KANTIPUR ENGINEERING COLLEGE

(Affiliated to Tribhuvan University)

Dhapakhel, Lalitpur

[Subject Code: CT755]

A MAJOR PROJECT FINAL REPORT ON

NETWORK INTRUSION DETECTION SYSTEM USING LSTM

Submitted by:

Aman Devkota [KAN076BCT010]

Ankur Karmacharya [KAN076BCT013]

Prashad Adhikary [KAN076BCT056]

A MAJOR PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR IN COMPUTER ENGINEERING

Submitted to:

Department of Computer and Electronics Engineering

February, 2024

NETWORK INTRUSION DETECTION SYSTEM USING LSTM

Submitted by:

Aman Devkota [KAN076BCT010]

Ankur Karmacharya [KAN076BCT013]

Prashad Adhikary [KAN076BCT056]

Supervised by:

Babu Ram Dawadi

Supervisor’s Designation

Second Line of Designation (if required)

A MAJOR PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR IN COMPUTER ENGINEERING

Submitted to:

Department of Computer and Electronics Engineering

Kantipur Engineering College

Dhapakhel, Lalitpur

February, 2024

ABSTRACT

blablablaa

Keywords *– LSTM, SDN, ,*

ACKNOWLEDGMENT

We would like to express sincere gratitude to Department head Er. Rabindra Khati, Project Co-ordinator Er. Bishal Thapa and all the faculty members of Kantipur Engineering College for the continuous support during this project for their patience, motivation,enthusiasm, and immense knowledge. Their guidance helped us in all time of research, development and implementation of this project.

Finally we would like to thank our family and friends for all the support and encouragement.

Aman Devkota	[KAN076BCT010]
Ankur Karmacharya	[KAN076BCT013]
Prashad Adhikary	[KAN076BCT056]

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

DOS: Denial of Service

FN: False Negative

FP: False Positive

IDS: Intrusion Detection System

LSTM: Long Short Term Memory

R2L: Root to Local

TN: True Negative

TP: True Positive

U2R: User to Root

CHAPTER 1
INTRODUCTION
1.1 Background

With the increasingly deep integration of the Internet and society, the Internet is changing the way in which people live, study and work, but the various security threats that we face are becoming more and more serious. How to identify various network attacks, especially unforeseen attacks, is an unavoidable key technical issue. An Intrusion Detection System (IDS), a significant research achievement in the information security field, can identify an invasion, which could be an ongoing invasion or an intrusion that has already occurred. In fact, intrusion detection is usually equivalent to a classification problem, such as a binary or a multi class classification problem, i.e., identifying whether network traffic behavior is normal or anomalous, or a five-category classification problem, i.e., identifying whether it is normal or any one of the other four attack types: Denial of Service (DOS), User to Root (U2R), Probe (Probing) and Root to Local (R2L). In short, the main motivation of intrusion detection is to improve the accuracy of classifiers in effectively identifying the intrusive behavior.

Machine learning methodologies have been widely used in identifying various types of attacks, and a machine learning approach can help the network administrator take the corresponding measures for preventing intrusions. However, most of the traditional machine learning methodologies belong to shallow learning and often emphasize feature engineering and selection; they cannot effectively solve the massive intrusion data classification problem that arises in the face of a real network application environment. With the dynamic growth of data sets, multiple classification tasks will lead to decreased accuracy. In addition, shallow learning is unsuited to intelligent analysis and the forecasting requirements of high-dimensional learning with massive data. In contrast, deep learners have the potential to extract better representations from the data to create much better models. As a result, intrusion detection technology has experienced rapid development after falling into a relatively slow period.[1]

A well-known method of securing the network is through implementing an Intrusion Detection System (IDS). IDS was originally implemented in 1980. The main aim of their work was to introduce a mechanism which differentiates between benign activities from malicious ones. Further research was carried out to optimizing this methodology to aid monitoring the network traffic in case of attacks, this system is now known as Network Intrusion Detection System (NIDS) . In NIDS, the detection system is inspecting the incoming and outgoing network traffic from all hosts in real time and based on certain criteria, it can detect and identify the attack, then, take the suitable security measures to stop or block it, which significantly reduces the risk of damage to the network. However, due to the rapid increase in the complexity of the cyber-security attacks, the current methods used in NIDS are failing to sufficiently address this issue.[2]

IDSs can be divided into two categories according to the main detection technology: misuse detection and anomaly detection. Misuse detection is a knowledge-based detection technology. A misuse detection system needs to clearly define the features of the intrusion, then identify the intrusion by matching the rules. Misuse detection can achieve a high accuracy and low false alarm rate. However, it needs to build a feature library and cannot detect unknown attacks. In contrast, anomaly detection is a behavior-based detection technology. First, it needs to define the normal activities of a network, and then check whether the actual behavior has deviated from the normal activities. Anomaly detection needs only to define a normal state of a specific network, without prior knowledge of intrusion. Thus, it can detect unknown attacks, although there may be a high false alarm rate. At present, network structure is becoming more and more complicated, and intrusion methods are following the trend of diversification and complication, creating more challenges for IDSs.

