NETWORK INTRUSION DETECTION SYSTEM

A PROJECT REPORT

Submitted by

HARISH B

(2019202016)

A final report of the project
submitted to the Faculty of

INFORMATION AND COMMUNICATION ENGINEERING

in partial fulfillment for the award of the degree
of

MASTER OF COMPUTER APPLICATIONS
in

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600 025

JUNE 2022

ANNA UNIVERSITY

CHENNAI - 600 025

BONA FIDE CERTIFICATE

Certified that this project report titled **NETWORK INTRUSION DETECTION SYSTEM** is the bona fide work of **HARISH B (2019202016)** who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

PLACE: Chennai
DATE: 22-06-2022

Ms. B SIVA SHANKARI
TEACHING FELLOW
PROJECT GUIDE
DEPARTMENT OF IST, CEG
ANNA UNIVERSITY
CHENNAI 600025

COUNTERSIGNED

Dr. S SRIDHAR
HEAD OF THE DEPARTMENT
DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600025

ABSTRACT

The Machine Learning based Network Intrusion Detection System detects anomalies through ML algorithms by analysing behaviours of packets. An Intrusion Detection System is primarily used for protection of network and information system. It monitors the operation of a network. Intrusion Detection System monitors network of computers for attacks that are aimed at stealing information. Applying Machine Learning can result in low False Alarm Rate and high detection rate. Machine Learning based Network Intrusion Detection System learns the characteristics of attack traffic based on training data. This approach can provide more robust and more accurate classification with the same classification dataset compared to existing approaches, it will be used as one of the feasible solutions to overcome weakness and limitation of existing Machine Learning based Network Intrusion Detection System. Decision Tree is used for classification and to predict data as normal or intrusive. Model performance was analysed on features extracted from the KDD dataset. In the case of Network Intrusion Detection System models, the network traffic instance will be predicted to belong to either benign (normal) or attack class.

TAMIL ABSTRACT

மெஷின் லேர்னிங் அடிப்படையிலான நெட்வொர்க் ஊடுருவல் கண்டறிதல் அமைப்பு, பாக்கெட்டுகளின் நடத்தைகளை பகுப்பாய்வு செய்வதன் மூலம் எம்எல் அல்காரிதம்கள் மூலம் முரண்பாடுகளைக் கண்டறிகிறது. ஒரு ஊடுருவல் கண்டறிதல் அமைப்பு முதன்மையாக நெட்வொர்க் மற்றும் தகவல் அமைப்பின் பாதுகாப்பிற்காகப் பயன்படுத்தப்படுகிறது. இது நெட்வொர்க்கின் செயல்பாட்டை கண்காணிக்கிறது. ஊடுருவல் கண்டறிதல் அமைப்பு தகவல்களைத் திருடுவதை நோக்கமாகக் கொண்ட தாக்குதல்களுக்கு கணினிகளின் வலையமைப்பைக் கண்காணிக்கிறது. மெஷின் லேர்னிங்கைப் பயன்படுத்துவதால் குறைந்த தவறான அலாரம் வீதம் மற்றும் உயர் கண்டறிதல் விகிதம் ஏற்படலாம். இயந்திர கற்றல் அடிப்படையிலான நெட்வொர்க் ஊடுருவல் கண்டறிதல் அமைப்பு பயிற்சி தரவின் அடிப்படையில் தாக்குதல் போக்குவரத்தின் பண்புகளை கற்றுக்கொள்கிறது. இந்த அணுகுமுறை தற்போதுள்ள அணுகுமுறைகளுடன் ஒப்பிடும்போது அதே வகைப்பாடு தரவுத்தொகுப்புடன் மிகவும் வலுவான மற்றும் மிகவும் துல்லியமான வகைப்படுத்தலை வழங்க முடியும், இது தற்போதுள்ள இயந்திர கற்றல் அடிப்படையிலான நெட்வொர்க் ஊடுருவல் கண்டறிதல் அமைப்பின் பலவீனம் மற்றும் வரம்புகளை சமாளிக்க சாத்தியமான தீர்வுகளில் ஒன்றாகப் பயன்படுத்தப்படும். டிசிஷன் ட்ரீ வகைப்பாடு மற்றும் தரவை இயல்பான அல்லது ஊடுருவும் என கணிக்க பயன்படுகிறது. *KDD* தரவுத்தொகுப்பிலிருந்து பிரித்தெடுக்கப்பட்ட அம்சங்களில் மாதிரி செயல்திறன் பகுப்பாய்வு செய்யப்பட்டது. நெட்வொர்க் ஊடுருவல் கண்டறிதல் அமைப்பு மாதிரிகள் விஷயத்தில், நெட்வொர்க் ட்ராஃபிக் நிகழ்வு தீங்கற்ற *(சாதாரண)* அல்லது தாக்குதல் வகுப்பைச் சேர்ந்ததாகக் கணிக்கப்படும்.

