Proceedings of 10th IOE Graduate Conference
Peer Reviewed
ISSN: 2350-8914 (Online), 2350-8906 (Print)
Year: 2021 Month: October Volume: 10

Anomaly Detection in Computer Networks using Multilayer Perceptron

Bishal Heuju [a], Daya Sagar Baral [b]

[a, b] *Department of Electronics and Computer Engineering, Pulchowk Campus, IOE, Tribhuvan University, Nepal*
Corresponding Email: [a] b.heuju@gmail.com, [b] dsbaral@pcampus.edu.np

Abstract
The tremendous growth of internet and computer networks is making it easier for people to stay connected with each other and share resources at their ease. With the increasing networks, there has been continuous growth in malicious network attacks in intent to steal and exploit the personal information of an individual, business or an organization. In this work, a method for the classification of network anomalies is presented using Multilayer Perceptron on NSL-KDD and UNSW-NB15 dataset to train the classification model. The NSL-KDD and UNSW-NB15 datasets are preprocessed and then fitted using MLPClassifier. Grid Search is used for parameter optimization evaluated using different analysis metrics. The model is trained for both multiclass and binary class classification which can classify the 5 classes for KDD dataset and 10 classes for UNSW-NB15 dataset. The experimental results show that Grid Search found the optimum parameters for the neural network to be 2 hidden layers with 100 neurons in each layer and learning rate of 0.001; and performs fairly well for both multiclass and binary classification.

Keywords
anomaly, networks, NN, KDD-99, UNSW-NB15

1. Introduction

A random unexpected behavior in a system is an anomalous behavior, simply known as anomaly. Typical hardware and/or software failures can also cause anomalous behavior. Existence of such undetected presence of anomaly in a computer network manifests most of the security breaches. Detecting and mitigating these anomalies in computer network is important. Analyzing these kinds of behavior manually is tedious, boring and prone to human errors. For its simplification, it can be automated using machine learning algorithms. Machine learning based anomaly detection in computer networks is gaining popularity in the last decade. These kinds of anomaly detection identify patterns in data to classify the intrusion as anomaly.

2. Literature Review

A significant amount of research has been done for anomaly detection. Many of these researches are based on machine learning techniques with KDD99 dataset while some are statistical analysis to detect intrusion in the networks.

A survey of distance and similarity measures used in network intrusion anomaly detection was done in a research [1]. It presented an overview of the use of distance and similarity measures. Researches published with exemplary use of distance measures in the field were presented in the research. It also presented the areas that required further focus to improve the performance of the distance measure for anomaly detection. Distance based measures of classification and clustering were reviewed and analyzed. Clustering with k-NN, supervised classification were some of the surveyed algorithms in the research.

Another research, in 2019, proposed decision tree and Support Vector Machine algorithms to detect attack signature on a specialized dataset [2]. The dataset contained regular profiles and several DoS attacks in wireless sensor networks, constructed to classify four types of DoS attacks: Blackhole, Grayhole, Flooding and Scheduling. Their result showed that the performance of decision tree was better than of SVM.

The performance of various intrusion detection techniques was done by a research in 2018 [3]. To resolve the concerns of performance, a comparative study using Support Vector Machine, Random Forest, and Extreme Learning algorithms was done. The NSL-KDD dataset was used in the research work. The results from the research showed that the extreme learning algorithm is better in classifying the intrusions in terms of precision, recall and accuracy.

A research paper, in 2017, presented the review on different techniques and intrusion classification on KDD99 dataset [4]. A new and effective technique could be implemented by classifying the different network issues, which can categorize and identify intrusions in the KDD99 dataset. They concluded that a single technique is not able to provide accurate anomaly detection rate. They suggested to use an efficient automatic hybrid technique is to achieve accurate detection rate which could also reduce the false prediction rate as well as decrease the time complexity.

3. Methodology

3.1 System Model

The system model proposed is shown in Figure 1. The dataset is a collection of raw data. Important relevant features are extracted from this dataset and preprocessed to prune and remove any redundancy from the data. The obtained dataset is training dataset which is the input to the machine learning algorithm.

