NSL-KDD dataset is shown in following tables

Table 4.1: The accuracy and training time (second) of RNN-IDS with different layer and learning rate for binary classification

	Accuracy	Time
Layer 4 with Learning Rate 0.01	81.43	2700
Layer 8 with Learning Rate 0.01	84.12	2525
Layer 16 with Learning Rate 0.01	87.73	2647
Layer 32 with Learning Rate 0.01	87.48	2807
Layer 4 with Learning Rate 0.3	66.83	6918
Layer 8 with Learning Rate 0.3	61.56	7123
Layer 16 with Learning Rate 0.3	64.63	7023
Layer 32 with Learning Rate 0.3	64.42	7393

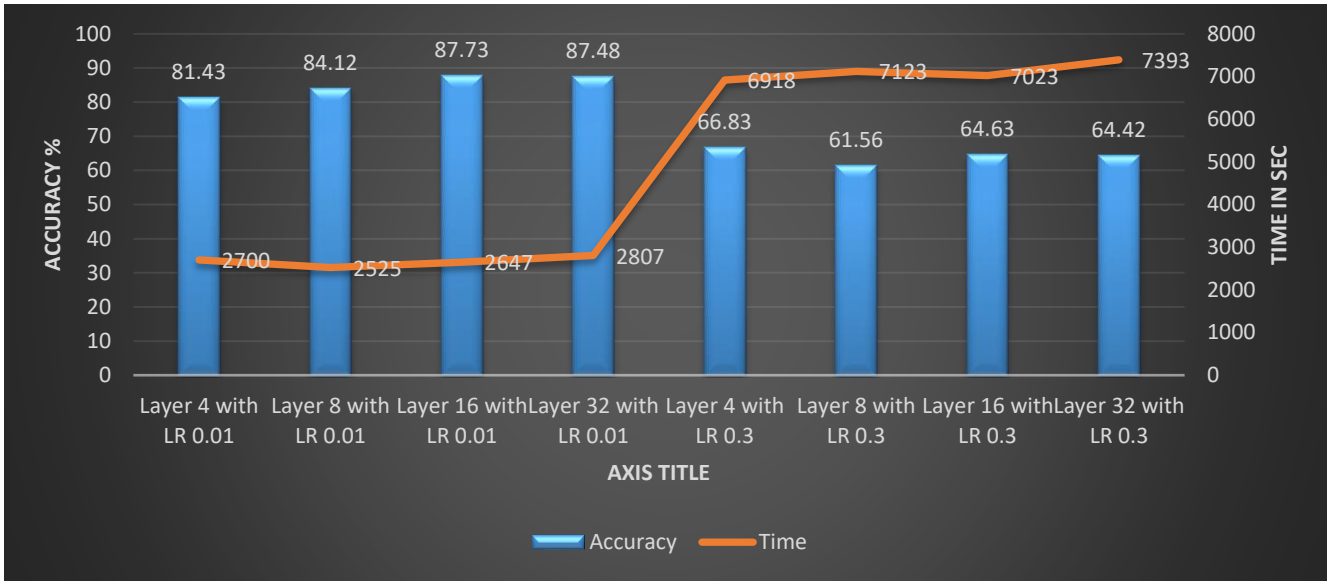

Figure 4.1: The accuracy and training time (second) of RNN-IDS with different layer and learning rate for binary classification

Table 4.2: Confusion matrix of 2-category classification on KDDTest+ with LR 0.01

	TP	FP	TN	FN
Layer 4 with Learning Rate 0.01	17453	5715	337	37
Layer 8 with Learning Rate 0.01	18407	4761	338	36
Layer 16 with Learning Rate 0.01	19259	3909	338	36
Layer 32 with Learning Rate 0.01	19224	3944	338	36

Table 4.3: The testing of RNN-IDS based on KDDTest+ with LR 0.01

	Accuracy
Layer 4 with Learning Rate 0.01	80.6
Layer 8 with Learning Rate 0.01	84.4
Layer 16 with Learning Rate 0.01	87.8
Layer 32 with Learning Rate 0.01	87.7

Table 4.4: Confusion matrix of 2-category classification on KDDTest-21 with LR 0.01

	TP	FP	TN	FN
Layer 4 with Learning Rate 0.3	12224	6792	573	14
Layer 8 with Learning Rate 0.3	13101	5915	572	15
Layer 16 with Learning Rate 0.3	14124	4892	573	14
Layer 32 with Learning Rate 0.3	13933	5083	572	15