ACKNOWLEDGEMENT

The satisfaction that accompanies the success would be incomplete without mentioning the names of people who made it possible.

I would like to express my earnest thanks to **Ms. B Siva Shankari**, Teaching Fellow, Department of Information Science and Technology, CEG, Anna University for her valuable guidance, encouragement and attitude that has driven the project work in a steady pace to a successful completion.

I would like to thank **Dr. S Sridhar**, Professor and Head, Department of Information Science and Technology, CEG, Anna University for his kind support.

I express my sincere thanks to the project committee, **Dr. Saswati Mukherjee** , Professor, Department of Information Science and Technology, **Dr. M Vijayalakshmi** , Associate Professor, Department of Information Science and Technology, **Dr. E Uma** , Assistant Professor, Department of Information Science and Technology, **Ms. P S Apirajitha** , Teaching Fellow, Department of Information Science and Technology, **Ms. C M Sowmya** , Teaching Fellow, Department of Information Science and Technology, for their valuable guidance that have led to the betterment of the project

Harish B

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

DT	Decision Tree
DL	Deep Learning
IDS	Intrusion Detection System
ML	Machine Learning
ML – NIDS	Machine Learning based Network Intrusion Detection System
NIDS	Network Intrusion Detection System

CHAPTER 1

INTRODUCTION

1.1 DOMAIN OF PROJECT

Network Security is an emerging field in the IT sector. As more devices are connected to the internet, the attack surface for hackers is steadily increasing. An IDS is the combination of two words “intrusion” and “detection system.” Intrusion refers to an unauthorized access to the information within a computer or network systems to compromise its integrity, confidentiality, or availability. While detection system is a security mechanism for the detection of such illegal activity. So, IDS is a security tool that constantly monitors the host and network traffic to detect any suspicious behavior that violates the security policy and compromises its confidentiality, integrity, and availability. The IDS will generate alerts about detected malicious behavior to the host or network administrators. The task is to mirror all the incoming and outgoing network traffic to NIDS for performing traffic monitoring to detect intrusions.

1.2 PROBLEM STATEMENT

Network Intrusion Detection Systems (NIDSs) using pattern matching have a fatal weakness in that they cannot detect new attacks because they only learn existing patterns and use them to detect those attacks.

1.3 MOTIVATION AND OBJECTIVE

The Objective of this project is to perform classification to improve detection of malicious traffic. Research which traffic features and machine

learning algorithms that are suitable for detecting different kinds of malicious traffic. Write scripts to extract those features. Train machine learning models from a labeled dataset with malicious traffic. Evaluate the performance of the different models and scripts.