The recurrent neural network (RNN) has failed to become a mainstream network model in the past few years due to difficulties in training and computational complexity. In recent years, with the development of deep learning theory, RNN began to enter a rapid development period. Currently, RNN has already been applied successfully to handwriting and speech recognition. The main feature of RNN is that it circulates information in a hidden layer which can remember information processed previously, leading to a structural advantage for the processing of time series information. Correspondingly, many intrusion behaviors can be abstracted as specific time series of events from the underlying network. So, RNN is considered suitable for building an IDS.[3]

1.2 Problem Statement

The existing Network Intrusion Detection Systems face challenges in accurately and efficiently detecting and preventing network intrusions, thereby compromising the overall security of computer networks. These challenges include high false-positive rates, limited scalability, inability to detect novel or sophisticated attacks, and the difficulty in distinguishing between legitimate and malicious network traffic. Addressing these issues is crucial to enhance the performance and reliability of NIDS, enabling proactive identification and prevention of network intrusions while minimizing false alarms.

1.3 Objectives

The primary objective of this project:

i. To develop a robust and accurate system that can effectively detect and notify network intrusions or malicious activities within a computer network.

1.4 Project Features

The project will be able to accomplish following:

- Anomaly Detection
- Real-time Monitoring
- Traffic Preprocessing

1.5 Application Scope

Network Intrusion Detection System has various applications in areas such as Enterprise networks, Internet Service Providers(ISPs), Cloud Computing environment, Government networks and so on. The application scope of network intrusion detection systems using deep neural networks extends beyond these areas, as network security is crucial in nearly every sector that relies on secure and reliable communication. By deploying these systems, organizations can proactively detect and respond to network intrusions, minimize the impact of attacks, and protect their assets, data, and operations from potential threats.

1.6 System Requirement

1.6.1 Development Requirements

1.6.1.1 Software Requirements

- Windows/Linux/Mac
- VMware/Virtualbox
- Mininet, POX controller
- Jupyter Notebook
- Python IDE

1.6.1.2 Hardware Requirements

- PC with at least 4-8 GB RAM
- Higher graphics of at least 2 GB

1.6.2 Deployment Requirements

1.6.2.1 Software Requirements

- Windows/Linux/Mac
- VMware/Virtualbox
- Mininet, POX controller

1.6.2.2 Hardware Requirements

- More than 1.5 GHz clock speed
- Minimum 4 GB RAM

1.7 Project Feasibility

1.7.1 Technical Feasibility

The technical feasibility assessment is focused on gaining in understanding of the present technical resources required by the system and their applicability to the expected needs of the proposed system. Regarding the proposed system, the technical requirement includes a PC.

1.7.2 Operational Feasibility

The user will not need any formal knowledge about programming so our project is operationally feasible.

1.7.3 Economic Feasibility

The purpose of the economic feasibility assessment is to determine the positive economic benefits to the user that the proposed system will provide. Most of the software used for the development is free. Thus, the project is economically feasible.