Table 4.5: The testing of RNN-IDS based on KDDTest-21 with LR 0.01

	Accuracy
Layer 4 with Learning Rate 0.01	66.8
Layer 8 with Learning Rate 0.01	70.7
Layer 16 with Learning Rate 0.01	75.2
Layer 32 with Learning Rate 0.01	74.2

Table 4.6: Confusion matrix of 2-category classification on KDDTest+ with LR 0.3

	TP	FP	TN	FN
Layer 4 with Learning Rate 0.3	13579	9589	342	32
Layer 8 with Learning Rate 0.3	13268	9900	341	33
Layer 16 with Learning Rate 0.3	11556	11612	384	26
Layer 32 with Learning Rate 0.3	12174	10994	345	29

Table 4.7: The testing of RNN-IDS based on KDDTest+ with LR 0.3

	Accuracy
Layer 4 with Learning Rate 0.3	65.3
Layer 8 with Learning Rate 0.3	64
Layer 16 with Learning Rate 0.3	57.3
Layer 32 with Learning Rate 0.3	59.7

Table 4.8: Confusion matrix of 2-category classification on KDDTest-21 with LR 0.3

	TP	FP	TN	FN
Layer 4 with Learning Rate 0.3	8796	10226	575	12
Layer 8 with Learning Rate 0.3	8850	10166	581	6
Layer 16 with Learning Rate 0.3	7711	11305	584	13
Layer 32 with Learning Rate 0.3	8214	10802	583	4

Table 4.9: The testing of RNN-IDS based on KDDTest-21 with LR 0.3

	Accuracy
Layer 4 with Learning Rate 0.3	51.6
Layer 8 with Learning Rate 0.3	51.8
Layer 16 with Learning Rate 0.3	46.8
Layer 32 with Learning Rate 0.3	49

4.1.2 Multiclass Classification

For multiclass intrusion classification the number of epochs for each of the training is800. For each of the training 4, 8, 16 and 32 layer neural network with 1 hidden layer and learning rate 0.01 and 0.3. The observed value on the NSL-KDD dataset is shown in following tables

Table 4.10: The accuracy and training time (second) of RNN-IDS with different layer and learning rate for multiclass classification

	Accuracy	Time
Layer 4 with Learning Rate 0.01	85.58	3305
Layer 8 with Learning Rate 0.01	89.56	3185
Layer 16 with Learning Rate 0.01	93.72	3674
Layer 32 with Learning Rate 0.01	91.48	4837
Layer 4 with Learning Rate 0.3	71.83	9618
Layer 8 with Learning Rate 0.3	66.56	9153
Layer 16 with Learning Rate 0.3	70.63	8993
Layer 32 with Learning Rate 0.3	70.42	9063