1.4 ABOUT THE PROJECT

A NIDS developed using ML methods usually involves following three major steps that are Data preprocessing phase, Training phase and Testing phase. The dataset is first preprocessed to transform it into the format suitable to be used by the algorithm. This stage typically involves encoding and normalization. Sometimes, the dataset requires cleaning in terms of removing entries with missing data and duplicate entries, which is also performed during this phase. The preprocessed data is then divided randomly into two portions, the training dataset, and the testing dataset. The ML algorithm is then trained using the training dataset in the training phase. The time taken by the algorithm in learning depends upon the size of the dataset and the complexity of the proposed model. Normally, the training time for the ML models requires more training time due to its deep and complex structure. Once the model is trained, it is tested using the testing dataset and evaluated based on the predictions it made. In the case of NIDS models, the network traffic instance will be predicted to belong to either benign (normal) or attack class.

CHAPTER 2

LITERATURE SURVEY

This Chapter explains about the literature survey made on the existing system,analyzing the problem statements and issues with the existing system and proposed objectives for the new system.

J. Alikhanov, R. Jang, M. Abuhamad, D. Mohaisen, D. Nyang and Y. Noh [1], Machine Learning (ML) based Network Intrusion Systems (NIDSs) operate on flow features which are obtained from flow exporting protocols. Recent success of ML and Deep Learning (DL) based NIDS solutions assume such flow information (e.g., avg. packet size) is obtained from all packets of the flow. However, often in practice flow exporter is deployed on commodity devices where packet sampling is inevitable. As a result, applicability of such ML based NIDS solutions in the presence of sampling is an open question. In this study, we explore the impact of packet sampling on the performance and efficiency of ML-based NIDSs. Unlike previous work, our proposed evaluation procedure is immune to different settings of flow export stage. Hence, it can provide a robust evaluation of NIDS even in the presence of sampling. Through sampling experiments, we established that malicious flows with shorter size are likely to go unnoticed even with mild sampling rates such as 1/10 and 1/100. Next, using the proposed evaluation procedure we investigated the impact of various sampling techniques on (NIDS) detection rate and false alarm rate. Detection rate and false alarm rate is computed for three sampling rates, for four different sampling techniques and for three classifiers.

T. Kim and W. Pak [2], proposed that the types of ML-NIDS are packet-based methods that use packet data directly as features, and session-based methods that use statistical data for a logical group called a session instead of packets as features. The packet-based method can be classified in two ways: one detection method uses a single packet to detect a pattern for malicious data in every packet received, and the other detection method uses multiple packets, storing and combining packets belonging to the same session into one dataset that is used for detection. Both the single-packet detection method and the multi-packet detection method search for malicious code or patterns in the packet payloads. However, attacks exploiting normal packets, like a Distributed denial-of-service (DDoS), are hard to detect with the packet-based method, and the pattern-matching algorithm can easily be bypassed by adding random data to the payload. When using session features, it is impossible to bypass the NIDS just by adding some dummy data. In addition, regardless of the packet size or the length of the session, the size of the entire feature is always the same, so the session-based method is more advantageous than the packet-based method for handling large volumes of traffic. The NIDS using session features mostly uses machine learning algorithms to classify the received traffic. So far, various ML-NIDSs have been developed and are expected to overcome the weaknesses of the PM-NIDS. Inevitably, malicious users are developing various methods to bypass the ML-NIDS (largely divided into white box, gray box, and black box methods), depending on what information can be used.

D. Han et al. [3] Machine learning (ML), especially deep learning (DL) techniques have been increasingly used in anomaly based network intrusion detection systems. Many adversarial attacks have been proposed to evaluate the robustness of ML based NIDSs. Unfortunately, existing attacks mostly focused on feature space and/or white-box attacks, which make impractical assumptions in real-world scenarios, leaving the study on practical gray/black box attacks largely unexplored. To bridge this gap, we conduct the first systematic study

of the gray/black-box traffic-space adversarial attacks to evaluate the robustness of ML-based NIDSs. Our work outperforms previous ones in the following aspects: (i) practical the proposed attack can automatically mutate original traffic with extremely limited knowledge and affordable overhead while preserving its functionality; (ii)generic the proposed attack is effective for evaluating the robustness of various NIDSs using diverse ML/DL models and non payload based features; (iii) explainable we propose an explanation method for the fragile robustness of ML based (NIDS). Experimental results show our attack is effective with execution cost and the proposed defense method can effectively mitigate such attacks.