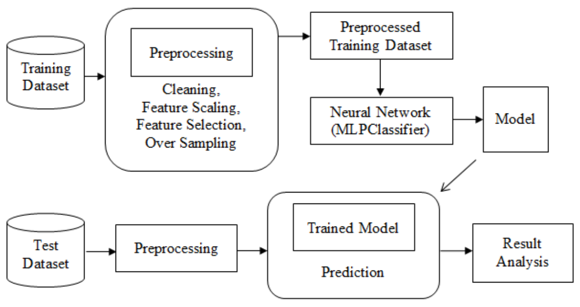
Training Dataset
Preprocessing
Cleaning, Feature Scaling, Feature Selection, Over Sampling
Preprocessed Training Dataset
Neural Network (MLPClassifier)
Model
Test Dataset
Preprocessing
Trained Model
Prediction
Result Analysis

Figure 1: System Model Diagram

In case of Neural Networks, the training data set is fed as input to the neurons. The neurons activate or do not activate according the input. The connections between the neurons change as they are trained. The final trained network gives us the model. This trained network is evaluated on data it has not been directly trained on. Then the network or model is fiddled with to improve the performance of the model. The trained model then is able to detect anomaly with certain confidence.

3.2 Dataset Description

3.2.1 KDD CUP 99 Dataset

The KDD CUP 99 dataset is a modification of the DARPA 98 dataset to detect anomaly in network traffic. The dataset was subsequently filtered for use in the International Knowledge Discovery and Data Mining Tools Competition, which resulted to the KDD CUP 99 dataset [5]. The KDD99 has following five classes of patterns:

- Normal: Non-attack data
- DoS (Denial of Service) Attacks: Prevent users from receiving services.
- U2R (User to Root) Attacks: Unauthorized root user operations by a user.
- R2L (Remote to Local) Attacks: Unauthorized access from a remote to local network.
- Probe Attacks: Attacks performed to find information about the server.

Each of these intrusion classes are divided further into subclasses based on the specific procedure used for the attack. Each of the pattern has 41 features. Each of these features are allocated to one of the three categories: basic features, content features and traffic features.

3.2.2 NSL-KDD CUP 99 Dataset

The KDD99 dataset is found to have lots of redundancy and irregular data. To overcome the drawbacks of KDD99 dataset, NSL-KDD dataset was created in 2009 with an effort by Tavallaee et al. [6]. The NSL-KDD dataset had following aspects of improvement over the original KDD99 dataset:

- Unnecessary records from the training data were eliminated
- Redundancy was eliminated
- Number of records in training and test sets were proportionally distributed
- Data had more homogeneous distribution

Although the original KDD99 dataset is 20 years old, it is widely used in academic research purposes. The NSL-KDD dataset has improved the original KDD dataset and used for research in machine learning and anomaly detection in computer networks.

Table 1: NSL-KDD Class Distribution

Class	Training Set	Test Set
Normal	67,343	9,711
DoS	45,927	7,456
Probe	11,656	2,421
R2L	995	2,756
U2R	52	200
Total	125,952	22,544

The NSL-KDD dataset is more concise and efficient than the standard KDD99 dataset. The number of records in the NSL-KDD dataset for train sets (KDDTrain+.txt) and for test sets (KDDTest+.txt) are reasonably small and the whole set can be used without the need to select only a portion of the datasets. The Table 1 shows the distribution of different attack types in the NSL-KDD train and test dataset.

3.2.3 UNSW-NB15 Dataset

The UNSW-NB15 dataset [7] was published in 2015 which is a newer dataset in the field of network intrusion. Numerous studies have shown that the KDD99 and NSLKDD benchmark dataset, generated almost two decades ago, do not include the modern network traffic and low footprint attacks. The UNSW-NB15 dataset was created to tackle this problem and create a hybrid real modern normal and synthesized attack activities that occur in the modern network traffic. Novel and existing methods were used for the generation of the features of the UNSW-NB15 dataset.

The use of the UNSW-NB15 dataset allows us to explore a newer dataset of 2015, which consists of modern network attacks. Similar as in the case with NSL-KDD, it is preprocessed and implemented with neural network and validated.

