***Dissertation on***

**“**Cloud based Network Intrusion Detection System using Deep Learning Algorithms**”**

*Submitted in partial fulfilment of the requirements for the award of degree of*

**Bachelor of Technology**

**in**

**Computer Science & Engineering**

**UE17CS490B – Capstone Project Phase - 2**

***Submitted by:***

|  |  |
| --- | --- |
| **Archana C**  **Chaitra H P**  **Khushi M**  **T P Nandini** | **PES1201701384**  **PES1201701370**  **PES1201701416**  **PES1201700064** |

*Under the guidance of*

|  |
| --- |
| **Dr. Sivaraman Eswaran**  Associate professor  PES University |
|  |

**January - May 2021**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

FACULTY OF ENGINEERING

**PES UNIVERSITY**

(Established under Karnataka Act No. 16 of 2013)

100ft Ring Road, Bengaluru – 560 085, Karnataka, India

**PES UNIVERSITY**

(Established under Karnataka Act No. 16 of 2013)

100ft Ring Road, Bengaluru – 560 085, Karnataka, India

**FACULTY OF ENGINEERING**

**CERTIFICATE**

*This is to certify that the dissertation entitled*

**‘Cloud based Network Intrusion Detection System using Deep Learning Algorithms’**

*is a bonafide work carried out by*

|  |  |
| --- | --- |
| **Archana C**  **Chaitra H P**  **Khushi M**  **T P Nandini** | **PES1201701384**  **PES1201701370**  **PES1201701416**  **PES1201700064** |

in partial fulfilment for the completion of eighth semester Capstone Project Phase - 2 (UE17CS490B) in the Program of Study - Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period Jan. 2021 – May. 2021. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 8th semester academic requirements in respect of project work.

|  |  |  |
| --- | --- | --- |
| Signature  Dr. Sivaraman Eswaran  Designation | Signature  Dr. Shylaja S S  Chairperson | Signature  Dr. B K Keshavan  Dean of Faculty |

**External Viva**

|  |  |
| --- | --- |
| **Name of the Examiners**  **1.** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Signature with Date**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_    **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
|  |  |

**DECLARATION**

We hereby declare that the Capstone Project Phase - 2 entitled **“Cloud based Network Intrusion Detection System using Deep Learning Algorithms”** has been carried out by us under the guidance of Dr. Sivaraman Eswaran, Associate professor and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester January – May 2021. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

|  |  |  |
| --- | --- | --- |
| **PES1201701384** | **Archana C** |  |
| **PES1201701370** | **Chaitra H P** |  |
| **PES1201701416** | **Khushi M** |  |
| **PES1201700064** | **T P Nandini** |  |

**ACKNOWLEDGEMENT**

I would like to express my gratitude to Dr. Sivaraman Eswaran, Department of Computer Science and Engineering, PES University and Prof. Prasad Honnavalli, Director, Center for Information Security, Forensics and Cyber Resilience for their continuous guidance, assistance, and encouragement throughout the development of this UE17CS490B - Capstone Project Phase – 2.

I am grateful to the project coordinators, Prof. Silviya Nancy and Prof. Sunitha R for organizing, managing, and helping with the entire process.

I take this opportunity to thank Dr. Shylaja S S, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department. I would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this project could not have been completed without the continual support and encouragement I have received from my family and friends.

**ABSTRACT**

The internet traffic and the number of attempts at malicious access to any organization have increased over the years. The attacker can take advantage of this and over flood the organization with dummy traffic and make the systems unresponsive. The model prefers DL algorithm over ML algorithm because DL learns new features which majorly define a model on its own without any human intervention, whereas ML requires the user to feed the important features to be considered by the model. The proposed system offers a scalable solution using DL algorithms to increase the responsiveness of the NIDS during high loads, hence increasing the reliability. The experimental results shows that DNN with four hidden layers achieved the best accuracy of 95.02% and least accuracy of 88.75% was achieved by a LSTM with two layers whereas highest achieved from ML algorithm is 86% using Random Forest.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Title** | **Page No.** |
|  | **INTRODUCTION** | **01** |
|  | **PROBLEM DEFINITION** | **04** |
|  | **LITERATURE SURVEY**  3.1) A Deep Learning Method With Filter Based Feature Engineering for Wireless Intrusion Detection System  3.2) A Novel Two-Stage Deep Learning Model for Efficient Network Intrusion Detection  3.3) A hybrid deep learning model for efficient intrusion detection in big data environment  3.4) Genetic convolutional neural network for intrusion detection systems  3.5) A Deep Learning Approach for intrusion Detection using Recurrent Neural Networks  3.6) Deep Learning Approach for Intelligent Intrusion Detection System  3.7) Deep Learning for Proactive Network Monitoring and Security Protection  3.8) A Novel Intrusion Detection Model for Detecting Known and Innovative Cyberattacks using Convolutional Neural Network  3.9) A Deep Neural Network for Network Intrusion Detection  3.10) Evaluation of machine learning algorithms for intrusion detection system | **05** |
| **4.**  **5.**  **6.**  **7.**  **8.**  **9.**  **10.**  **11.**  **12.** | **DATASET**  **PROJECT REQUIREMENTS SPECIFICATION**  **HIGH LEVEL DESIGN**  **LOW LEVEL DESIGN**  **SYSTEM DESIGN**  **PROPOSED METHODOLOGY**  **IMPLEMENTATION AND PSEUDOCODE**  **EXPERIMENTS RESULTS AND DISCUSSION**  **CONCLUSION AND FUTURE WORK REFERENCES/BIBLIOGRAPHY**  **APPENDIX A DEFINITIONS, ACRONYMS AND ABBREVIATIONS**  **APPENDIX B USER MANUAL (OPTIONAL)** | **22**  **24**  **25**  **26** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  | |  |
|  | |  |
|  | |  |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
| 3.1 | Feed Forward Neural Network Model | 6 |
| 3.2 | Autoencoder Model for Feature Extraction | 9 |
| 3.3 | Hybrid CNN-LSTM Model for feature extraction | 12 |
| 3.4 | Genetic Algorithm Model | 14 |
| 3.5 | RNN Model with Forward Propagation | 16 |
| 3.6 | Deep Neural Network using back propagation | 18 |
| 3.7 | Development and deployment of intelligent module for network supervising IDS functionality improvement. | 19 |
| 3.8 | Architecture of the CNN Model | 21 |
| 3.9 | LuNet Model | 23 |
| 4.1 | Screenshot of few rows of Dataset | 26 |
| 6.1 | System Design | 28 |
| 7.1 | Flowchart of model methodology | 29 |
| 8.0 | Correlation matrix of the features | 32 |
| 8.1 | LSTM Cell | 33 |
| 8.2 | GRU cell | 35 |
| 8.3 | Convolutional Neural Network | 37 |
| 8.4 | Structure of Deep Neural Network | 38 |
|  |  |  |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Table No.** | **Title** | **Page No.** |
| 4.1 | Description of features in UNSW-NB15 Dataset | 25 |
| 9.1 | Results using classical ML algorithms | 39 |
| 9.2 | Accuracy of different Deep Learning Algorithms | 40 |
| 9.3 | Accuracy of DNN with different hidden layers | 40 |
| 9.4 | Confusion Matrix for the DNN model | 41 |
|  |  |  |

**CHAPTER-1**

**INTRODUCTION**

Network Security has become an important aspect because computer networks are used extensively in all the fields. Various sensitive user data is handled by networks which are prone to various kinds of attacks. There are a variety of tools like firewall, antimalware, antivirus etc. used to detect and exploit attacks. These attacks are diverse and are evolving with sophisticated algorithms which makes it undetectable by traditional tools leading to data breach. These kinds of malicious attacks pose security challenges which makes it important to have a reliable and flexible IDS.