2.1 OBSERVATION FROM SURVEY

The literature survey helped to gain a better insight with reference to the various techniques used for the same along with their current performance limitations and corresponding improvement, Suggestions given by respective authors.

CHAPTER 3

SYSTEM DESIGN

This module consists system design of the project with its preliminary design such as overall Architecture diagram and process flow diagram which tells about the modules integration in the project.

3.1 SYSTEM ARCHITECTURE

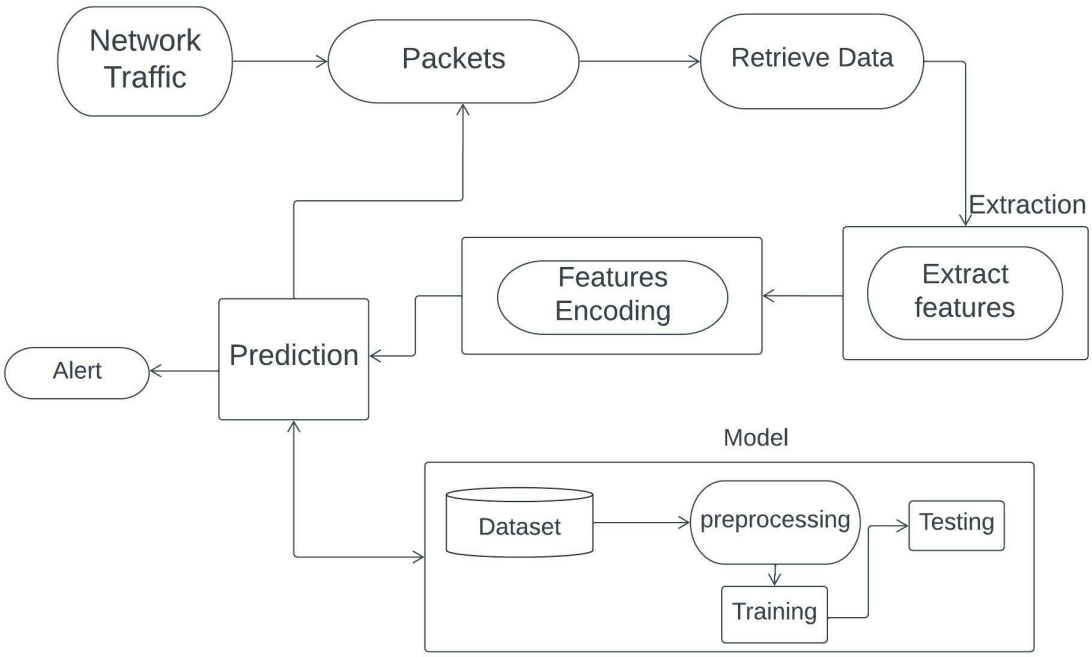

Figure 3.1: System Architecture

In Figure 3.1 first network interface is chosen to record the network traffic. To record the network traffic packet sniffer is used to collect the packets which flows in the network traffic. Packets are collected in pcap file for analysis. Packets will be kept flowing the network so packets will be recorded simultaneously.

Received data is stored as part file, then all the parts are merged into single file.Data is converted into features for analysis using Machine Learning Model.

3.2 DATASET DETAILS

DARPA Intrusion Detection Evaluation Program was prepared and managed by MIT Lincoln Labs. The objective was to survey and evaluate research in intrusion detection. A standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment, was provided.

Lincoln Labs set up an environment to acquire nine weeks of raw TCP dump data for a local-area network (LAN) simulating a typical U.S. Air Force LAN. They operated the LAN as if it were a true Air Force environment, but peppered it with multiple attacks.