The UNSW-NB15 dataset represents 9 classes of attacks categories. The dataset consists of 49 features and a variety of normal and attack distribution activities. The dataset is divided into training and test set. The training set consists of 175,341 records and the test set consists of 82,332 records including different types of attacks and normal class.

The class distribution present in UNSW-NB15 dataset is tabulated in the Table 2. It can be seen that there is class imbalance in the dataset. The normal class has highest number of records while the worms class has the lowest number of records.

Table 2: UNSW-NB15 Class Distribution

Class	Training Set	Test Set
Normal	56,000	37,000
Analysis	2,000	677
Backdoor	1,746	583
DoS	12,264	4,089
Exploits	33,393	11,132
Fuzzers	18,184	6,062
Generic	40,000	18,871
Reconnaissance	10,491	3,496
Shellcode	1,133	378
Worms	130	44
Total	175,341	82,332

3.3 Preprocessing and Feature Selection

Classification of the anomalies is a predictive modeling problem. There are many reasons why raw data typically cannot be used directly for model training. The learning algorithm needs the data to be numeric values. Some algorithms need specific requirement of the data. The data needs to be noise and error free and should be corrected if any such noise and errors are present. Complex and irrelevant data should be removed as they may do more harm in the learning process. Therefore, the raw data should be preprocessed before using the data to train the model and evaluate the model trained.

The KDD dataset has imbalance in the attack classes distribution due to redundancy and because some attacks are less probable than others. The NSL-KDD dataset has already removed the redundancies from the original dataset. To decrease the imbalance in the data for proper classification and to make the dataset reliable, a seried of common preprocessing steps are carried out for both KDD and UNSW-NB15 dataset.

3.3.1 Data Cleaning

The NSL-KDD dataset has already been cleaned from the standard KDD99 dataset and the missing values and redundancies present within the data is taken into account and removed as necessary. Thus, this step is not necessary for NSL-KDD dataset. The UNSW-NB15 dataset has been processed to remove the errors, fill missing values and remove redundancies in the data.

3.3.2 Feature Scaling

Feature scaling is a technique used in machine learning for preprocessing the data to standardize the independent features present in the data to a fixed range to prevent algorithms to be biased by the features with greater values. For neural network which uses gradient descent as optimization technique are sensitive to feature scaling. By keeping the features in a similar scale, it helps the gradient descent to converge more quickly to the local minima. Standard Scaler using the $StandardScalar()$ provided by scikit-learn library is used in this work which scales each feature such that the new distribution is centered around 0 and having standard deviation 1. StandardScaling is done by calculating the mean and standard deviation and then the feature is scaled as,

$$x_i' = \frac{x_i - mean(x)}{stdev(x)}$$

Both the NSL-KDD dataset and UNSW-NB15 dataset were standardized using the standard scaler to scale the feature values.

3.3.3 Categories Transformation

One-hot-encoding is used here, which is a popular technique used for encoding the categorical values into numeric values. The categorical data is removed and a new binary variable is added for each unique category with 1 indicating the existent and 0 indicating the non-existent of the feature.

3.3.4 Feature Selection

In classification problems, not all the features in the dataset are important. Some features present incorrect correlations and some may even be redundant, and thus can causes problems in the classification process. With the extra features present in the dataset, the computation complexity increases and may also impact the overall accuracy of the classification system. Feature selection selects a subset of the features present in the dataset, selecting the features which best classifies the data. The feature selection process improves the performance of the trained model as well as provides a fast and cost effective model.

F-statistic method of feature selection is used which is appropriate for numerical input and categorical data. It is a univariate feature selection which selects k best features based on the univariate statistical tests.