An Intrusion Detection System (IDS) is a software application or a device that monitors a network for any suspicious activity and an alert is issued to the management system on discovery of such abnormal events. IDS can be classified into the following categories:

1. Network Intrusion Detection system (NIDS) : This IDS monitors the network traffic flowing in and out of the entire system to detect traces of malicious activity.It is usually placed at the crucial points where all the traffic traverses.
2. Host Intrusion Detection System (HIDS) : This IDS monitors a particular host’s/system’s important OS files, application logs, system calls etc. for malicious activities.It can be installed on any individual devices such as desktop, servers.

IDS uses the below methods to identify the attacks :

1. Signature based : Intrusions can be detected by classifying access to the systems into predefined signature, patterns, malicious activities. Signatures need to be updated regularly and it is accurate in finding known types of attacks, but this won’t take care of the novel ways devised by attackers to breach the security of a system.
2. Anomaly based : It analyses the normal behaviour of the network and alerts when there’s a deviation.It is used to detect unknown attacks. Problem with this system is that it produces a high false positive rate as any deviation from normal baseline is considered as an attack.
3. Hybrid based : low positive rate and Accuracy can be achieved by using a combination of signature based and Anomaly based.

Anything that violates the CIA triad must be caught by the Intrusion Detection System. NIDS usually has a large false positive rate and classical ML classifiers have high computation rate with high FPR because they learn the characteristic of simple TCP/IP features locally whereas, Deep learning is a complex subnet of machine learning that

learns hierarchical feature representations and hidden sequential relationships by passing the TCP/IP information on several hidden layers making it reduce the false positive rate. Since the DL algorithms require extensive computational resources, the advent of GPUs and cloud‐based platforms can ease the way for the implementation of DL‐based methods.

To enhance the performance of IDS many studies are focusing on applying Machine Learning (ML) techniques both supervised and unsupervised. Some of the ML algorithms used are Naive bayes, Support Vector Machine (SVM), Random forest, Decision Trees, K-Nearest Neighbour (KNN) , Decision Trees etc. IDS built using ML have achieved a high accuracy with small amounts of input data. However if the large dataset is used it’s very time consuming and has high latency for large dataset.There are some issues with these classical algorithms. ML techniques fail in multi class classification because of more number of features. There are difficulties such as overfitting, inducing high bias due to redundant or irrelevant features. Most of the studies have used old publicly available dataset which doesn’t cover today's wide range of attacks. There is also the issue of scalability, where the model can take a long time to respond if it is bombarded with requests.

Deploying the IDS on cloud has a lot more advantages over traditional deployment.Cloud computing makes use of the internet to deliver their service. It offers different services like data storage, database, software etc. Infrastructure as a service(IaaS), Platform as a Service(PaaS) and Software as a Service(SaaS) are the types of Cloud computing. Availability, scalability, elasticity, speed, reliability are some of the advantages of cloud computing. Since the amount of traffic a network will receive is unpredictable allocating resources for IDS will play a key role. Allocating more resources than required will incur more cost and less resources results in delay of intrusion detection. Hence using services like cloud which can scale up and down as per the requirements will ensure the availability of IDS.

So the proposed idea is to use a deep learning based Network Intrusion Detection System which will be hosted on the cloud to classify malicious activities. The proposed method trains a model using DL algorithms that learns about the characteristics of a malicious activity using a pre-existing dataset and with each detection improves and optimizes the model. It will be hosted on the cloud and be configured as microservices which can be scaled depending on the network traffic.

The signature based IDS has evolved over the years into anomaly based IDS to detect anomalous behaviour posed by the attacker. However, these attacks are always evolving in

quality and both quantity. According to PurpleSec “In 2017 there were over 130 large-scale, targeted breaches in the U.S. per year, and that number is growing by 27% per year”. With increase in the internet traffic world wide , the influx of network traffic into any system has also seen a major spike over the years. The attacker can take advantage of this and over flood the system with dummy traffic and make the systems unresponsive. In such environments having a single running instance of an IDS to alert the organization about a potential attack in real-time can be quite challenging. The proposed system

offers a scalable solution using DL algorithms to increase the responsiveness of the IDS during high loads , hence increasing the reliability. The IDS can be scaled up or down

depending on the traffic. This way there won’t be concerns regarding investment on infrastructure and wastage of resources when the load is high and low respectively.

The model prefers DL algorithm over ML algorithm because DL learns new features which majorly define a model on its own without any human intervention, whereas ML requires the user to feed the important features to the model. Moreover, DL performs the optimization like deciding on the weights for each input on its own to improve the accuracy of the model. The proposed model has three instances on the cloud for database operations, classification and re-training of the model in frequent time-intervals respectively.

The proposed model uses the UNSW-NB15 dataset for NIDS.

**CHAPTER-2**

**PROBLEM DEFINITION**

The signature based IDS has evolved over the years into anomaly based IDS to detect anomalous

behaviour posed by the attacker. However, these attacks are always evolving in quality and both quantity. According to PurpleSec “In 2017 there were over 130 large-scale, targeted breaches in the U.S. per year, and that number is growing by 27% per year”.

With increase in the internet traffic world wide , the influx of network traffic into any system has also seen a major spike over the years. The attacker can take advantage of this and over flood the system with dummy traffic and make the systems unresponsive.

In such environments having a single running instance of an IDS to alert the organization about a potential attack in real-time can be quite challenging. The proposed system offers a scalable solution using DL algorithms to increase the responsiveness of the IDS during high loads , hence increasing the reliability. The IDS can be scaled up or down depending on the traffic. This way there won’t be concerns regarding investment on infrastructure and wastage of resources when the load is high and low respectively.

The model prefers DL algorithm over ML algorithm because DL learns new features which majorly define a model on its own without any human intervention, whereas ML requires the user to feed the important features to the model. Moreover, DL performs the optimization like deciding on the weights for each input on its own to improve the accuracy of the model.

The proposed model has three instances on the cloud for database operations, classification and re-training of the model in frequent time-intervals respectively.

**CHAPTER-3**

**LITERATURE SURVEY**

**3.1 Literature Survey 1**

**A Deep Learning Method With Filter Based Feature Engineering for Wireless Intrusion Detection System [1]**

**Abstract**

The authors have proposed a model by coupling filter based feature selection with Feed Forward Deep Neural Networks (FFDNNs) algorithm. NSL-KDD dataset is used in this study. They have compared their results with the following Machine Learning algorithms namely Support Vector Machine, Naive Bayes, K-Nearest Neighbor and Decision Tree. In Feature Extraction they have used a filter model which depends on the data's inherent nature rather than the classifier used .

**Methodology**

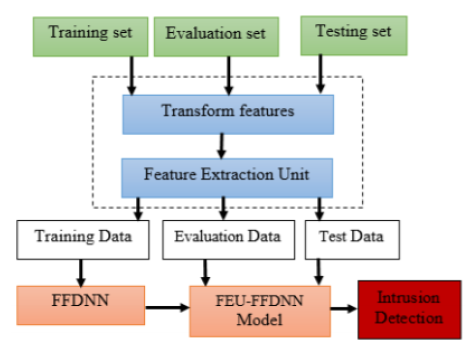


Figure 3.1: Feed Forward Neural Network Model

First step of this architecture consists of the separation of raw data. Split the dataset into evaluation and reduced training sets. They have used an Evaluation dataset for validating the training process. The test contains different data from the training and validation sets. Process of Two-way normalization and feature transformation on the data constitutes the second step of this model.