1.7.4 Schedule Feasibility

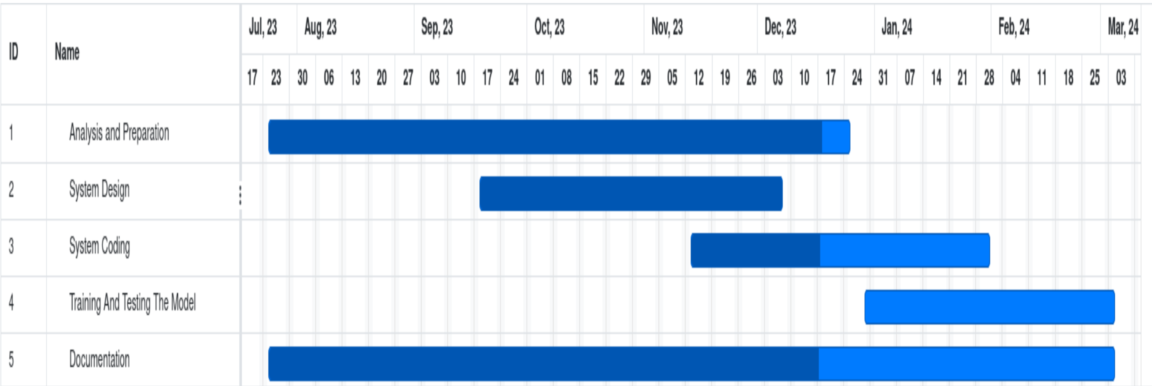

Figure 1.1: Gantt Chart

CHAPTER 2
LITERATURE REVIEW

2.1 Related Projects

2.1.1 Suricata

Suricata is a high performance, open source network analysis and threat detection software used by most private and public organizations, and embedded by major vendors to protect their assets. It was developed by the Open Information Security Foundation (OSIF) and is a free tool used by enterprises, small and large. The system uses a rule set and signature language to detect and prevent threats. Suricata can run on Windows, Mac, Unix and Linux. As we know intrusion detection “detects” and “alerts” a threat. In contrast, an intrusion prevention system also takes action on the event and attempts to block the traffic. Suricata can do both and also does well with deep packet inspection, making it perfect for pretty much any kind of standard security monitoring initiatives for a company.

2.1.2 Snort

Snort is the foremost Open Source Intrusion Prevention System (IPS) in the world. Snort IPS uses a series of rules that help define malicious network activity and uses those rules to find packets that match against them and generates alerts for users. Snort can be deployed inline to stop these packets, as well. Snort has three primary uses: As a packet sniffer like tcpdump, as a packet logger- which is useful for network traffic debugging, or it can be used as a full-blown network intrusion prevention system. Snort can be downloaded and configured for personal and business use alike.

2.2 Related Works

In the paper ”A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks” a deep learning approach for intrusion detection system using recurrent neural networks(RNN-IDS) was proposed and the performance of the model was compared with traditional machine learning classification methods in both binary and multiclass classification. The study found that the proposed model improved the accuracy of the intrusion detection and provided a new research method.[1]

In ”Using Deep Learning Techniques for Network Intrusion Detection”, Convolutional Neural Networks and Recurrent Neural Networks were used to design an intelligent detection system that was able to detect different network intrusions and the performance of this model was evaluated using various evaluation metrices to find the best model for the intrusion detection system. Among the tested learning techniques, CNN was found to have outperformed the other techniques with accuracy, F1 score,recall and precision.[2]

In the paper ”An Intrusion Detection System Using a Deep Neural Network With Gated Recurrent Units”, deep learning theory was applied to intrusion detection system and a model was developed that had automatic feature extraction. In this paper, the characteristics of the time-related intrusion was considered and a novel IDS that consisted of a recurrent neural network with gated recurrent units (GRU), multilayer perceptron (MLP), and softmax module was proposed. Experiments on the datasets used showed that the proposed system had leading performance. Comparative experiments showed that the GRU is more suitable as a memory unit for IDS than LSTM, and proved that it is an effective simplification and improvement of LSTM. Moreover, the bidirectional GRU can reach the best performance compared with the other methods.[3]

In the paper ”A Flow-Based Anomaly Detection Approach With Feature Selection Method Against DDoS Attacks in SDNs”, the aim is to provide an overview of the existing research and studies related to DDoS attack detection in SDN networks. It aims to identify the gaps in the current knowledge and highlight the need for further research in this area. It also helps to establish the theoretical foundation for the proposed approach and validate its effectiveness by comparing it with previous works. Additionally, it provides a context for understanding the significance and relevance of the research conducted in the document.[4]

In the paper ”Securing IoT and SDN systems using deep-learning based automatic intrusion detection”, Secured Automatic Two-level Intrusion Detection System (SATIDS) based on an improved Long Short-Term Memory (LSTM) network was proposed. Deep learning techniques, such as LSTM networks, have been utilized to detect and prevent network attacks. The use of LSTM networks in cyber defense has several benefits, including the ability to gain insights into threats and stay ahead of opponents. The SATIDS system aims to accurately identify suspicious activities and types of attacks in IoT and SDN networks. It utilizes an improved LSTM network that classifies traffic into normal or attack, determines the attack category, and defines the attack sub-type. The system has been trained and tested using realistic datasets, such as the ToN-IoT and InSDN datasets. The performance of the SATIDS system has been compared to other IDSs, and it has been found to achieve better performance in terms of accuracy, precision, F1-score, and detection rate. The system has shown high accuracy in differentiating between normal and anomalous traffic, as well as in detecting specific types of attacks, such as backdoor attacks and DDOS attacks. Overall, the proposed SATIDS system based on LSTM network offers an effective approach to intrusion detection in IoT and SDN networks. It provides accurate identification of attacks and can help enhance the security of these networks.[5]

CHAPTER 3

METHODOLOGY

3.1 Working Mechanism

The development of Network Intrusion Detection System involves major steps which is depicted in the diagram given below:

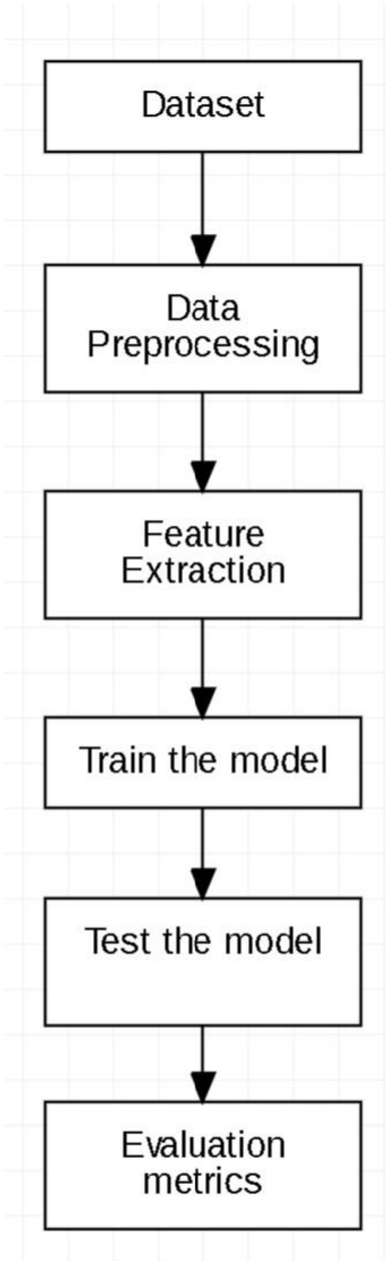

Figure 3.1: Working mechanism of Network Intrusion Detection System

3.1.1 Data set

InSDN is a comprehensive Software-Defined Network (SDN) dataset for Intrusion detection system evaluation. The new dataset includes the benign and various attack categories that can occur in different elements of the SDN standard. InSDN considers different attack, including DoS, DDoS, brute force attack, web applications, exploitation, probe, and botnet. Furthermore, the normal traffic in the generated data covers various popular application services such as HTTPS, HTTP, SSL, DNS, Email, FTP, SSH, etc. The dataset was generated by using four virtual machines (VMs). The first virtual machine is a Kali Linux one and represents the attacker server. The secondary machine is a Ubuntu 16.4 one, and acts on the ONOS controller. Third is an Ubuntu 16.4 machine to serve for the Mininet and OVS switch. The forth virtual machine is a Linux one based on metasploitable 2 to provide vulnerable services for demonstrating common vulnerabilities.[6]

SN	Features	SN	Features
1	Src Port	41	Pkt Len Min
2	Dst Port	42	Pkt Len Max
3	Protocol	43	Pkt Len Mean
4	Flow Duration	44	Pkt Len Std
5	Tot Fwd Pkts	45	Pkt Len Var
6	Tot Bwd Pkts	46	FIN Flag Cnt
7	TotLen Fwd Pkts	47	SYN Flag Cnt
8	TotLen Bwd Pkts	48	RST Flag Cnt
9	Fwd Pkt Len Max	49	PSH Flag Cnt
10	Fwd Pkt Len Min	50	ACK Flag Cnt
11	Fwd Pkt Len Mean	51	URG Flag Cnt
12	Fwd Pkt Len Std	52	CWE Flag Count
13	Bwd Pkt Len Max	53	ECE Flag Cnt
14	Bwd Pkt Len Min	54	Down/Up Ratio
15	Bwd Pkt Len Mean	55	Pkt Size Avg
16	Bwd Pkt Len Std	56	Fwd Seg Size Avg
17	Flow Byts/s	57	Bwd Seg Size Avg
18	Flow Pkts/s	58	Fwd Byts/b Avg
19	Flow IAT Mean	59	Fwd Pkts/b Avg
20	Flow IAT Std	60	Fwd Blk Rate Avg
21	Flow IAT Max	61	Bwd Byts/b Avg
22	Flow IAT Min	62	Bwd Pkts/b Avg
23	Fwd IAT Tot	63	Bwd Blk Rate Avg
24	Fwd IAT Mean	64	Subflow Fwd Pkts
25	Fwd IAT Std	65	Subflow Fwd Byts
26	Fwd IAT Max	66	Subflow Bwd Pkts
27	Fwd IAT Min	67	Subflow Bwd Byts
28	Bwd IAT Tot	68	Init Fwd Win Byts
29	Bwd IAT Mean	69	Init Bwd Win Byts
30	Bwd IAT Std	70	Fwd Act Data Pkts
31	Bwd IAT Max	71	Fwd Seg Size Min
32	Bwd IAT Min	72	Active Mean
33	Fwd PSH Flags	73	Active Std
34	Bwd PSH Flags	74	Active Max
35	Fwd URG Flags	75	Active Min
36	Bwd URG Flags	76	Idle Mean
37	Fwd Header Len	77	Idle Std
38	Bwd Header Len	78	Idle Max
39	Fwd Pkts/s	79	Idle Min
40	Bwd Pkts/s	80	Label