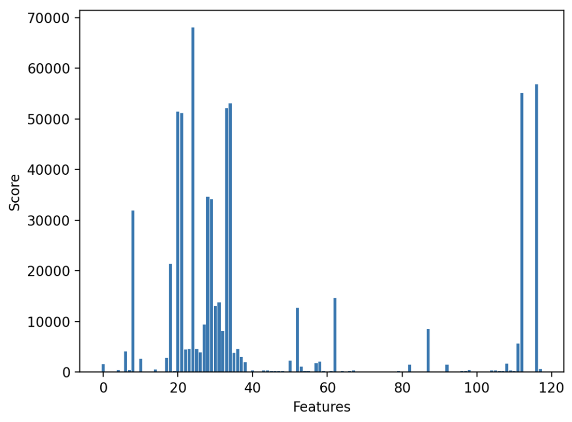

Figure 2: Scores of the features of KDD dataset using F-test

The Figure 2 shows an example of the score of the features obtained by F-test for KDD dataset As feature selection, k best features with the highest scores are selected.

3.3.5 Resampling Unbalanced Dataset

Oversampling of the minority class using Synthetic Minority Oversampling Technique (SMOTE)[8] is used to tackle the problem of class imbalance. In order to prevent the low prediction accuracy on the minority class, which are anomalies, the aim is to randomly increase the minority example in the dataset by replicating them and thus balance the class distribution. The training examples in the minority classes are synthetically generated by SMOTE by linear interpolation and randomly selecting one or more k-nearest neighbour of the minority class for each example.

3.3.6 Training

The system is trained using neural network, multilayer perceptron. It learns by passing input through a series of weighted neuron layers and adjusting their weights to successfully replicate the desired pattern in output. The output of a single neuron is the weighted sum of the inputs and a bias, and applied by an activation function which helps to introduce non-linearity to the model.

Mathematically, it can be represented as:

$$y_{out} = f(b + \sum_i w_i x_i)$$

where, f is the activation function (ReLU, Sigmoid, etc.) and b is the bias

A basic neural network is multi-layer perceptron. It is a supervised learning algorithm that can learn non-linear models as well making them able to perform multiclass classifications.

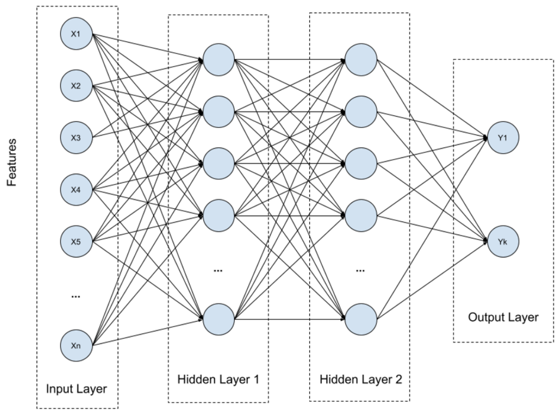

Figure 3: Multi-layer perceptron

The topology of the neural network consists of two hidden layers with 100 neurons in each hidden layer. The output layer size is five for multiclass classification and two for binary classification. The MLPClassifier provided in scikit-learn library in Python is used which implements MLP algorithm that trains using backpropagation. The hidden layers are activated using ReLU function given as,

$$ReLU(x) = max(0,x)$$

The MLPClassifier takes the sigmoid function for activation for output layer in binary classification.

$$sigmoid(x) = \frac{1}{1+e^{-x}}$$

For a multiclass classification with more than two classes, the MLPClassifier passes through a softmax function instead of logistic function.

$$softmax(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_i}}$$

3.4 Analysis Metrics

To calculate the performance of the proposed system, we will use Accuracy, Precision, Recall, F1-Score. Accuracy is the closeness of measurement to a specific value, given as,

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

Precision is the fraction of correctly classified positive examples divided by the number of examples labeled by the system, given as,

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

Recall, also known as true positive rate or sensitivity, is the probability that the model correctly identifies the anomaly detected.

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

F1-Score is the harmonic mean of precision and recall. It gives the single measure of comparison and higher is better.

$$F1-Score = \frac{2*Precision*Recall}{Precision+Recall}$$

3.5 Parameters Optimization

The parameters of the model need to be searched for the best validation score in the hyper-parameter space. For finding the optimum parameters for the best model, the GridSearchCV provided by scikit-learn is used. It is an exhaustive search technique which exhaustively searches over all the specified parameter values in the estimator. The search consists of an estimator, a cross-validation, parameter space and score function. The GridSearchCV consider all the parameters combinations and is slower at finding the optimal parameter combinations. A 3 fold cross-validation with training, test and validation set was used with different number of neurons and hidden layers, and different learning rates. GridSearchCV was used to run through all the possible combinations and find the optimal parameters giving the best score.