The last step of the architecture is training and testing of the models using FFDNNs and the Feature Extraction Unit-Feed Forward Deep Neural Network. Information Gain (IG) based algorithm is used to rank all the features which is a function of Feature Extraction Unit. Training is done in the following steps: Forward propagation using ReLu. Backpropagation of errors (stochastic descent) and updating the weights and biases using cost function.

**Evaluation of the Model**

Evaluation is done in two phases namely Before Feature Reduction and After Feature Reduction.

In the Before Feature Reduction phase, two kinds of classification are done namely Binary classification and the Multiclass classification. In Binary Classification, all features are used. We used a varying number of hidden layer nodes based on trial and error. 99.69% accuracy was obtained by training a model with 0.05 learning rate and 3 hidden layers consisting of 30 neurons on the KDD Evaluation set and KDDTest+ gave an accuracy of 86.76%. In Multiclass classification, all features are used. 86.62% accuracy was obtained by training the Feed Forward Deep Neural Network model with 0.05 learning rate and 3 hidden layers consisting of 60 neurons.

In the After Feature Reduction phase, 21 ranked features are used. In this phase also two kinds of classification are done namely Binary Classification and the Multiclass classification. In Binary Classification, varying numbers of hidden layer nodes are used based on trial and error. The best performing model gave an accuracy of 99.37% on train and 86.19 on test set by training a model with 0.05 learning rate and 3 hidden layers consisting of 30 neurons. In Multiclass classification, highest accuracy was achieved to be 99.54% on train and 86.19 on test set by training a model with 0.05 learning rate and 150 neurons.

**Conclusion**

For the Multiclass and Binary Classification problems, the Feed Forward Deep Neural Networks models with full and a Feature Extraction Unit-reduced feature space achieved superior performance compared to other ML classifiers .

**3.2 Literature Survey 2**

**A Novel Two-Stage Deep Learning Model for Efficient Network Intrusion Detection [2]**

**Abstract**

They propose a stacked Autoencoder (AE) based two staged DL model by using soft-max for classification . The initial stage is used for binary classification where it classifies a network traffic as attack or benign and gives a probability value. The probability value is fed as an input to the second stage along with the initial features to classify into different classes of attacks.This model is evaluated on two benchmark datasets: UNSW-NB15 dataset and KDD99 dataset.

**Methodology**

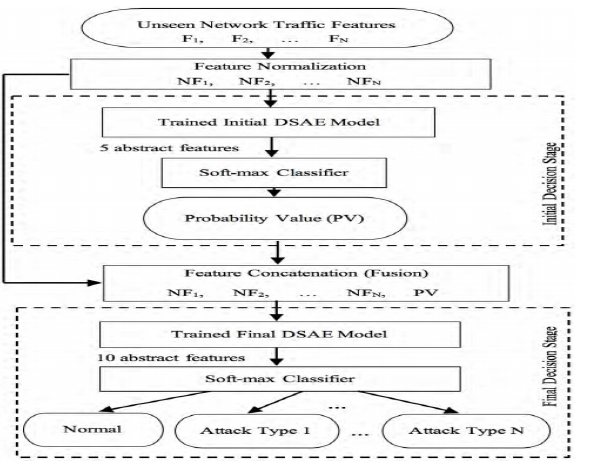
****

Figure 3.2: Autoencoder Model for Feature Extraction

The basic unit of this model is Autoencoder (AE) . An autoencoder is a type of ANN which uses unsupervised learning to learn efficient data codings. It contains one input layer,one or more hidden layers and one output layer. AE consists of two processes: i. Encoding process which tries to compress the input data using given weights. ii. Decoding process which tries to reconstruct back the input from the compressed data and reduces error by backpropagation.

The proposed model consists of two stages. Both the stages use Deep Neural Network because of its high performance and speed which makes it suitable for real time classification. It comprises a Deep Stacked Autoencoder (DSAE) with two hidden layers and a softmax layer on top for classification. They have used a semi supervised approach. In the first stage the first layer is fed with unlabeled features as an input which uses unsupervised learning for pre training, output of this layer is fed to the input of the second layer. These two layers are stacked together with a softmax layer whose output is taken to the second stage. The second stage uses supervised learning for fine tuning which uses back-propogation to tune parameters to reduce prediction error. Unseen networks features are classified by the model after completing the training phase. The first stage, which is binary classification, reduces overfitting problems and focuses more on abnormal instances in order to mitigate bias.

**Experiments and Evaluation**

The proposed model is evaluated on two publicly available datasets : new UNSW-NB15 dataset which comprises more complex types of attacks and the benchmark KDD99 dataset . Ten-fold cross validation is used for evaluation so that all data points are used both in training and testing.

Results for KDD99 dataset : In the initial stage the model was able to extract 10 abstract features out of 41 given input features and classified them into normal or abnormal based on the probability value of the softmax layer. It achieved 99% accuracy (145486 correctly classified out of 14558). In the final stage the 41 features along with the probability value from the initial stage were combined and 5 abstract features were extracted from it. The accuracy achieved in this stage was 99% ( 145580 correct out of 145586 instances).

Results for UNSW-NB15 dataset : 89.711% accuracy and 0.1018 False Alarm Rate (FAR) was achieved in the initial stage.212007 were classified correctly out of 236323. The multi-class classification model achieved 89.13 % accuracy with 0.7495 FAR.

The TSDL model’s average execution time for classifying one instance is 3.37 micro seconds.

**Conclusion**

Feature extraction plays an important role in improving the performance of the model.Rather than using classical linear transformation algorithms for feature extraction using DL techniques which are able to extract abstract features is a better option. As we can see from the results choosing a dataset which is new and contains a wide range of attacks plays an important role in training an IDS.

**3.3 Literature Survey 3**

**A hybrid deep learning model for efficient intrusion detection in big data environment [3]**

**Abstract**

This paper proposes a model which uses a hybrid of CNN and weight-dropped LSTM.

Hybrid deep learning model is used to efficiently detect network intrusions based on Convolutional Neural Network and a weight-dropped Long short term memory network. The deep Convolutional Neural Network is used to extract necessary and meaningful features from the Intrusion Detection System big data and weight dropped long short term memory to retain long-term dependencies among extracted features, this is done to prevent overfitting on recurrent connections.

**Methodology**

In this paper, they have used the deep convolutional neural network to extract the meaningful features from the network data traffic, exploiting its speed due to its weight sharing property. They have used the Long short term memory network to retain long-term dependencies among extracted features and to avoid the gradient vanishing problem. They have also used the drop-connect regularization technique on the hidden-to-hidden weight matrices within the Long short term memory to avoid the overfitting problem. Optimization is done based on the trial and error. Dataset used is UNSW-NB15 (has over 100GB of real time network traffic data from Jan-Feb 2015) and ISCX2012.

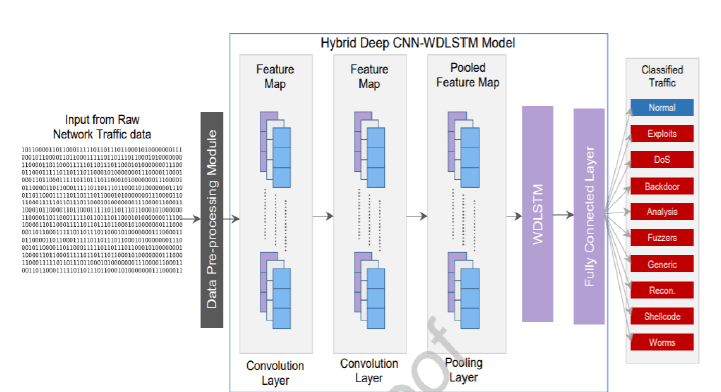
****

Figure 3.3: Hybrid CNN-LSTM Model for feature extraction

**Model Details**

The deep Convolutional Neural Network-Weight dropped long short term memory model has two one dimensional convolutional layers, one one dimensional maximum pooling layer, one one dimensional weight dropped long short term memory layer, and one fully connected layer.

