Intrusion Detection System Using Machine Learning (Final Report)

Vasisht Duddu
2015137
IIIT-Delhi, India
vasisht15137@iiitd.ac.in

Shubham Khanna
2015179
IIIT-Delhi, India
shubham15179@iiitd.ac.in

Anubhav Jain
2015129
IIIT-Delhi, India
anubhav15129@iiitd.ac.in

1. Introduction

With increasing complexity of systems, the threats to these systems are increasing exponentially. Attackers are resorting to more sophisticated malware to avoid detection mechanisms and exploit systems of critical importance. Attackers and malware writers keep coming up with new sophisticated malware like polymorphic and metamorphic malwares which are capable of changing their code dynamically. This creates a need to use advanced defence mechanisms to deal with unknown problems. Present malware detection schemes use heuristics and rely on hard coded features for detecting malware.

With the growth of machine learning, the detection can be improved by training the ML model to learn from past data an predict/classify files as malware or benign. This approach allows to protect against attacks which may not have been seen before.

We previously explored using machine learning techniques to detect malware in Windows PE32 executable files. We trained multiple models on the data set and optimized it for best possible generalization and found gradient boosted trees to perform well as compared to he state-of-the-art.We trained other models like random forests, logistic regression, SVMs, Ada-boost Classifier and Gradient Boosted Trees.. We found that gradient boosted trees gave us the best results with a classification accuracy of 98.45% and a false positive of 0.71%.

In this work, we explore using neural network to classify the network traffic as benign or malicious. We use multiple neural network architectures and evaluate their performance by using classification accuracy and false alarm rate using confusion matrix. We further evaluate the trained model using the AUC score and ROC curves.

2. Related Work

A lot of research interest has developed tin the area of intrusion detection sysem using machine learning algorithms. Some of the research and state-of-the-art results can be seen in the table.

Moustafa *et al.*[3][4] proposed a novel feature selection algorithm which is used to remove the irrelevant features and extract the most relevant features from the UNSW dataset for Intrusion Detection Systems.

Mogal *et al.* used Central Points of attribute values with apriori algorithm to select high ranked feature and remove irrelevant features on the UNSW dataset.

Source	Model	Accuracy	False Positives
Chowdhury *et al.*[1]	SVM	88.03	4.2
Chowdhury *et al.*[1]	SVM(with processing)	98.76	0.09
Moustafa *et al.*[4]	Expectation-Maximisation clustering	77.2	13.1
Moustafa *et al.*[4]	Logistic Regression	83.0	14.5
Moustafa *et al.*[4]	Naive Bayes	79.5	23.5
Mogal *et al.*[6]	Naive Bayes	99.96	-
Mogal *et al.*[6]	Logistic Regression	99.89	-

3. Data Set

Describe the dataset you are using, no. of samples in training, validation and test set. If you are extracting features, please describe the ones you have already explored in a subsection. In another subsection, you should specify what evaluation metrics are you going to use.

Category	Training Set	Testing Set
Normal	56,000	37,000
Analysis	2,000	677
Backdoor	1,746	583
DoS	12,264	4089

Category	Training Set	Testing Set
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

The UNSW-NB15 data set has several advantages when compared to the other data sets. First, it contains real modern normal behaviors and contemporary synthesized attack activities. Second, the probability distribution of the training and testing sets are similar. Third, it involves a set of features from the payload and header of packets to reflect the network packets efficiently.

Hence, we decided to use this dataset as compared to others.

4. Evaluation

4.1. Feature extraction

Network packets consist of a wide variety of features which negatively affects detection of anomalies. These features include some irrelevant or redundant features which reduce the efficiency of detecting attacks, and increase False Alarm Rate (FAR).

We used feature selection to reduce the 49 features to a smaller set of feature to get the most relevant features for differentiating benign network traffic from malicious network traffic.

For extracting the features, we can manually plot histograms for each feature and check if it can help to classify the result. Another method is to remove each feature and see the effect on the outcome.

For this dataset, we used the Tree-based feature selection for extracting the relevant features.