4. Results

The MLPClassifier was implemented using python3 and scikit-learn library of python using 2.3 GHz 8-core 9th-generation Intel Core i9 processor, with 16 GB RAM. The model was analyzed with the test dataset.

The distribution of training and test dataset present in the NSL-KDD shows that there is relatively large number of normal and DoS attacks data present. However, the R2L and U2R data is very low in the training dataset. This class imbalance was tackled by oversampling of the minority records in the training dataset using SMOTE.

Table 3: Attack Class Classification Accuracies for KDD

Attack Class	Accuracy
Normal	96%
DoS	83%
Probe	59%
R2L	6%
U2R	7%

Table 4: Binary Classification Accuracies for KDD

Class	Accuracy
Normal	92%
Attack	68%

The training dataset is used to fit the classifier model. The model is evaluated and the classifier's overall accuracy on the test dataset is displayed in Table 3 for multiclass attack classification and Table 4 for binary classification.

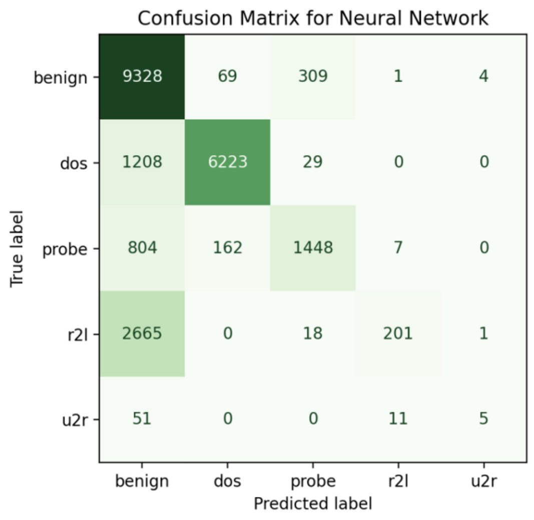

Figure 4: Confusion Matrix for Attack Class Classification for KDD

The same confusion matrices results are shown in Figure 4 for attack class classification and Figure 5 for binary classification. The confusion matrix for attack class classification shows that 9,328 items are correctly classified as normal class, 6,223 are correctly identified as DoS, 1,448 as probe, 20 as R2L and 5 as U2R. Similarly, the confusion matrix for binary classification shows that 8,983 items are correctly identified as normal and 8,763 items are correctly identified as attacks.

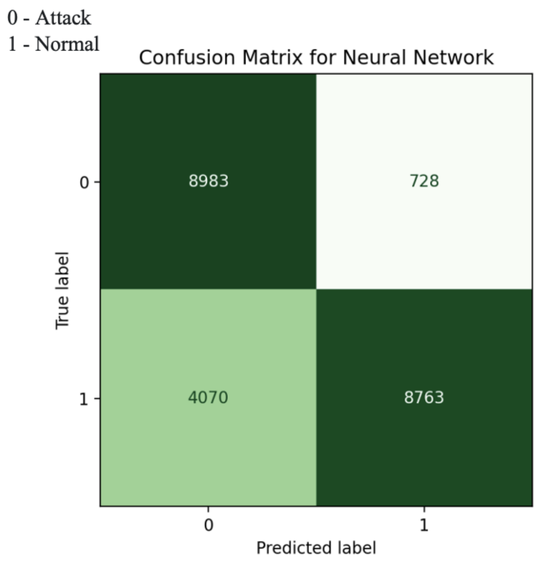

Figure 5: Confusion Matrix for Binary Classification for KDD

The evaluation of the classifiers' performance is done using the metrics Precision, Recall and f1-score as aforementioned. The performance measure for attack class classification is shown in Table 5 and for binary classification is shown in Table 6.