They have used Rectified Linear Unit(ReLU) as the activation function for training the model in the two convolutional layers. To avoid overfitting, they have used dropout technique. The one dimensional weight dropped long short term memory network learns dependencies among the extracted features, ignoring some of the weights randomly to prevent overfitting.

The output of the one dimensional weight dropped long short term memory layer is next passed to the fully connected layer, which contains a softmax activation function to classify intrusions for detection.

**Results and Conclusion**

The accuracy of the model was calculated to be 96.97%. The average execution time of the model is very low, that is an average of 0.002383ms for one instance. This makes it more effective for real-time intrusion detection systems.This shows that this model can be employed on bigger datasets and be used in real time intrusion detection systems too.

**3.4 Literature Survey 4**

**Genetic convolutional neural network for intrusion detection systems[4]**

**Abstract**

This paper proposes a genetic algorithm based exhaustive search and fuzzy C-means clustering algorithm to select an improved feature subset which can be used for Network Intrusion Detection System. They propose three layers for constructing features using genetic algorithms, CNN and Fuzzy C-means to improve the final detection performance. Bagging classifier is used to select the CNN structure. The dataset used for training and evaluating their model is NSL-KDD.

This paper proposes a genetic algorithm based exhaustive search and fuzzy C-means clustering algorithm to select improved feature subset which is used for Network Intrusion Detection System

**Methodology**

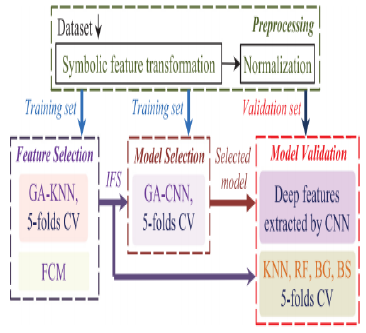


Figure 3.4: Genetic Algorithm Model

There are three phases in this methodology :

Feature selection phase - uses genetic algorithm combined with FCM and K-nearest neighbour. KNN is used as a fitness function to select the majority vote based on the distance between the test record and selected features with high frequency of occurrence. FCM is used for feature improvement by adding additional features.

Model selection phase - A CNN model is constructed using input layer, Convolutional layer (for feature selection), Relu activation function, Max Pooling layer (to reduce computational complexity), Fully Connected layer and an output layer. Genetic Algorithm is used to select the different number of connections between these layers.

Model Validation phase - Three CNN models are selected from the previous models and the model which gives highest accuracy is selected.

**Evaluation and Results**

NSL-KDD dataset used to evaluate the model in this study. Total of 33 features are selected from the model 28 from GA and 5 features from FCM.

Out of the three CNN models selected from the model selection phase 98.24% is the highest accuracy and 0.52 FPR is achieved. The proposed model has two parts: feature extraction and classification.

**Conclusion**

Feature selection is an important aspect for building an IDS for real time usage. This paper has employed a combination of different algorithms to extract best possible features.

**3.5 Literature Survey 5**

**A Deep Learning Approach for intrusion Detection using Recurrent Neural Networks[5]**

**Abstract**

This paper presents a model which is designed and implemented using RNN. Performance of the model is studied for both multiclass classification and binary classification and how the accuracy is impacted by different combinations of the learning rate and number of neurons. NSL-KDD dataset is used to compare the performance of multiclass classification using different ML algorithms like NB, SVM, RF, multi layered perceptron.

**Proposed Methodology**

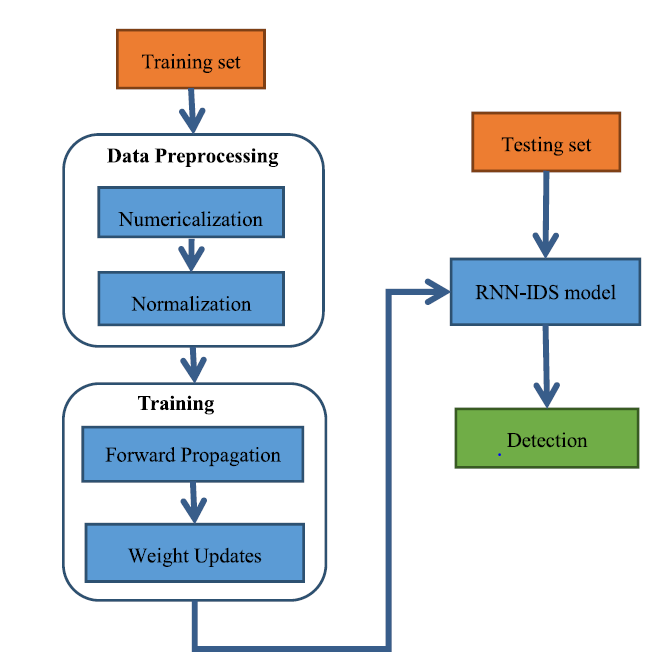
****

Figure 3.5: RNN Model with Forward Propagation

Widely used dataset for IDS experiments is the NSL-KDD dataset. The dataset is normalized and numericalized before it is used. There are two parts in this Recurrent Neural Network IDS implementation namely Forward propagation and Backward propagation. Calculating output values is the responsibility of Forward Propagation. Updating the weights by passing the accumulated residuals is the responsibility of Backward Propagation. Evaluation is done through metrics like Accuracy which defines the TP and TN over all the four values namely TP, TN, FP, FN.

**Experimental results**

Binary Classification achieved the highest accuracy with 0.1 learning rate and number of hidden nodes as 80. Multi-class classification achieved the highest accuracy with 0.5 learning rate and number of hidden nodes as 80.

**Conclusion**

When compared to classical methods proposed Recurrent Neural Network-intrusion detection systems have high accuracy for both multi-class and binary classification.

**3.6 Literature Survey 6**

**Deep Learning Approach for Intelligent Intrusion Detection System [6]**

**Abstract**

This paper presents a Deep Neural Network model with one input layer, five hidden layers and an output layered architecture on the KDD99 dataset as the hybrid intrusion detection system approach. They have used ReLU as activating function, stochastic gradient descent as optimization method and cross entropy as loss function and. The challenges faced in this paper are the dataset preprocessing, algorithm development, efficiency and scalability.

**Methodology**

The DNN is trained using backpropagation. All layers are fully connected and the number of neurons in the hidden layers is varied depending on the dataset. The last layer is the classifier layer. Binary Classification uses sigmoid as an activation function. Multi-class classifiers use sigmoid as it’s activation function . Batch normalization and regularization were used to avoid overfitting.

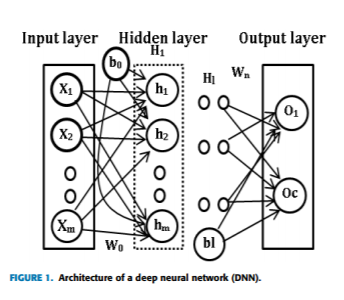
****

Figure 3.6: Deep Neural Network using Back Propagation

**Results**

For KDDCup 99 and NSL KDD datasets, most of the Deep Neural Network topologies showed training accuracy upto 93% and False positive rate was close to 0% in many cases.