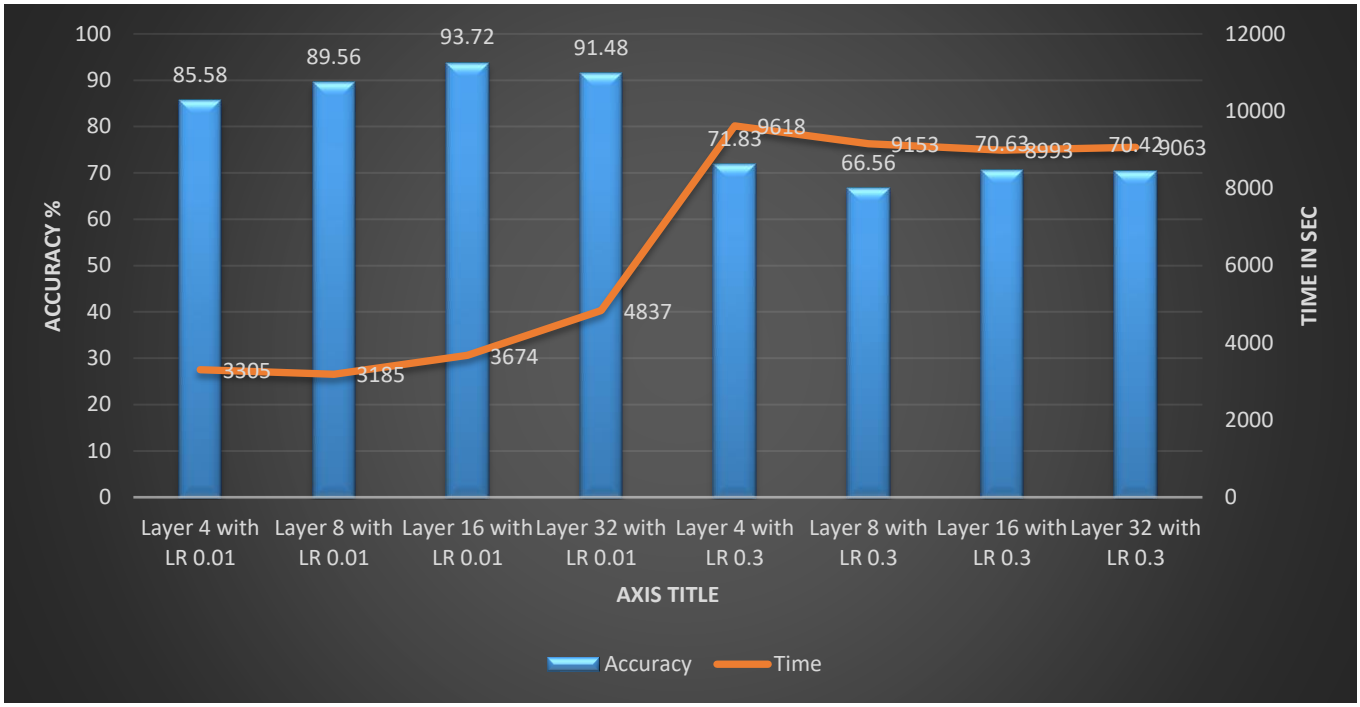

Figure 4.2: The accuracy and training time (second) of RNN-IDS with different layer and learning rate for multiclass classification

Table 4.11: Confusion matrix of multi-class classification on KDDTest+ with LR 0.01

	TP	FP	TN	FN
Layer 4 with Learning Rate 0.01	18923	3462	387	51
Layer 8 with Learning Rate 0.01	18385	4761	364	30
Layer 16 with Learning Rate 0.01	20419	3823	376	41
Layer 32 with Learning Rate 0.01	18523	2352	398	52

Table 4.12: The testing of RNN-IDS based on KDDTest+ with LR 0.01 for multi-class

	Accuracy
Layer 4 with Learning Rate 0.01	82.4
Layer 8 with Learning Rate 0.01	85.51
Layer 16 with Learning Rate 0.01	89.76
Layer 32 with Learning Rate 0.01	88.9

Table 4.13: Confusion matrix of multi-class classification on KDDTest-21 with LR 0.01

	TP	FP	TN	FN
Layer 4 with Learning Rate 0.3	11524	7562	535	20
Layer 8 with Learning Rate 0.3	13101	6975	585	22
Layer 16 with Learning Rate 0.3	15124	3812	570	24
Layer 32 with Learning Rate 0.3	12833	4373	572	35

Table 4.14: The testing of RNN-IDS based on KDDTest-21 with LR 0.01for multi-class

	Accuracy
Layer 4 with Learning Rate 0.01	70.53
Layer 8 with Learning Rate 0.01	74.6
Layer 16 with Learning Rate 0.01	79.9
Layer 32 with Learning Rate 0.01	79.3

Table 4.15: Confusion matrix of multi-class classification on KDDTest+ with LR 0.3

	TP	FP	TN	FN
Layer 4 with Learning Rate 0.3	13989	10089	398	45
Layer 8 with Learning Rate 0.3	14628	9834	375	30
Layer 16 with Learning Rate 0.3	11986	11361	401	35
Layer 32 with Learning Rate 0.3	10424	11404	390	39

Table 4.16: The testing of RNN-IDS based on KDDTest+ with LR 0.3 for multi-class

	Accuracy
Layer 4 with Learning Rate 0.3	69.23
Layer 8 with Learning Rate 0.3	68.66
Layer 16 with Learning Rate 0.3	61.39
Layer 32 with Learning Rate 0.3	60.71

Table 4.17: Confusion matrix of multi-class classification on KDDTest-21 with LR 0.3

	TP	FP	TN	FN
Layer 4 with Learning Rate 0.3	7756	10345	675	18
Layer 8 with Learning Rate 0.3	7830	10963	587	10
Layer 16 with Learning Rate 0.3	6611	13365	598	25
Layer 32 with Learning Rate 0.3	9754	12452	582	7

Table 4.18: The testing of RNN-IDS based on KDDTest-21 with LR 0.3 for multi-class

	Accuracy
Layer 4 with Learning Rate 0.3	58.27
Layer 8 with Learning Rate 0.3	59.73
Layer 16 with Learning Rate 0.3	51.02
Layer 32 with Learning Rate 0.3	50.53

4.1.3 Comparison

In [18], the results obtained by J48, Naive Bayesian, Random Forest, Multi-layer Perceptron, and Support Vector Machine which is shown in following table .These results are all based on the same benchmark – the NSL-KDD dataset.