4.2. Measures Used

We used four measures for evaluating classification: TP (true positive) is the number of correctly classified attacks, TN (true negative) is the number of correctly classified normal records, FP (false positive) is the number of misclassified attacks, FP (false negative) denotes the number of misclassified normal records. We used confusion matrix to compute these values.

The accuracy is the percentage of the correctly classified records over all the rows of data set, whether correctly or incorrectly classified.

We further compute precision and recall from these values: precision is the fraction of correctly classified attacks to all attack records and the recall is the fraction of correctly classified attacks to the number of correctly classified attacks and misclassified attacks.

Accuracy=(TP+TN)/(TP+TN+FP+FN)

Misclassification=(FP+FN)/(TP+TN+FP+FN)

Sensitivity=(TP)/(TP+FN)

Specificity=,(TN)/(TN+FP)

False Positive=(FP)/(TN+FP)

Precision Score=(TP)/(TP+FP)

We further used the ROC curve and AUC score to quantify the model quality and performance.

5. Methodology

We started by processing the data by removing some features that were not relevant. We did not consider the attack category, protocol, state and service used. We found the dataset to be non-linearly seperable after analyzing the data.

We tried to use pre-processing techniques like standard scalar method, however, the accuracy we obtained after training was 100%. Hence, we did not use any other pre-processing techniques.

We tried multiple architectures for neural networks and changed the values of learning rate, learning momentum and dropout rate to improve the performance. Further, we used different activation functions like ReLU, Sigmoid, Tanh and Softmax at the output as well as a combination of all the above. We used 25 iterations to train the neural network.

The table below shows the architectures and corresponding activation functions that were used for training the neural network.

Based on the results, we tried to improve the performance of the best neural network. However, we could not improve the performance the models by a significant amount and we observed a trade-ff between the model accuracy and the false positives.

6. Results and Analysis

The number of iteration were between 25-30. This is less and we can increase the number of iteration to improve the performance of the model. Trying a few more architectures could have helped us to improve the performance of the model.

We observed that when our model's FAR was low the accuracy was low as well. We were unable to get a model that had a decent accuracy as well as a low FAR.

Our results were fairly close to some of the previous work done before. When we compare the FAR, we had the number of false positives to 7.47% which is comparable to the previous work done. However, the model's accuracy was a little low.

The best accuracy we achieved was 88.29% however, the model had a very high false positives and it is not advisable to use that model for intrusion detection.

The best model would thus be NN3 which has a low FAR but has a low accuracy. To improve the model accuracy we could use more number of iterations and change some parameters.

We have plotted the ROC curves of each neural network and plotted the iteration vs loss during the training.

7. Contributions

We have successfully completed our goals for the project and have completed all the deliverable for the project. We

ID	Architecture	Activation Functions	Accuracy	False Positives	AUC Score	Other
NN1	200,150,50	Sigmoid	88.29	32	83.72	learnrate=0.001
NN2	300,200,150,100,50,10	ReLU	88.21	32.58	83.70	learnrate=0.001
NN3	300,200,150,100,50,10	Tanh	75.55	7.47	79.25	learnrate=0.001
NN4	200,150,150,50,10,2	Sigmoid, ReLU, Tanh, Softmax	85.69	39.57	80.15	learnrate=0.01, Dropout=0.45
NN5	150, 300, 450, 50	Sigmoid, ReLU, Softmax	73.92	11.16	77.19	learnrate=0.0001,learnmom=0.9,dropout=0.45,