**3.7 Literature Survey 7**

**Deep Learning for Proactive Network Monitoring and Security Protection [7]**

**Abstract**

Main Challenge faced in this work is the harmony between large scale data processing and DL techniques. Most of the current intrusion detection systems are capable of responding to network activities in near real time. But these are largely reactive solutions. However, there is a lack of a proactive solution. This paper proposes a proactive solution.The dataset used is an existing system ZEEK/Bro real data. Training data has less time index than testing data. Big data stack comprising Apache Spark, Arrow and Parquet is used to cope with the offline development phase of the model.

**Model**

This model has three components namely intrusion detection system module, data processing module and proactive forecasting module. This model uses ZEEK/Bro net IDS. Data processing module manages offline and online processing and feature engineering. Data is stored in order of timestamp. Feature engineering involves feature selection and extraction, data cleaning, and data transformation . Interaction with the data repository and Model building is controlled by this module. Proactive forecasting Module produces Deep Learning Model using Machine Learning methodologies in the development phase.

**Conclusion**

There is no particular winning model, LSTM ( & Gradient Recurrent Unit) derivatives are the best candidates in terms of quality and performance.

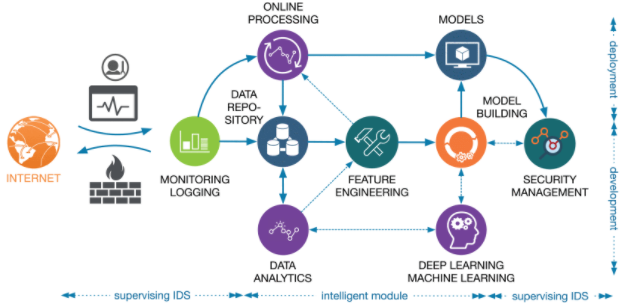
****

Figure 3.7:Development and deployment of intelligent module for network supervising IDS functionality improvement.

**3.8 Literature Survey 8**

**A Novel Intrusion Detection Model for Detecting Known and Innovative Cyberattacks using Convolutional Neural Network[8]**

**Abstract**

This paper introduces a CNN model to detect intrusion. The Canadian Institute for Cybersecurity Intrusion Detection System is used to train and test the model. They have tried two convolutional and two pooling layers in this research. Main challenges faced in this research are with respect to the dataset because of missing values and discrete dataset.

**Model**

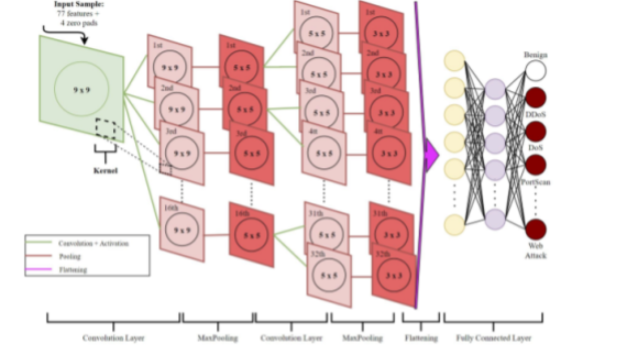
****

Figure 3.8: Architecture of the CNN Model

The model has three components namely convolutional layer, fully connected layer and pooling layer. Convolutional layer is used to create a feature map by extracting specific features from given input features. Eliminating noise and removing spatial dependency of the feature map is the task of pooling layer. This layer helps to avoid overfitting and reduces the dimensions by preserving useful information. Fully connected layer contains output neurons and helps in classifying the traffic as attack or benign. The proposed model consists of two layers of convolutional and pooling layer each. Replacing all the missing values by 0 and combining scattered files into a single concrete file is done as a pasrt of preprocessing.

**Results and Conclusion**

The model achieved overall accuracy of 98% with FPR as 1.02. The model is capable of detecting innovative attacks along with other known attacks. CNN helps in capturing complex characteristics of new cyberattacks.

**3.9 Literature Survey 9**

**A Deep Neural Network for Network Intrusion Detection[9]**

**Abstract**

In this paper, they have considered the existence of spatial and temporal features in the network traffic and they have proposed a model called LuNet which is the combination of CNN+RNN neural network. In this model. CNN and RNN learn input traffic data in sync with a gradually increasing granularity.

**Model**

In LuNet, they have mingled the CNN and RNN sub networks and synchronized both the CNN and RNN learning into multiple steps and each step can be performed by a combined CNN and RNN block called a LuNet block. CNN is placed before RNN at each level since CNN will extract high-level features from a large amount of data. At each level, both CNN and RNN can learn input to their full capacity without much interference.

**Conclusion**

Spatial features in the traffic data are learnt using CNN and temporal features but LSTM. Compared with other state-of-the-art techniques, LuNet can significantly improve the validation accuracy and reduce the False Positive Rate for Network Intrusion Detection Systems.

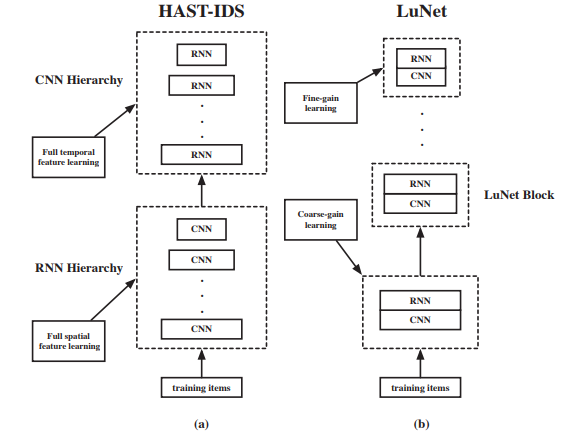
****

Figure 3.9: LuNet Model

**3.10 Literature Survey 10**

**Evaluation of machine learning algorithms for intrusion detection system[10]**

In order to enhance the detection rate of the intrusion detection system, [9] mainly focuses on false positive and false negative performance metrics. Dataset used in this paper is the KDD dataset. The implemented experiments demonstrate that the decision table classifier achieves the lowest value of false negative while the random forest classifier achieves the highest average accuracy rate.

**CHAPTER-4**

**DATASET**

**4. UNSW-NB 15 [8]**

The dataset is publicly available at:

<https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/>

The dataset used is UNSW-NB15[11] and it was collected at the Australian Center for Cyber Security(ACCS) in 2015 by the research group of cyber security. It is a network intrusion dataset. The raw network packets of the UNSW-NB15 data set are created by the IXIA PerfectStorm tool. The training set consists of 175,341 records. Testing set contains 82,232 records from the different types, attack and normal. UNSW-NB15 contains approximately 2.5 million records.



Using two different simulation periods of 15 hours and 16 hours, the dataset was generated. This dataset consists of both synthesized attack activities and real modern normal activities and of the network traffic. TCP dump is utilised to capture 100GB of the raw traffic. It contains nine families of attacks, namely, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worm.