Table 4.19: Detection Rate in binary classification for different algorithm [18]

DataSet	Algorithms		
	Naïve Bayes	Multilayer Perceptron	SVM
KDDTest+	76.56 %	77.41 %	69.52 %
KDDTest-21	55.77 %	57.34	42.29 %

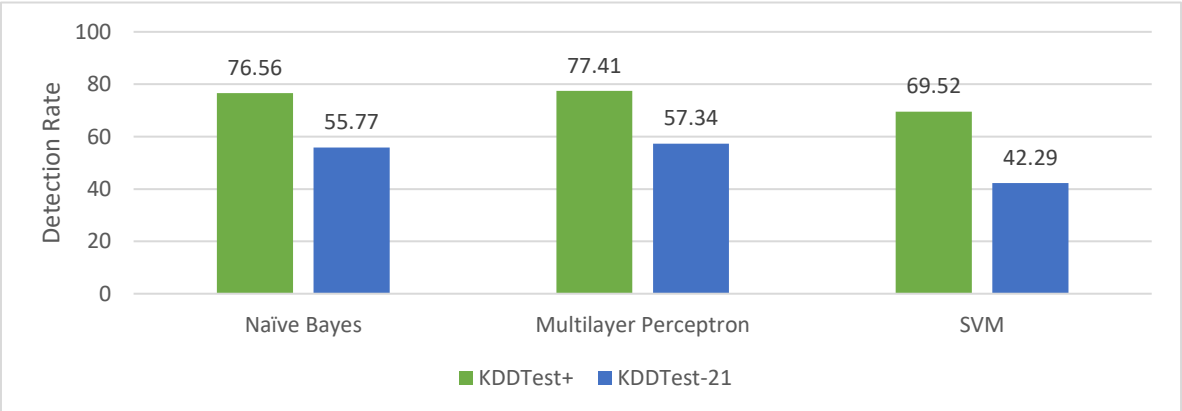

Figure 4.3: Performance of J48, Naïve Bayes, RF, MLP and SVM based on NSL-KDD dataset for binary classification

The overall comparison with the performance of RNN-IDS in binary classification is shown in following figure 4.4.