The raw training data was about four gigabytes of compressed binary TCP dump data from seven weeks of network traffic. This was processed into about five million connection records. Similarly, the two weeks of test data yielded around two million connection records.

A connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows to and from a source IP address to a target IP address under some well defined protocol. Each connection is labeled as either normal, or as an attack, with exactly one specific attack type. Each connection record consists of about 100 bytes of data about the traffic.

3.3 CATEGORIES OF DATA

The data set consists of categories such as duration (duration of the connection), protocol type (e.g. tcp, udp, etc.), service (e.g., http, telnet, etc.), src bytes (number of data bytes from source to destination), dst bytes (number of data bytes from destination to source), flag (normal or error status of the connection), land (1 if connection is from/to the same host/port; 0 otherwise), wrong fragment (number of wrong fragments), urgent (number of urgent packets), hot (number of "hot" indicators), num_failed_logins (number of failed login attempts), logged_in (1 if successfully logged in; 0 otherwise), num_compromised (number of "compromised"), root_shell (1 if root shell is obtained; 0 otherwise), su_attempted(1 if "su root" command attempted; 0 otherwise), num_root (number of "root" accesses), num_file_creations (no of file creation operations), num_shells (number of shell prompts), num_access_files (number of operations on access control files), num_outbound_cmds (number of outbound commands in an ftp session), is_hot_login (1 if the login belongs to the "hot" list; 0 otherwise), is_guest_login (1 if the login is a "guest"login; 0 otherwise). From this dataset columns which are in categorical data is converted into numerical data.

3.4 LIST OF MODULES

1. Retrieve Features
2. Encoding of Categorical data
3. Classification using Decision Tree
4. Prediction using Trained Model

3.5 RETRIEVE FEATURES

Script is used to retrieve data from a unlabeled traffic which is created by packet sniffer. This takes in the pcap files and converts them into a readable file format that stacks each data part as flows where the first packet determines the forward (source to destination) and backward (destination to source) directions. Retrieved packets will be used to collect data for analysis. Packet details are stored in location for features analysis.

Figure 3.2: Retrieve features

3.6 ENCODING CATEGORICAL DATA

Features are extracted from each flow and stored as dataset format with attributes. From the extracted features protocol type(type of the protocol), service(network service on the destination), src bytes (number of data bytes from source to destination) features were selected for analysis purposes. Categorical data such as protocol, service, flag is converted into integer data for classification.

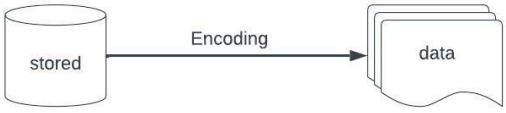

Figure 3.3: Encoding categorical data

3.7 CLASSIFICATION USING DECISION TREE

Here we use the concept of Decision Tree for classification methods. The last module includes the classification in which Machine Learning algorithms will be used. Sklearn is a open source library for numerical computation that makes machine learning faster and easier. Train and test data is taken from KDD Dataset and encoded to train the Decision Tree model.

Figure 3.4: Classification using Decision Tree

3.8 PREDICTION USING TRAINED MODEL

After Training and Testing the Decision Tree Classifier using KDD dataset, the model is saved using the pickle library. Then the saved model is loaded to predict the attack or normal from the given input. Input taken from the unlabelled file, the columns protocol, service, flag are encoded and used as features for prediction. The prediction is then decoded to categorical data, such as normal or an attack. Finally, the output is displayed on the console.

Figure 3.5: Prediction using trained model

CHAPTER 4

IMPLEMENTATION AND ALGORITHMS

This chapter explains in detail about the various modules in the system. Each module includes the input for the module, process flow for the module and output for the module in detail.

4.1 ENCODING

Encoding categorical data is a process of converting categorical data into integer format so that the data with converted categorical values can be provided to the different models for training and classification.