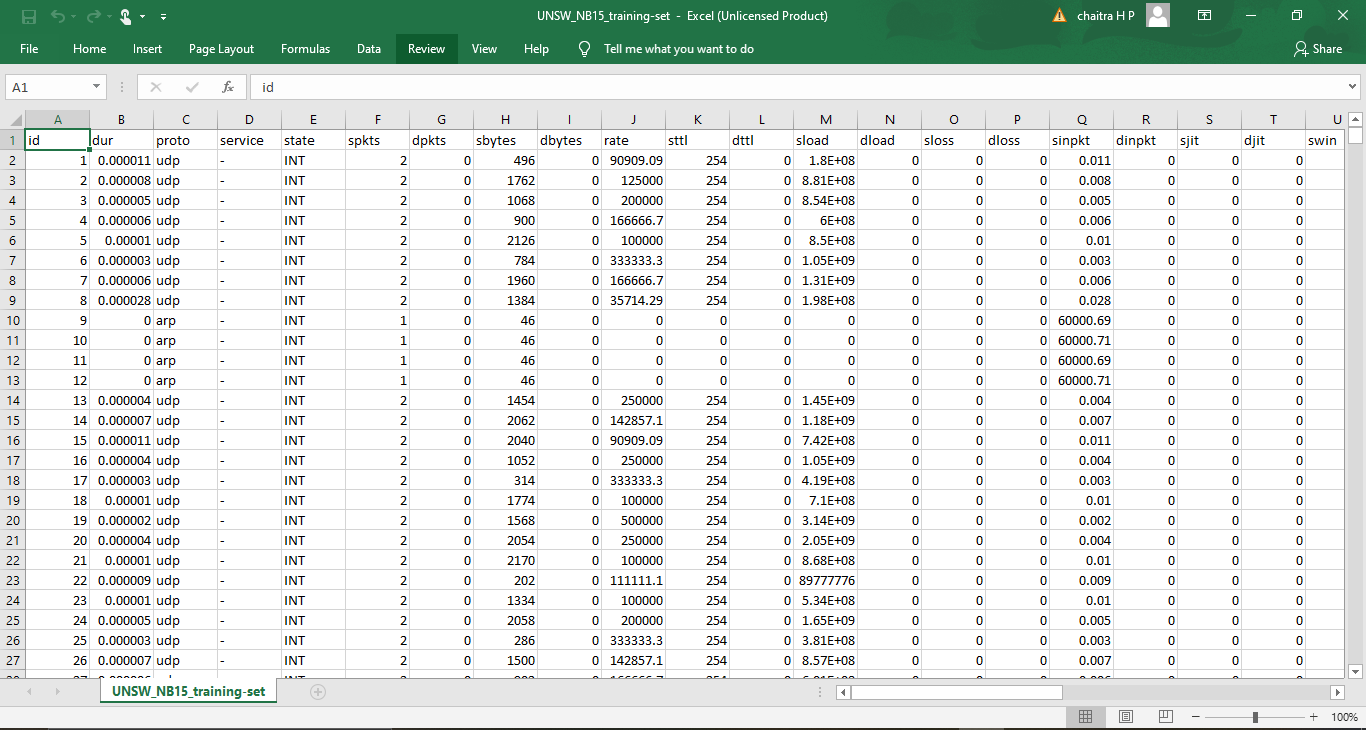


Figure 4.1: Screenshot of few rows of dataset

**CHAPTER-5**

**PROJECT REQUIREMENTS SPECIFICATIONS**

**5.1 External Interface Requirements**

**5.1.1 Hardware Requirements**

* GPUs for high computation.
* 4GB RAM (minimum)
* Intel i5 processor

**5.1.2 Software Requirements**

* Python
* Jupyter Notebook
* Cloud instances eg: Amazon EC2
* Tensorflow (1.15)
* keras (2.2.5)

# 5.2 Non-Functional Requirements

# 5.2.1 Performance Requirement

* For Industrial usage the model must classify the given packet into attack or benign
* Model must not produce a High False Positive Rate and should not classify an attack as a benign packet.
* Models must be trained periodically on up-to-date network traffic features.

# 5.2.2 Security Requirements

* the model must be cryptic so as to not be taken advantage of by users

**CHAPTER-6**

**SYSTEM DESIGN**

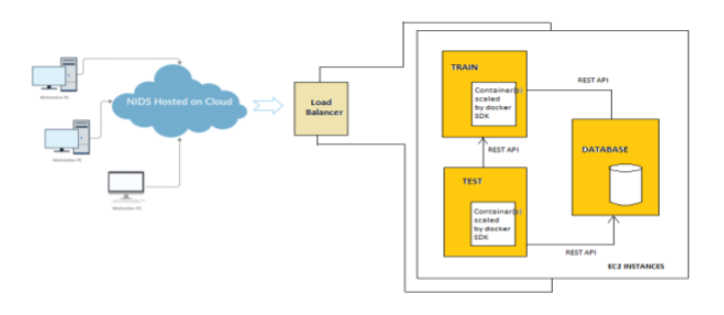


Figure 6.1 System design of the model on the cloud.

The proposed architecture is to build the IDS model as microservices on the cloud. The IDS will classify the input data instance as normal or different types of attacks based on the features of the initial stage. Using Neural Network models like Convolutional Neural network, Long short term memory or hybrid models with appropriate activation functions at each layer, the model will perform better than classic ML algorithms.

The model hosted on cloud will be such that, there will be 3 instances which will run the services for training, testing and storing the data as docker containers. There will be weekly updation of the database with the data that was tested to ensure the model works on updated real-time data.

During optimization, we are using hyperparameter tuning like learning rate, number and depth of hidden layers, number of epochs can be fixed after trial and experimentation.

**CHAPTER-7**

**PROPOSED METHODOLOGY**

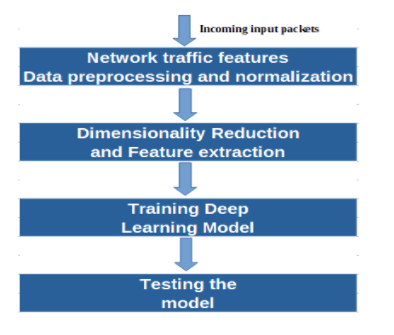
****

Figure 7.1 Flowchart of the Methodology

The model will get the incoming traffic features from the training dataset. The traffic is fed to the neural network model which extracts features and trains the model to classify traffic as attack or benign. The testing dataset records are used to verify the performance of the model such as accuracy, precision etc. A two stage model is used in this study: Initial stage is to extract minimal meaningful features which best represent the data. Next stage is to classify the input data instance as normal or different types of attacks based on the features of the initial stage. Neural Networks like Deep Neural Networks and some other models are used with appropriate activation functions at each layer. During optimization, hyperparameter tuning like learning rate, number and depth of hidden layers are used, number of epochs can be fixed after trial and experimentation. In an optimization algorithm, Learning Rate is a tuning parameter which determines the step size that needs to be taken at every iteration when we are moving towards a minimum of a loss function.

**CHAPTER-8**

**IMPLEMENTATION AND PSEUDOCODE**

For this phase, We have tried to implement classical Machine Learning (ML) models like Naive Bayes, K Nearest Neighbor (KNN), Random Forest, Decision Trees and Adaboost. We have used built in models from sklearn library. We started by preprocessing the dataset which will be used for model implementation. All the categorical data is converted to numerical using the sklearn label\_encoder function. To remove all the correlated columns pandas correlation function with pearson correlation coefficient is used. We were able to reduce 42 features to 29 which can highly improve the performance of the model.

First we have used the basic Machine Learning classifier - Logistic Regression approach - where we can’t solve nonlinear problems because Logistic Regression is defined for linear decision surfaces but we can’t find linear data in real-life scenarios.Next we have used the basic Machine Learning classifier - Naive Bayes. The basic assumption of this algorithm is that the features are independent which may not be true in many cases and reduces the performance. We can consider Naive Bayes as an acyclic graph between expected output and the number of features. After that we have used the K - Nearest Neighbor. By using KNN, we might get a low prediction stage due to the large dataset. And also the KNN requires more memory because it needs to save all the training data.

Then we have used the Decision Tree. The major limitation of the Decision tree is that if we change a small amount of data, then it will lead to a large change in the structure of the decision tree. Extending the Decision Trees, we have used the Random Forest which will help in reducing overfitting in the decision trees which in turn helps in increasing the accuracy. Random Forest is suitable for both categorical and continuous values. And also the trees which are inside the forest should be figured because every individual tree inside a forest predicts the expected output and then we will use voting technique to select the expected output which contains largest votes in number.