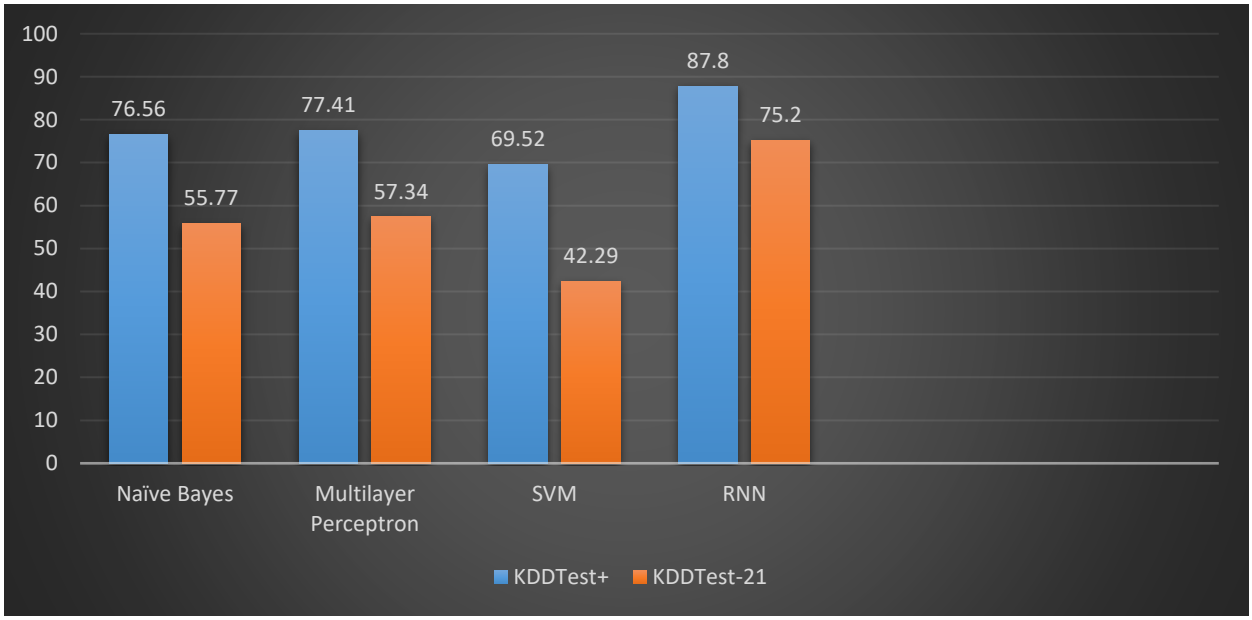

Figure 4.4: Performance of RNN-IDS and the other models in the binary classification.

Table 4.20: Detection Rate in multiclass classification for different algorithm [18]

DataSet	Algorithms		
	Naïve Bayes	Multilayer Perceptron	SVM
KDDTest+	74.4 %	78.1 %	74 %
KDDTest-21	55.77 %	58.4 %	50.7 %

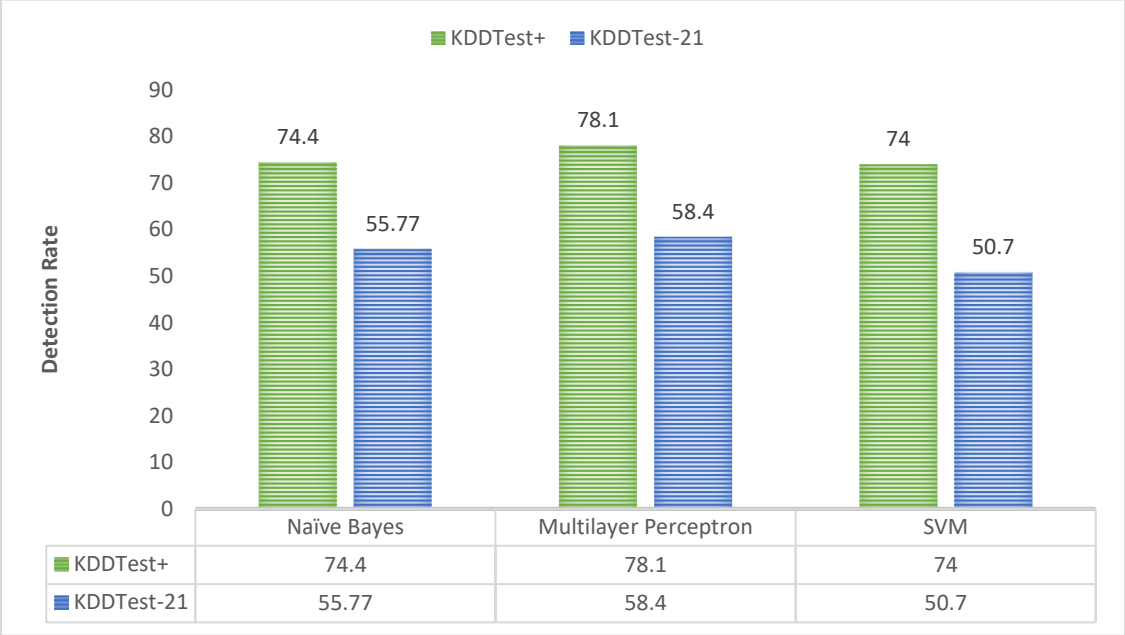

Figure 4.5: Performance of J48, Naïve Bayes, RF, MLP and SVM based on NSL-KDD dataset in multiclass classification

The overall comparison with the performance of RNN-IDS in binary classification is shown in following figure 4.6.