Figure 3.2: Features of InSDN dataset

3.1.2 Data Preprocessing

We used the InSDN dataset. Because the input should be a numeric matrix, we must convert some non-numeric features, into numeric form using label encoding. For the efficient training of neural networks, input data should be transformed by performing some pre-processing known as data normalization. It is used where inputs are widely divergent. Without such a process, networks would take a long time to train. Different schemes can be used to normalise the input data before it is fed to the input layer of neural network. We have used Z-score normalization to normalize the attributes of our dataset. Z-score normalization refers to the process of normalizing every value in a dataset such that the mean of all of the values is 0 and the standard deviation is 1. Mathematically,

$$Newvalue = \frac{x - \mu}{\lambda} \tag{3.1}$$

where x : Original value,

μ : Mean of data,

λ : Standard deviation of data.

3.1.3 Feature Extraction

Principal component analysis, or PCA, is a statistical procedure that allows you to summarize the information content in large data tables by means of a smaller set of "summary indices" that can be more easily visualized and analyzed. Algorithm for PCA is given as:

i. $\mu = \frac{1}{p}\sum_{k=1}^{p} x_k$, where, x_k is a pattern, $p =$ number of patterns, x is the feature matrix.

ii. Find the covariance matrix

$C = \frac{1}{p}\sum_{k=1}^{p}(x_k - \mu)(x_k - \mu)^T$ where T represents matrix transposition.

iii. Compute Eigen values λ_i and Eigen vectors v_i of covariance matrix

$Cv_i = \lambda_i v_i \qquad (i = 1, 2, 3...q), q =$ number of features

iv. Estimating high-valued Eigen vectors

(a) Arrange all the Eigen values (λ_i) in descending order.

(b) Choose a threshold value, θ

(c) Number of high-valued λ_i can be chosen so as to satisfy the relationship $(\sum_{i=1}^{s} \lambda_i)(\sum_{i=1}^{q} \lambda_i)^{-1} \geq \theta$, where $s =$ number of high valued λ_i chosen

(d) Select Eigen vectors corresponding to selected high valued λ_i

v. Extract low dimensional feature vectors (principal components) from raw feature matrix.

$P = V^T x$, where V is the matrix of principal components and x is the feature matrix.

3.1.4 Train and test the model

The model used in this project for NIDS is: Long Shortterm Memory (LSTM).

3.1.4.1 Long Short term Memory (LSTM)

LSTM is a popular deep learning technique in RNN for time series prediction. While standard RNNs outperform traditional networks in preserving information, they are not very effective in learning long term dependencies due to the vanishing gradient problem. An LSTM is well-suited to classify and/or predict time-series data. There are several architectures of LSTM units. A common architecture is composed of a memory cell, an input gate, an output gate and a forget gate. The mathematical formulation of the LSTM cell is given below:

$$f_t = \sigma(x_t W_f + H_{t-1} U_f) \tag{3.2}$$

$$o_t = \sigma(x_t W_o + H_{t-1} U_o) \tag{3.3}$$

$$S_t = \sigma(S_{t-1} * f_t + i_t * H_t^{'}) \tag{3.4}$$

$$i_t = \sigma(x_t W_i + H_{t-1} U_i) \tag{3.5}$$

$$H_t^{'} = tanh(x_t W_g + H_{t-1} U_g) \tag{3.6}$$

$$H_t = tanh(S_t) * o_t \tag{3.7}$$

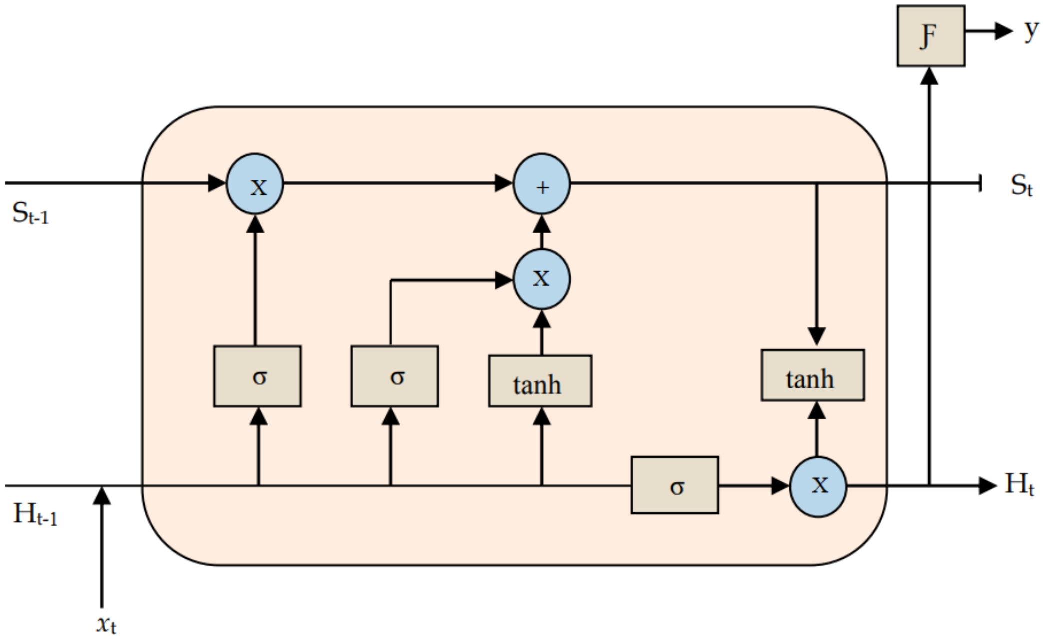

Figure 3.3: LSTM

3.1.5 Evaluation Metrics

The most important performance indicator (Accuracy, AC) of intrusion detection is used to measure the performance of the model. In addition to the accuracy, we introduce the detection rate and false positive rate. The True Positive (TP) is equivalent to those correctly rejected, and it denotes the number of anomaly records that are identified as anomaly. The False Positive (FP) is the equivalent of incorrectly rejected, and it denotes the number of normal records that are identified as anomaly. The True Negative (TN) is equivalent to those correctly admitted, and it denotes the number of normal records that are identified as normal. The False Negative (FN) is equivalent to those incorrectly admitted, and it denotes the number of anomaly records that are identified as normal.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.8}$$

Precision is the number of actual attacks as a proportion of the number classified as attacks.

$$Precision = \frac{TP}{TP + FP} \tag{3.9}$$

Predicted

	Classes	DDoS	Normal	Probe
Actual	DDoS	TN	FP	TN
	Normal	FN	TP	FN
	Probe	TN	FP	TN

Figure 3.4: Confusion Matrix

True Positive Rate shows the percentage of the number of records identified correctly over the total number of anomaly records.

$$TruePositiveRate = \frac{TP}{FN + TP} \tag{3.10}$$

False Positive Rate is the percentage of the number of records rejected incorrectly is divided by the total number of normal records.