This phase we have implemented DL models like LSTM, GRU, CNN and DNN. These models are hosted as microservices on AWS cloud.

Microservice is a service-based application development. Here, big applications will be divided into smaller independent service units. It is the process of implementing service oriented architecture by dividing the entire application as a collection of interconnected services where each service will serve only one business need.

We are using Amazon AWS cloud services for our implementation. Amazon offers IT services in the form of web services which is also known as cloud computing. The model hosted on cloud will be such that, there will be three instances which will run the services for training, testing and storing the data as docker containers. We are using Docker Containers, REST APIs over postman and load balancer.

For testing these REST APIs, we are using postman.

It scales the load balancer as the incoming traffic changes over time. It increases the availability of your application. We can add one or more listeners to the load balancer.

Batch normalization is used to improve the accuracy. By normalizing the input of each layer in the network, not only on the input layer, Batch Normalization significantly reduces the training time. It is a technique which is used for very deep neural networks training which standardizes the inputs to a layer for each of the mini-batch. This will have the effect of stabilizing the learning process. This dramatically reduces the number of training epochs which are required to train deep networks. Batch Normalization is usually added after the activation function of the output layer or before the activation function of the input layer.

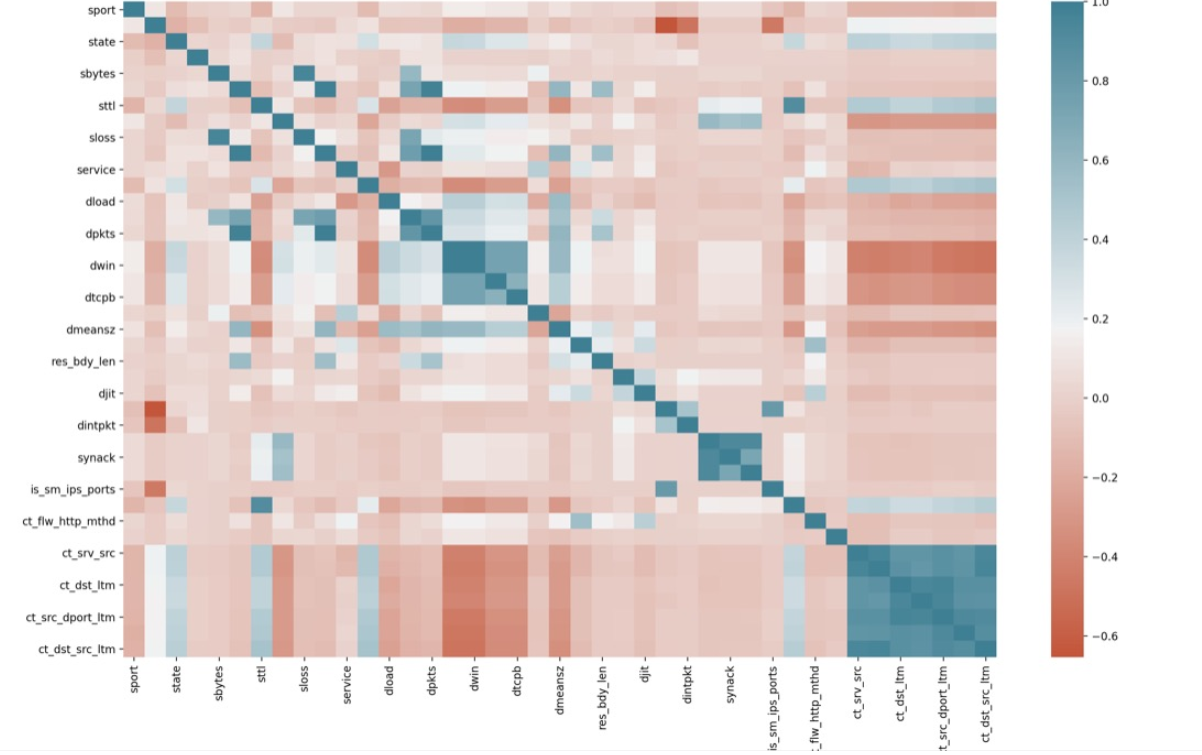


Figure 8.0 Correlation matrix of the features

**8.1 Long Short Term Memory**

Long Short-Term Memory(LSTM) is an improvement on RNN. RNN suffers from vanishing gradient problems so the layers won’t learn much and will forget the information it has learned. When we have a long sequence to process and predict the output RNN may forget some of the starting important information hence suffers from short memory.

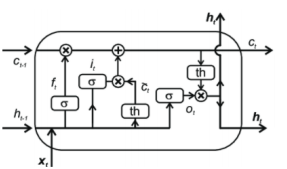


Figure 8.1 LSTM cell

Gates are the internal mechanism of LSTM which helps in regulating the flow of information. The important data in the sequence is decided by gates and it stores that information for further processing. LSTM keeps the relevant information which helps them in predictions and non relevant data is thrown away. LSTM uses three gates called Forget gate, Input gate and Output gate. These three gates help in deciding what information to add, output, and forget from memory. LSTM keeps information in the memory with these gates only as long as required. In cell state, all the information will be stored. the hidden state is used for computing the output i.e., for making predictions.Forget state helps in deciding information which needs to be removed from the cell state or memory.

The following set of equations represents the computations of forget gate, write gate and output gate.

W(f), u(f) = input to f(t)

W(i), u(i) = input to i(t)

W(g), u(g) = input to g(t)

W(o), V(o) = input to O(t)

1. Forget gate:

f(t) = σ (W(f) h(t − 1) + U(f) x(t))

1. Write gate:

i(t) = σ (W(i) h(t − 1) + U(i) x(t))

g(t) = tanh (W(g) h(t − 1) + U(g) x(t))

1. Output gate:

O(t) = σ (W(o) h(t − 1) + V(o) x(t))

1. h(t) = O(t) ⊙ tanh(c(t))
2. c(t) = c(t − 1) ⊙ f(t) + i(t) ⊙ g(t)

**8.2 Gated Recurrent Unit**

Gated Recurrent Unit(GRU) is also an improvised version of Standard Recurrent Neural Networks(RNN). GRU solves the vanishing gradient problem which is present in the RNN by using two gates namely Update gate and Reset gate. It is basically a combination of an input gate and a forget gate, which is present in LSTM. Update gate can be used in determining how much of the previous knowledge needed to be passed along into the future. It is similar to the output gate in an LSTM. Reset gate can be used in determining how much of the previous knowledge to forget. It is similar to the combination of input and forget gate in an LSTM. Sigmoid function regulates the update gate. Unlike LSTM, GRU will not use cell state, instead it will use the hidden state to transfer information. Training wise GRU is faster especially when compared to LSTM. GRU saves a lot of memory since it will not use cell state.

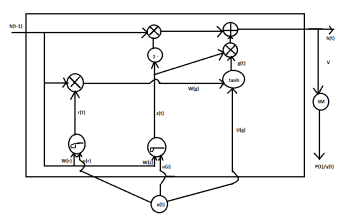


Figure 8.2 GRU cell

The following set of equations represents the computations for Reset gate and Update gate.

1. Reset gate:

r(t) = σ(W(r) h(t − 1) + U(r) x(t))

1. Update gate:

Z(t) = σ(W(z) h(t − 1) + U(z) x(t))

1. g(t) = tanh(u(g) x(t) + w(g)(r(t) ⊙ h(t − 1)))
2. h(t) = (h(t − 1) ⊙ (1 − z(t))) + g(t) ⊙ z(t)

Gated Recurrent Units are a slight variation of Long short term memory network. It has one less gate and is wired slightly differently than Long Short Term Memory.