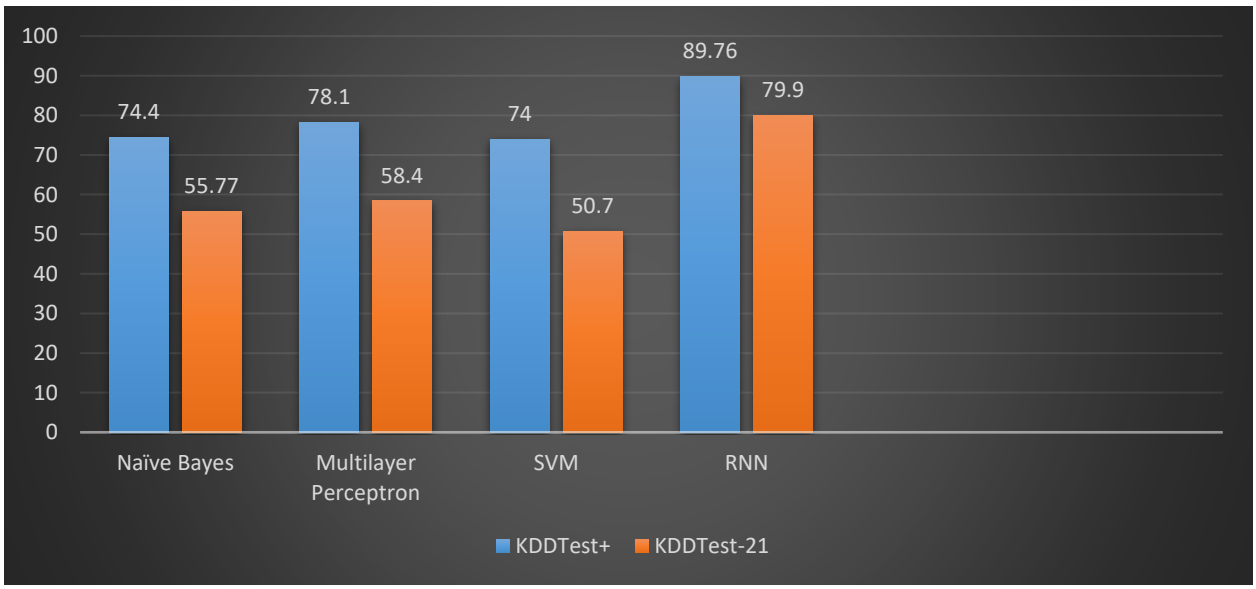

Figure 4.6: Performance of RNN-IDS and the other models in the multiclass classification.

CHAPTER 5

CONCLUSION

Based on the same benchmark dataset i.e. using KDDTrain+ as the training set and KDDTest+ and KDDTest-21 as the testing set, the experimental results show that for both binary and multiple classification, the intrusion detection model of RNN-IDS training through the training set has higher accuracy than the other machine learning methods i.e. Naive Bayesian, Support Vector Machine and Multilayer Perceptron. For testing and training of RNN-IDS the training set and testing set data have been changed with certain values which shows no significance difference in the accuracy of the RNN-IDS algorithm.

In the future research, Bidirectional RNNs algorithm in the field of intrusion detection will be studied based on the LSTM model.

Reference

[1] C. F. Tsai, Y. F. Hsu, C. Y. Lin, and W. Y. Lin, Intrusion Detection by Machine Learning: A Review," Expert Systems with Applications, 2009.

[2] Y. LeCun, Y. Bengio, and G. Hinton, ``Deep learning," Nature, May 2015.

[3] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, Self-taught Learning: Transfer Learning from Unlabeled Data," in Proceedings of the 24th International Conference on Machine Learning, ICML '07, (New York, NY, USA), 2007.

[4] A. Karpathy, The unreasonable effectiveness of recurrent neural networks. Andrej Karpathy Blog. [Online]. Available: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[5] X. Peng, L.Wang, X.Wang, and Y. Qiao, ``Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice," Comput. Vis. Image Understand., vol. 150, Sep. 2016.

[6] H. Soliman, N. Hikal, A. Sakar," A comparative performance evaluation of intrusion detection techniques for hierarchical wireless sensor networks", Egyptian Informatics Journal (2012) , Communications Engineering Dept., Faculty of Engineering, Mansoura University, Mansoura, Egypt

[7] A. Javaid, Q. Niyaz,W. Sun, and M. Alam, ``A deep learning approach for network intrusion detection system," presented at the 9th EAI Int. Conf. Bio-inspired Inf. Commun Technol. (BIONETICS), New York, NY, USA, May 2016.

[8] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, ``Deep learning approach for network intrusion detection in software defined networking," in Proc. Int. Conf. Wireless Netw. Mobile Commun. (WINCOM), Oct. 2016.