$$FalsePositiveRate = \frac{FP}{FP + TN} \tag{3.11}$$

3.2 System Diagram

3.2.1 Use case diagram

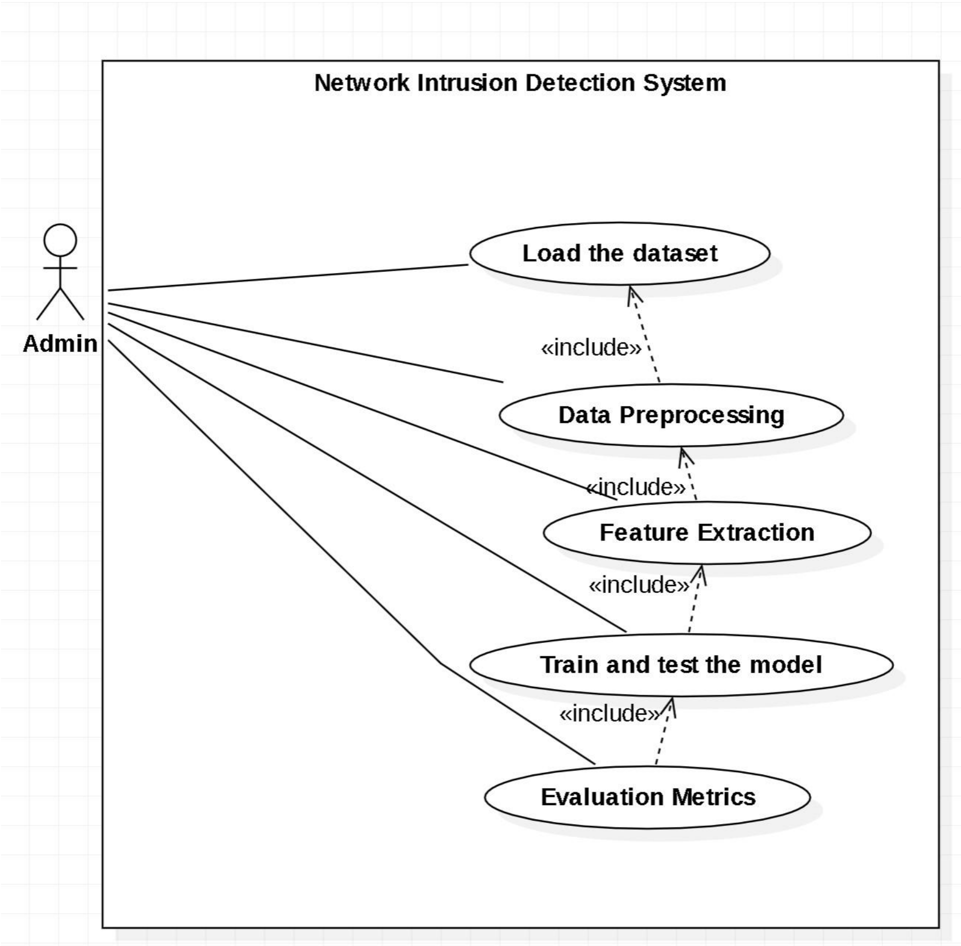

Figure 3.5: Use case Diagram of Network Intrusion Detection System

3.2.2 Software Development Model

Incremental model is a method of software engineering that combines the elements of waterfall model in iterative manner. It involves both development and maintenance. In this model requirements are broken down into multiple modules. Incremental development is done in steps from analysis design, implementation, testing/verification, maintenance. Each iteration passes through the requirements, design, coding and testing phases. The first increment is often a core product where the necessary requirements are addressed, and the extra features are added in the next increments. The core product is delivered to the client. Once the core product is analyzed by the client, there is plan development for the next increment.

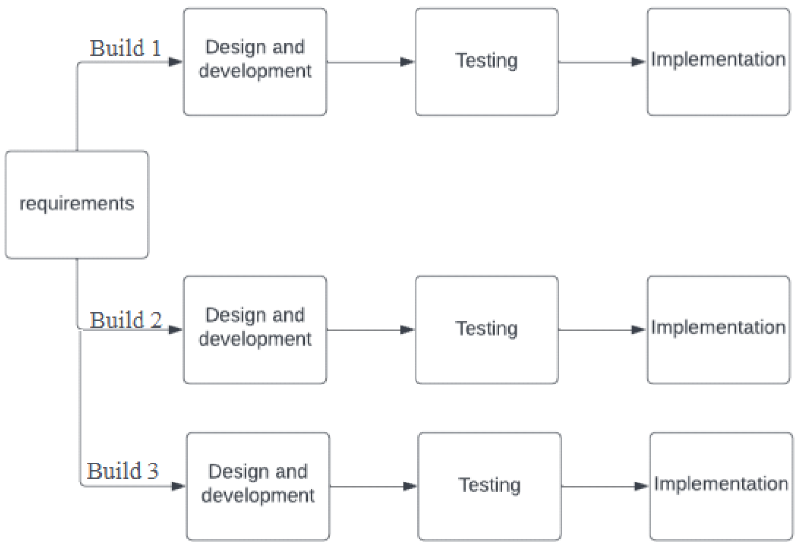

Figure 3.6: Incremental Model

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Result

We have completed the development of the project along with obtaining the desirable output. We divided the original dataset into train and test set with 80/20 split. We then trained the LSTM model for the train set and checked the classification report and confusion matrix for both the test set and the validation set that we created.

	precision	recall	f1-score	support
0	0.99	1.00	1.00	14706
1	1.00	0.98	0.99	13685
2	0.98	1.00	0.99	12351
micro avg	0.99	0.99	0.99	40742
macro avg	0.99	0.99	0.99	40742
weighted avg	0.99	0.99	0.99	40742
samples avg	0.99	0.99	0.99	40742