Update gate determines what amount of the information should be kept from the last state and what amount of the information should be let in from the previous layer. Reset gate functionality is similar to that of the forget gate of Long Short Term Memory network.

Gated Recurrent Units are less complex than Long Short Term Memory because it has less number of gates. If the dataset is small then Gated Recurrent Units is preferred

otherwise Long Short Term Memory for the larger dataset. Gated Recurrent Unit exposes the complete memory and hidden layers but Long Short Term Memory does not.

**8.3 Convolutional Neural Network**

The complete network represents a single differentiable score function. They have a loss function either SVM or Softmax on the fully-connected layer. Unlike an Artificial Neural Network, the convolutional neural networks layers have neurons arranged in 3 dimensions namely width, height, depth.

Each layer transforms the 3D input volume to a 3D output volume with some differentiable function that may or may not have parameters. All these three layers stack together to form a Convolutional Neural Network.

ReLU as an activation layer will apply an elementwise activation function. Pool layers can be applied to perform a downsampling operation along the spatial dimensions. Fully

connected layers will compute the class scores. The idea behind Convolutional Neural Network is a moving filter which passes through the image. This moving filter applies to a certain neighbourhood of nodes.

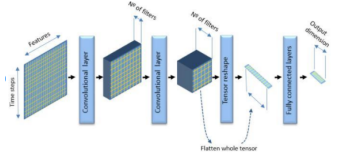


Figure 8.3 Convolutional Neural Network

**8.4 Deep Neural Network**

The aim of Deep Neural Networks is to receive a set of inputs, perform progressively complex calculations on them, and give output to solve real world problems like classification. Deep Neural Networks use the gradient descent method for optimizing the network and minimising the loss function.

Every node in the hidden and output layers has its own classifiers. Inputs are taken by the input layer and passes it onto its scores to the upcoming hidden layer for further activation and this goes on till the output is reached. This kind of progress from left to right from input to output in the forward direction is called forward propagation. Every node in the Deep Neural Network is a perception that mimics a neuron in a biological neural network. The prediction of accuracy depends on its weights and biases.

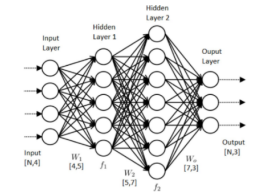


Figure 8.4 Structure Of DeepNeural Network

**CHAPTER-9**

**RESULTS AND DISCUSSION**

****

Table 9.1 shows the accuracy of different Machine Learning Algorithms. Among the different Machine Learning models implemented, Random Forest and Decision Tree achieved the highest accuracy of 86%.

Table 9.2 shows the accuracy of different Deep Learning Algorithms. From Table 9.2, it seems that adding more number of hidden layers to LSTM and GRU did not improve the accuracy whereas that was not the case with DNN. The LSTM and GRU achieved around 88.78%, 88.79% accuracy respectively. Deep Neural Network is implemented with two to four hidden layers among which four hidden layers gave the highest accuracy of 95.02% with precision 85.92%. Adding more number of hidden layers improved the accuracy of DNN.

Table 9.2: Accuracy of different Deep Learning Algorithms

|  |  |
| --- | --- |
| **DEEP LEARNING ALGORITHMS** | |
|  | **Accuracy** |
| **LSTM with 1 hidden layer** | **88.81%** |
| **LSTM with 2 hidden layers** | **88.75%** |
| **LSTM with 3 hidden layers** | **88.78%** |
| **GRU with 1 hidden layer** | **88.79%** |
| **GRU with 2 hidden layers** | **88.79%** |
| **GRU with 3 hidden layers** | **88.76%** |
| **CNN with 2 convolutional layers** | **93.11%** |

Table 9.3: Accuracy of DNN with different hidden layers

|  |  |  |
| --- | --- | --- |
| **DNN WITH DIFFERENT HIDDEN LAYERS** | | |
|  | **Accuracy** | **PRECISION** |
| **1 hidden layer** | **93.71%** | **84.96%** |
| **2 hidden layers** | **93.81%** | **72.90%** |
| **3 hidden layers** | **94.43%** | **85.32%** |
| **4 hidden layers** | **95.02%** | **85.92%** |

Table 9.4: Confusion Matrix for the DNN model

|  |  |  |
| --- | --- | --- |
|  | **Predicted benign** | **Predicted attack** |
| **Actual benign** | 196299 (TN) | 3511 (FP) |
| **Actual attack** | 7355 (FN) | 21440 (TP) |

Table 4 represents the confusion matrix for the Deep Neural Network model. True positive tells that the traffic is predicted as an attack correctly by the model. True negative tells that the traffic is predicted as a benign correctly by the model. False positives tell that the model predicted an attack but they are benign. False negatives tell that the model predicted as benign but actually they are the attacks.

**CHAPTER-10**

**CONCLUSION AND FUTURE WORK**

Instances of an AWS are used to run the model on the cloud and configure the docker containers accordingly. We have implemented different Deep Learning Models like Deep Neural Networks, Long Short Term Memory and Gated Recurrent Unit. The Performance Metrics are calculated for the proposed model and comparing it with the other studies. The Deep Neural Network model achieved good accuracy. Using batch normalization has significantly improved the accuracy of the model. This model can offer real time detection provided, the input to the model for testing is given according to the format specified and follows necessary.

Our study shows that the Deep Learning Models have the ability to outperform Machine Learning models when it comes to implementation of an intrusion detection system. Combination of different DL models could be used to further improve the accuracy of the model. A proactive model which can detect future attacks may be implemented in future for prior detection of attacks.

**REFERENCES/BIBLIOGRAPHY**

[1] Sydney Mambwe Kasongo And Yanxia Sun, “A deep learning method with filter based feature extraction for wireless intrusion detection system”,IEEE, 2020.

[2] Farrukh Aslam Khan (Senior Member, IEEE), Abdu Gumaei , Abdelouahid Derhab, and Amir Hussain , “A Novel Two-Stage Deep Learning Model for Efficient Network Intrusion Detection”, IEEE 2019.

[3] Mohammad Mehedi Hassan, Abdu Gumaei, Ahmed Alsanad, Majed Alrubaian, Giancarlo Fortino, “A hybrid deep learning model for efficient network intrusion detection in big data environment “,ScienceJournal 2020.

[4] Minh Tuan Nguyen, Kiseon Kim, “Genetic convolutional neural network for intrusion detection systems “ ,2020

[5] Chuanlong Yin, Yuefei Zhu, Jinlong Fei, Xinzheng He “A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks”, IEEE 2017

[6] R.Vinayakumar, Mamoun Alazab, K. P. Soman , Prabaharan Poornachandran, Ameer Al-nemrat, And Sitalakshmi Venkatraman,”Deep Learning Approach for Intelligent Intrusion Detection System”, IEEE 2019

[7] Giang Nguyen, Stefan Dlugolinsky, Viet Tran And Alvaro Lopez Garcia, ” Deep Learning for Proactive Network Monitoring & Security Protection” , IEEE 2020.

[8] Samson Ho, Saleh Al Jufout, Khalil Dajani, Mohammad Mozumdar, “A Novel Intrusion Detection Model for Detecting Known and Innovative Cyberattacks Using Convolutional Neural Network”, IEEE 2021

[9] Peilun Wu, Hui Guo, “LuNet: A Deep Neural Network for Network Intrusion Detection”, IEEE 2019

[10] Mohammad Almseidin, Maen Alzubi, Szilveszter Kovacs, Mouhammd Alkasassbeh, “Evaluation of Machine Learning Algorithms for Intrusion Detection System”