Steps:

1: Input: Features retrieved from packet data
2: Output: Encoded values
3: Get features from input
4: Start with second column
5: **for** *i* in column **do**
6: Get the *protocol*
7: Get the *service*
8: Get the $flag$
9: **end for**
10: change protocol to numeric
11: change service to numeric
12: change flag to numeric

4.2 CLASSIFICATION

Decision Tree is a supervised machine learning algorithm where all the decisions were made based on some conditions. The decision tree has a root node and leaf nodes extended from the root node. These nodes were decided based on some parameters like Gini index, entropy, information gain.

Steps:

Input: Data with encoded feature
Output: Prints attack or normal
if stopping condition(S, F) = true **then**
    Leaf = createNode()
    leafLabel = classify(s)
    return leaf
end if
root = createNode()
root.test condition = findBestSpilt(S,F)
V = v | v a possible outcomecfroot.test condition
for each value v ε V **do**
    $S_v = s|root.testcondition(s) = v, s\varepsilon S$
    $Child = TreeGrowth(S_v, F)$
    Add child as descent of root and label the edge {root → child} as v
end for
Add child as descent of root and label the edge root → child as v

---

Using Decision Tree Model, Classification algorithm is implemented to train and test the KDD dataset. With the trained model we can use the trained model for prediction.

CHAPTER 5

RESULTS AND ANALYSIS

This chapter consists of the details about the experiments that have been performed along with their outcomes. The detailed result of the project is also portrayed in this chapter.

5.1 RESULTS

ENCODING

```
Raw data:
   0    1     2   3    4      5   6   7   8   9   ...  32   33   34   35   36  \
0   0  tcp  http  SF  215  45076   0   0   0   0  ...   0  0.0  0.0  0.0  0.0
1   0  tcp  http  SF  162   4528   0   0   0   0  ...   1  1.0  0.0  1.0  0.0

    37   38   39   40       41
0  0.0  0.0  0.0  0.0  normal.
1  0.0  0.0  0.0  0.0  normal.

[2 rows x 42 columns]
Encoded data:
   0   1   2   3    4      5   6   7   8   9   ...  32   33   34   35   36  \
0   0   0   0   0  215  45076   0   0   0   0  ...   0  0.0  0.0  0.0  0.0
1   0   0   0   0  162   4528   0   0   0   0  ...   1  1.0  0.0  1.0  0.0

    37   38   39   40       41
0  0.0  0.0  0.0  0.0  normal.
1  0.0  0.0  0.0  0.0  normal.

[2 rows x 42 columns]
```

Figure 5.1: Training using dataset

Categorical data in each row has been encoded into integer values. Encoding are always in range between 0 and n(categories)-1.

PREDICTION

```
5   ['normal']
0,udp,domain_u,SF,29,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0.00,0.00,0.00,0.00,0.50,1.00,0.00,10,3,0.30,0.30,0.30,0.00,0.00,0.0
0,0.00,0.00

6   ['normal']
0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,508,508,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,1.00,0.00,0
.00,0.00,0.00,0.00

7   ['dos']
0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,509,509,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,1.00,0.00,0
.00,0.00,0.00,0.00

8   ['dos']
0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,510,510,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,1.00,0.00,0
.00,0.00,0.00,0.00

9   ['dos']
0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,510,510,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,1.00,0.00,0
.00,0.00,0.00,0.00

10   ['dos']
0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,1.00,0.00,0
.00,0.00,0.00,0.00
```

Figure 5.2: Prediction of Packets

Prediction has been made using trained decision tree model. Label is printed for each flow.

5.2 ANALYSIS USING DECISION TREE

	precision	recall	f1-score	support
dos	1.00	1.00	1.00	1165674
normal	1.00	1.00	1.00	291128
probe	1.00	1.00	1.00	12370
r2l	0.98	0.95	0.96	346
u2r	0.67	0.83	0.74	12
accuracy			1.00	1469530
macro avg	0.93	0.96	0.94	1469530
weighted avg	1.00	1.00	1.00	1469530

```
Accuracy Score: 0.9999564486604561
Classifier Training time = 29.437461853027344
Classifier Prediction time = 0.2781538963317871
```

Figure 5.3: Accuracy for Decision Tree

Accuracy is shown using trained decision tree model. Training and Testing Time is printed for the KDD Dataset.