Figure 4.1: Classification Report for Test dataset

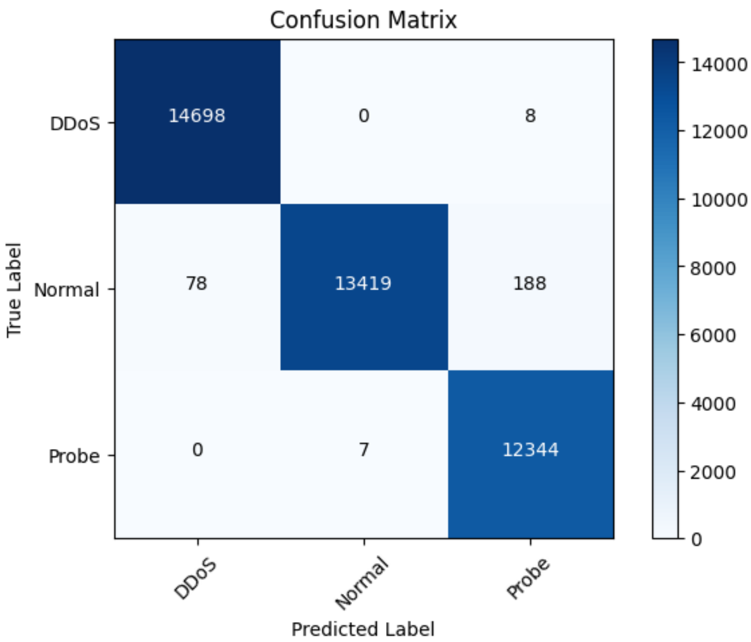

Figure 4.2: Confusion Matrix for Test dataset

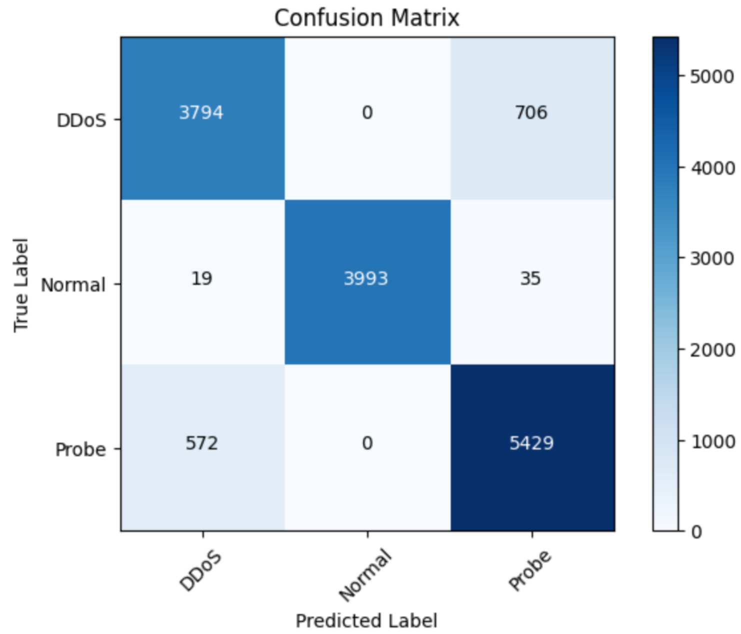

Figure 4.3: Confusion Matrix for Validation dataset

	precision	recall	f1-score	support
0	0.87	0.84	0.85	4500
1	1.00	0.99	0.99	4047
2	0.88	0.90	0.89	6001
accuracy			0.91	14548
macro avg	0.92	0.91	0.91	14548
weighted avg	0.91	0.91	0.91	14548

Figure 4.4: Classification Report for Validation dataset

In the above figures, 0 denotes DDoS class, 1 denotes Normal class and Probe class is denoted by 2. As we can see, the model achieved 91% accuracy for prediction of the labels in the validation set.

4.2 Discussion

bla bla bla

4.3 Limitations

1. hmmm
2. mmmm

CHAPTER 5

CONCLUSION AND FUTURE ENHANCEMENTS

5.1 Conclusion

blablablabla

5.2 Future Enhancements

1. bla bla
2. bla bla

REFERENCES

[1] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *Ieee Access*, vol. 5, pp. 21 954–21 961, 2017.

[2] S. Al-Emadi, A. Al-Mohannadi, and F. Al-Senaid, "Using deep learning techniques for network intrusion detection," in *2020 IEEE international conference on informatics, IoT, and enabling technologies (ICIoT)*. IEEE, 2020, pp. 171–176.

[3] C. Xu, J. Shen, X. Du, and F. Zhang, "An intrusion detection system using a deep neural network with gated recurrent units," *IEEE Access*, vol. 6, pp. 48 697–48 707, 2018.

[4] M. S. E. Sayed, N.-A. Le-Khac, M. A. Azer, and A. D. Jurcut, "A flow-based anomaly detection approach with feature selection method against ddos attacks in sdns," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 4, pp. 1862–1880, 2022.

[5] R. A. Elsayed, R. A. Hamada, M. I. Abdalla, and S. A. Elsaid, "Securing iot and sdn systems using deep-learning based automatic intrusion detection," *Ain Shams Engineering Journal*, vol. 14, no. 10, p. 102211, 2023.

[6] M. Elsayed, N.-A. Le-Khac, and A. Jurcut, "Insdn: A novel sdn intrusion dataset," *IEEE Access*, 09 2020.