Table 5: Performance Measure for Attack Class Classification for KDD

Attack Class	Precision	Recall	f1-Score
Normal	67%	97%	80%
DoS	96%	84%	90%
Probe	86%	60%	71%
R2L	94%	8%	14%
U2R	25%	25%	25%

Table 6: Performance Measure for Binary Classification for KDD

Attack Class	Precision	Recall	f1-Score
Normal	69%	9%	79%
Attack	92%	68%	79%

In the experiment, the model for binary classification (classification if it is attack or normal) and multiclass classification (classification among 4 types of attack classes and normal) were evaluated. From the above observations it can be seen that in case of binary classification, the accuracy for normal class is higher than for attack class. However, the precision and recall for the attack class is higher than for normal class. Similarly, for multiclass classification,

classification accuracy is highest for normal class and for R2L and U2R it is relatively low. The precision for DoS is higher and recall is higher for normal class. The f1-score is higher for DoS class.

Similarly, the models were trained for UNSW-NB15 dataset for multiclass classification (classification among 9 types of attack classes and normal) and binary classification (classification if it is attack or normal) and were evaluated. The distribution of training dataset and test dataset present in the UNSW-NB15 dataset shows that there is relatively large number of normal and generic classes in the dataset. However, the Worms and Shellcode have relatively low data compared to others in the dataset. It can be seen that there is class imbalance in the UNSW-NB15 dataset as well which was tackled by oversampling of the minority records in the training dataset using SMOTE.

Table 7: Attack Class Classification Accuracies on UNSW-NB15 dataset

Attack Class	Accuracy
Analysis	18%
Backdoor	18%
DoS	11%
Exploits	73%
Fuzzers	49%
Generic	97%
Normal	72%
Reconnaissance	80%
Shellcode	68%
Worms	27%

The training dataset is used to fit the classifier model. The model is evaluated and the classifier's overall accuracy on the testing dataset is displayed in Table 7 for multiclass attack classification and Table 8 for binary classification.

Table 8: Binary Classification Accuracies on UNSW-NB15 dataset

Class	Accuracy
Normal	71%
Attack	96%

The same confusion matrices results are shown in Figure 6 for the attack class classification and Figure 7 for binary classification. The confusion matrix for attack class classification shows that 23,330 items are correctly classified as normal class, 17,343 are correctly identified as Generic, and so on. Similarly, the confusion matrix for binary classification shows that 43,703 items are correctly identified as normal and 26,607 items are correctly identified as attacks.

Table 9: Performance Measure for Attack Class Classification for UNSW-NB15

Attack Class	Precision	Recall	f1-Score
Analysis	6%	18%	9%
Backdoor	4%	18%	7%
Dos	30%	12%	17%
Exploits	63%	74%	68%
Fuzzers	25%	50%	34%
Generic	99%	97%	98%
Normal	94%	72%	82%
Reconnaissance	81%	81%	81%
Shellcode	28%	68%	39%
Worms	17%	27%	21%

Table 10: Performance Measure for Binary Classification for UNSW-NB15

Attack Class	Precision	Recall	f1-Score
Normal	81%	96%	88%
Attack	94%	72%	82%

The evaluation of the classifiers' performance for the UNSW-NB15 dataset is also done using the same evaluation metrics as aforementioned. The performance measure for attack class classification of UNSW-NB15 dataset is shown in Table 9 and for binary classification is shown in Table 10.

In the experiment, the model for multiclass classification (classification among 9 types of attack classes and normal) and binary classification (classification if it is attack or normal) were evaluated. From the above observations it can be seen that the precision and recall for the attack class is higher than for normal class. For multiclass classification, accuracy is highest for generic class and for Analysis, Backdoor is relatively low. It was seen that the precision, recall and f1-score for generic attack class is higher. The generic attacks are able to be detected with more precision, and the false negative cases are less for generic class, which means the model is able to correctly identify these anomaly detected. In case of binary classification, the accuracy for attack class is higher than for attack class. The precision is higher for attack class and recall for the normal class is higher than for attack class. The f1-score is found to be higher for normal class.