PRECISION FOR 100% DATASET

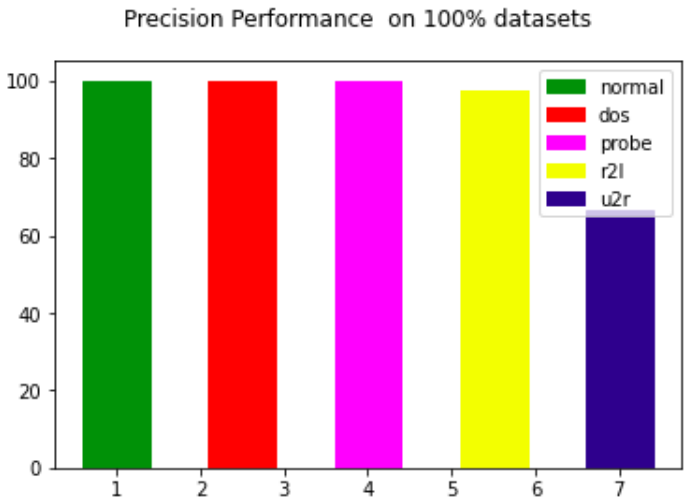

Figure 5.4: Precision for 100% dataset

Precision for each labels is shown using decision tree model. Each label is classified using KDD.

F1 SCORE FOR 100% DATASET

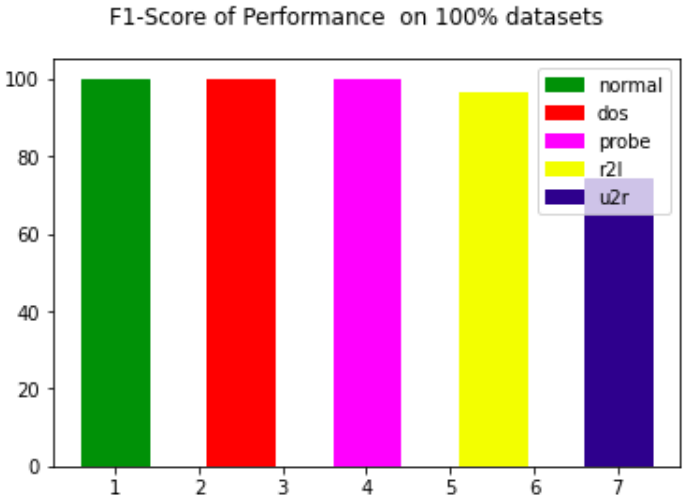

Figure 5.5: F1-Score for 100% dataset

F1 score for each labels is shown using decision tree model. Each label is classified using KDD.

SUPPORT FOR 100% DATASET

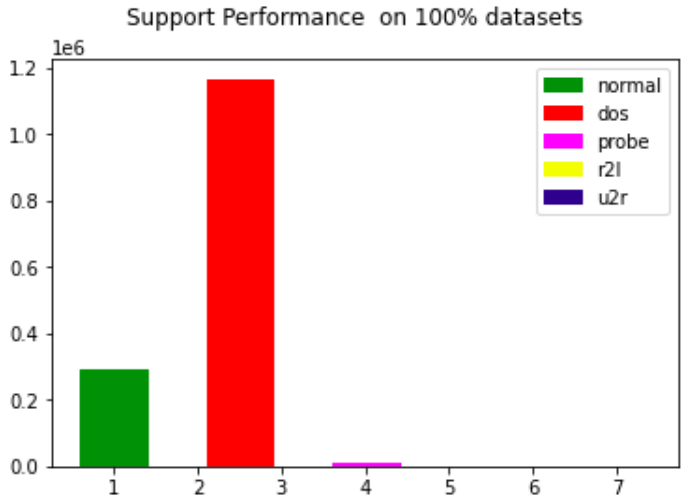

Figure 5.6: Support for 100% dataset

F1 score for each labels is shown using decision tree model. Each label is classified using KDD.