[9] M. Sheikhan, Z. Jadidi, and A. Farrokhi, ``Intrusion detection using reduced-size RNN based on feature grouping," Neural Comput. Appl., Sep. 2012.

[10] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep Learning and Its Applications to Machine Health Monitoring: A Survey," submitted to IEEE Transactions on Neural Networks and Learning Systems, vol. 14, no. 8, Dec 2016.

[11] K. Alrawashdeh and C. Purdy, “Toward an Online Anomaly Intrusion Detection System Based on Deep Learning,” in 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA). Anaheim, California, USA: IEEE, Dec 2016.

[12] K. Jin, S. Nara, S. Y. Jo, and K. Sang, “Method of intrusion detection using deep neural network,” in 2017 IEEE International Conference on Big Data and Smart Computing (BigComp). Hong Kong, China: IEEE, Feb 2017.

[13] L. You, Y. Li, Y. Wang, J. Zhang, and Y. Yang, “A deep learning based RNNs model for automatic security audit of short messages,” in 2016 16th International Symposium on Communications and Information Technologies (ISCIT). Qingdao, China: IEEE, Sep 2016.

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

[15] B. Dong and X. Wang, “Comparison deep learning method to traditional methods using for network intrusion detection,” in 2016 8th IEEE International Conference on Communication Software and Networks (ICCSN). Beijing, China: IEEE, jun 2016

[16] H. Lee, N. Kim, and J. Lee, “Deep Neural Network Self-training Based on Unsupervised Learning and Dropout,” The International Journal of Fuzzy Logic and Intelligent Systems, vol. 17, Mar 2007

[17] T. Mahbod, B. Ebrahim ,L. Wei, and G. Ali A. “A Detailed Analysis of the KDD CUP 99 Data Set”, Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2009)

[18] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, ``A detailed analysis of the KDDCUP 99 data set," in Proc. IEEE Computer Intelligence Security. Defense Appl., Jul. 2009

[19] D. Randall Wilson and M. Tony, “The Need for Small Learning Rates on Large Problems” in proceedings of the 2001 International Joint Conference on Neural Networks (IJCNN’01), 115-119, 2001

[20] M. Mehdi, Y. Khalid and B. Seddik, “Mining network traffics for intrusion detection based on Bagging ensemble Multilayer Perceptron with Genetic algorithm optimization”,

IJCSNS International Journal of Computer Science and Network Security, VOL.18 No.5, May 2018.

[21] M. Saurabh and S. Neelam, “Intrusion Detection using Naïve Bayes Classifier with Feature Reduction” Department of Computer Science, Banasthali University, Procedia Technology 4, 119-128, 2012

[22] J. Jayshree and R. Leena, “Intrusion Detection System using Support Vector Machine” International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868, 2013

Appendix

The NSL-KDD dataset each feature value separated by comma (,)

0,tcp,ftp_data,SF,491,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,1
50,25,0.17,0.03,0.17,0.00,0.00,0.00,0.05,0.00,normal,20

0,udp,other,SF,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,13,1,0.00,0.00,0.00,0.00,0.08,0.15,0.00,2
55,1,0.00,0.60,0.88,0.00,0.00,0.00,0.00,0.00,normal,15

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,123,6,1.00,1.00,0.00,0.00,0.05,0.07,0.00,25
5,26,0.10,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19

0,tcp,http,SF,232,8153,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,5,0.20,0.20,0.00,0.00,1.00,0.00,0.00,3
0,255,1.00,0.00,0.03,0.04,0.03,0.01,0.00,0.01,normal,21

0,tcp,http,SF,199,420,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,30,32,0.00,0.00,0.00,0.00,1.00,0.00,0.09,
255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,121,19,0.00,0.00,1.00,1.00,0.16,0.06,0.00
,255,19,0.07,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,166,9,1.00,1.00,0.00,0.00,0.05,0.06,0.00,25
5,9,0.04,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,117,16,1.00,1.00,0.00,0.00,0.14,0.06,0.00,2
55,15,0.06,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,remote_job,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,270,23,1.00,1.00,0.00,0.00,0.09,0.05,0.
00,255,23,0.09,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,133,8,1.00,1.00,0.00,0.00,0.06,0.06,0.00,25
5,13,0.05,0.06,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,205,12,0.00,0.00,1.00,1.00,0.06,0.06,0.00
,255,12,0.05,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,199,3,1.00,1.00,0.00,0.00,0.02,0.06,0.00,25
5,13,0.05,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21