Table 1. Results and Architecture of Neural Networks Used

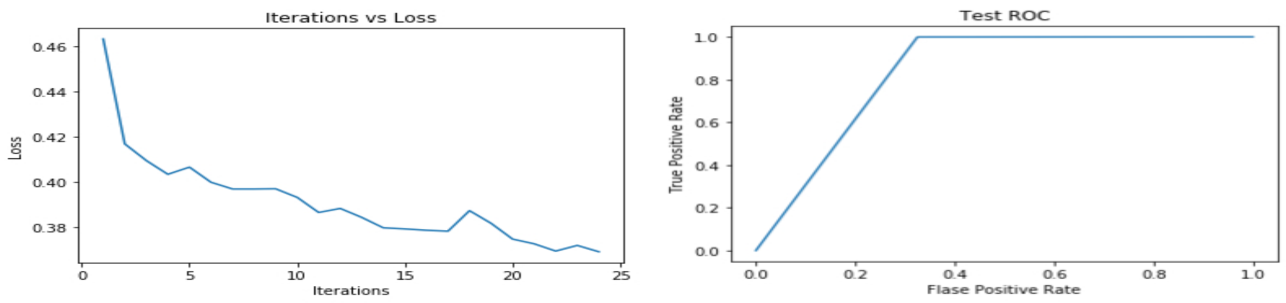

Figure 1. NN1: Iteration vs Loss and ROC curve

proposed to work on two datasets: PE32 malware dataset and UNSW dataset. We proposed to use multiple models on malware dataset and compare their performance. We observed that the performance of linear models were very poor as compared to other ensemble approaches and our results were close to the state-of-the-results. In the second half of the project, we proposed to work on UNSW network traffic dataset and explore using various neural network models which we completed successfully.

Following are the roles each member have followed and will follow for the rest of the project duration and evaluation:

- Vasisht Duddu: Model and parameter selection, feature Extraction, training and analysis for malware and network anomaly dataset
- Anubhav Jain: Data visualization for malware dataset and data processing for network anomaly dataset
- Shubham Khanna: Parameter tuning and learning curve analysis for malware dataset

Below is the details with respect to each file contributed by the team members for working on both the data-sets:

- Vasisht Duddu: NN1.ipynb, NN2.ipynb, NN3.ipynb, NN4.ipynb, NN5.ipynb, gradient_boosted_trees.ipynb, random_forest.ipynb, logistic_regression.ipynb
- Anubhav Jain: visualize.py, Reading data for UNSW Dataset
- Shubham Khanna: learning_curve.py, gradient_boosted_trees_version2.ipynb, random_forest_version2.ipynb, logistic_regression_version2.ipynb

References

[1] UNSW-NB15 data set: https://www.unsw.adfa.edu.au/australian-centre-for-cyber-security/cybersecurity/ADFA-NB15-data sets/

[2] Nour Moustafa, Jill Slay , ”UNSW-NB15: A Comprehensive Data set for Network Intrusion Detection systems”

[3] Nour Moustafa, Jill Slay , ”The significant features of the UNSW-NB15 and the KDD99 Data sets for Network Intrusion Detection Systems”

[4] Nour Moustafa, Jill Slay , ”A Hybrid Feature Selection For Network Intrusion Detection Systems: Central Points And Association Rules”

[5] Md Nasimuzzaman Chowdhury and Ken Ferens, Mike Ferens, ”Network Intrusion Detection Using Machine Learning”

[6] D.G. Mogal, S.R Ghungrad, B.B Bhusare, ”NIDS using Machine Learning Classifiers on UNSW-NB15 and KDDCUP99 Datasets”