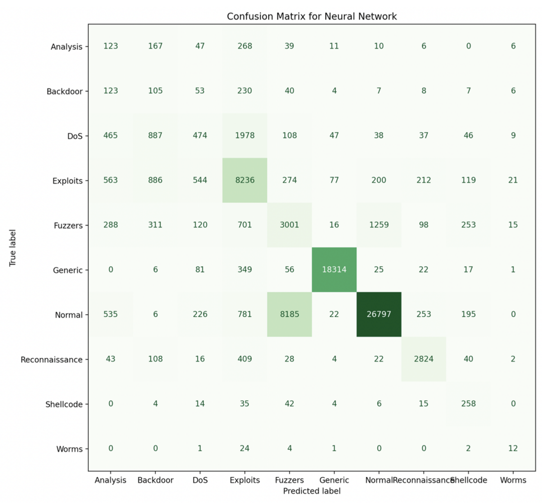

Figure 6: Confusion Matrix for Attack Class Classification of UNSW-NB15

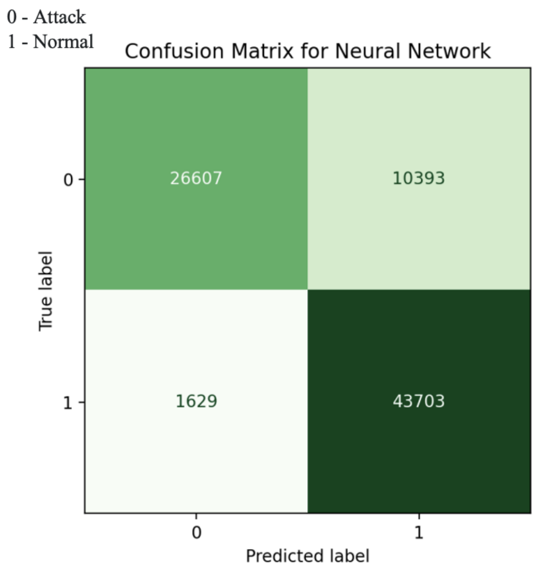

Figure 7: Confusion Matrix for Binary Classification of UNSW-NB15

5. Conclusion

An anomaly detection system was created using Multiplayer Perceptron. The KDD dataset is about two decades old and does not apprehend the modern data and attack cases in computer networks. Thus, the UNSW-NB15 dataset in addition to the NSL-KDD dataset was also used. The neural network was trained on both the NSL-KDD dataset and UNSW-NB15 dataset and evaluated. The experimental results show that Grid Search found the optimum parameters for the neural network to be 2 hidden layers with 100 neurons in each layer and learning rate of 0.001; and performs fairly well for both multiclass and binary classification. The trained Multilayer Perceptron is able to classify the 5 classes for KDD dataset and 10 classes for UNSW-NB15 dataset. Grid search was employed to search for the optimal parameters among a set of parameters. The neural network, however, has been optimized using grid search to search through a small set of parameters. To find the best model, a search through a bigger set of parameters is required.

References

[1] David J. Weller-Fahy, Brett J. Borghetti, and Angela A. Sodemann. A survey of distance and similarity measures used within network intrusion anomaly detection. *IEEE Communications Surveys Tutorials*, 17(1):70–91, 2015.

[2] Abdulaziz I. Al-issa, Mousa Al-Akhras, Mohammed S. ALsahli, and Mohammed Alawairdhi. Using machine learning to detect dos attacks in wireless sensor networks. In *2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT)*, pages 107–112, 2019.

[3] Iftikhar Ahmad, Mohammad Basheri, Muhammad Javed Iqbal, and Aneel Rahim. Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection. *IEEE Access*, 6:33789–33795, 2018.

[4] Rafath Samrin and D Vasumathi. Review on anomaly based network intrusion detection system. In *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, pages 141–147, 2017.

[5] Kdd cup 1999 data. `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`, 1999. [Online; accessed 2020].

[6] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the kdd cup 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pages 1–6, 2009.

[7] Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 Military Communications and Information Systems Conference (MilCIS)*, pages 1–6, 2015.

[8] Nitesh Chawla, Kevin Bowyer, Lawrence Hall, and W. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *J. Artif. Intell. Res. (JAIR)*, 16:321–357, 06 2002.