RECALL FOR 100% DATASET

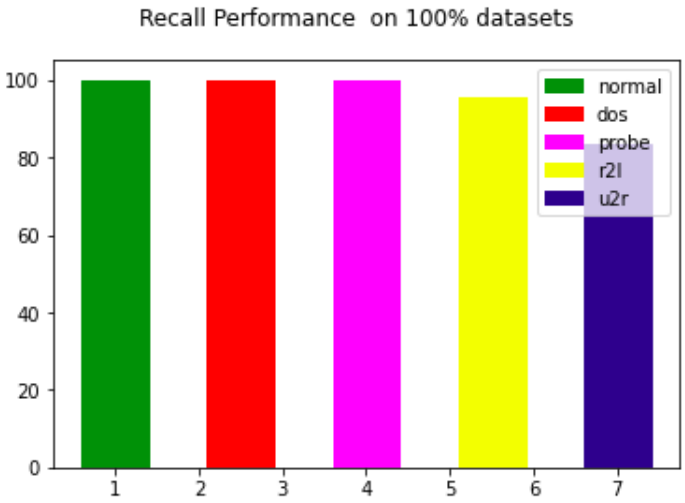

Figure 5.7: Recall for 100% dataset

Recall for each labels is shown using decision tree model. Each label is classified using KDD.

CHAPTER 6

CONCLUSION AND FUTURE WORK

The most important thing in the NIDS is the training dataset used to create the classifier model. It is impossible to obtain a training dataset including all network intrusions that occur in the wild. Rather, it is important to find a way to accurately detect an intrusion by utilizing an existing dataset, even if the intrusion data it contains are insufficient. Using KDD dataset, it has proven that the weaknesses of the existing NIDS can be greatly improved. However, if multiple features are considered, the number of sessions that can be processed per second decreases. Also, for some classes, improvement of the detection rate is not big. Despite these weaknesses, it is a great advantage to be able to broadly expand the classification range in the feature space by using a dataset consisting of limited data. In addition, classification speed can also be improved, so it is expected that Machine Learning Model, when installed in actual NIDS equipment, will be of great help in keeping large networks safe.

As future work, it will be focused on how to extend this current result to support multiple features. If the solution is successfully found, NIDS can maximize the classification detection rate without deteriorating the classification speed.

REFERENCES

[1] M. Abuhamad D. Mohaisen D. Nyang J. Alikhanov, R. Jang and Y. Noh. Investigating the effect of traffic sampling on machine learning-based network intrusion detection approaches. *IEEE Access*, 10:5801–5823, 2022.

[2] T. Kim and W. Pak. Robust network intrusion detection system based on machine-learning with early classification. *IEEE Access*, 10:10754–10767, 2022.

[3] Ying Zhong Wenqi Chen Jiahai Yang Shuqiang Lu Xingang Shi Xia Yin Dongqi Han, Zhiliang Wang. Evaluating and improving adversarial robustness of machine learning-based network intrusion detectors. *IEEE Journal on Selected Areas in Communications*, 39:2632–2647, 2021.

[4] M. Ambuludi D. Vallejo-Huanga and P. Morillo. Empirical exploration of machine learning techniques for detection of anomalies based on nids. *IEEE Latin America Transactions*, 19:772–779, 2021.

[5] D. Kotani D. Li and Y. Okabe. Improving attack detection performance in nids using gan. *IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, 19:817–825, 2020.

[6] S. Kim D. Shin J. Yoo, B. Min and D. Shin. Study on network intrusion detection method using discrete pre-processing method and convolution neural network. *IEEE Access*, 9:142348–142361, 2021.