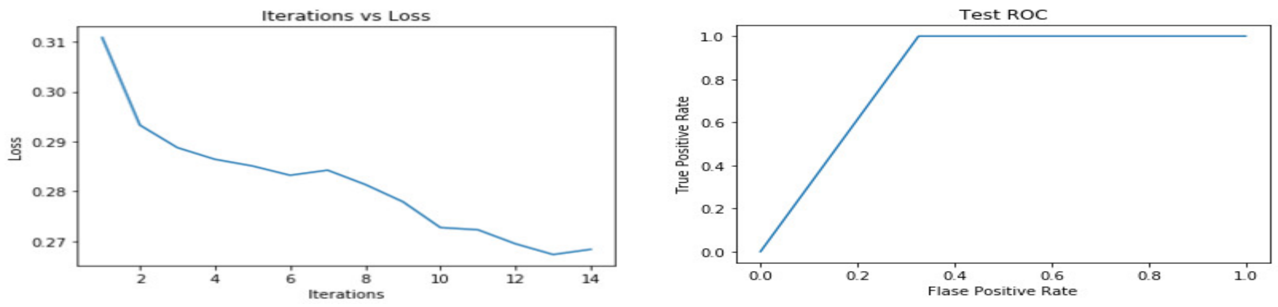

Figure 2. NN2: Iteration vs Loss and ROC curve

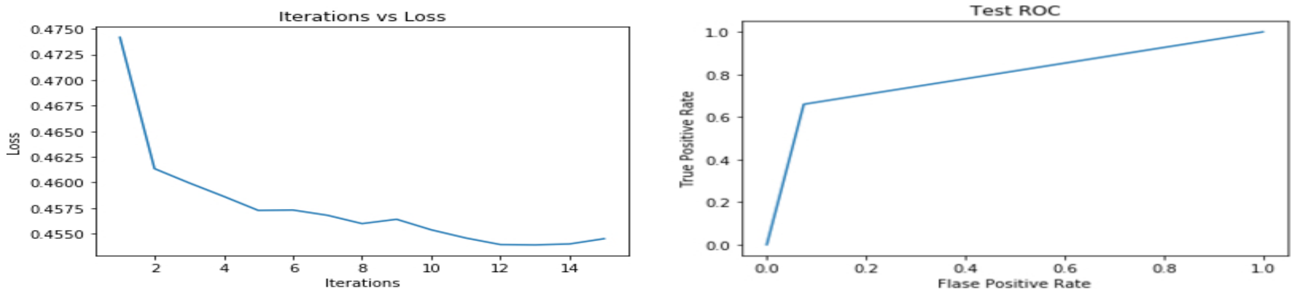

Figure 3. NN3: Iteration vs Loss and ROC curve

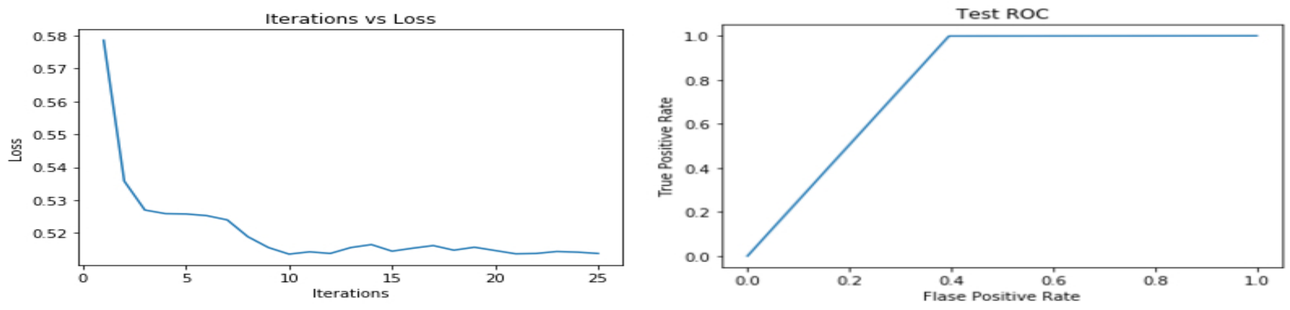

Figure 4. NN4: Iteration vs Loss and ROC curve

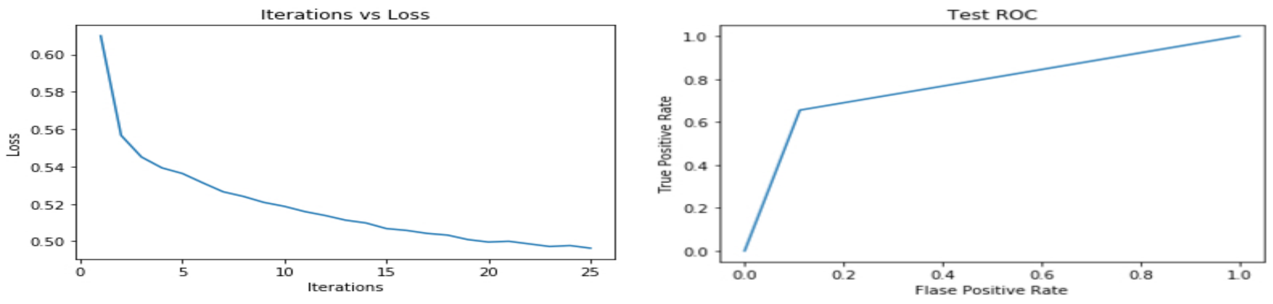

Figure 5. NN5: Iteration vs Loss and ROC curve