Jilin Daxue Xuebao (Gongxueban)/Journal of Jilin University (Engineering and Technology Edition)
ISSN: 1671-5497
E-Publication: Online Open Access
Vol: 41 Issue: 12-2022
DOI 10.17605/OSF.IO/A76HZ

MACHINE LEARNING TECHNIQUES FOR SENSING IOT BOTNETS

GADDE RAMESH

Research Scholar, Dept., of CSE, University College of Engineering (Autonomous)-UCE
Osmania University, Hyderabad, Telangana State. Email: gadde.ramesh@gmail.com

Dr. SURESH PABBOJU

Professor of Information Technology, Chaitanya Bharathi Institute of Technology-CBIT,
Hyderabad, Telangana State. Email: plpsuresh@gmail.com

Abstract

The usage of the Internet of Things (IoT) in today's world has led to several stern security issues like denial-of-service attacks by a huge collection of compromised IoT devices. Due to lack of proper security and unavailability of packet filtration, the IoT devices are easily compromised and can be a member of the zombie network. In spite of addressing several techniques in detecting IoT botnets, unaddressed challenges are still open to researchers. In this paper, a few machine-learning methods are introduced to recognize the existence of IoT botnets effectively. The machine-learning model detects the prediction of the IoT botnets based up on the information on the network traffic. Our proposed model achieves less false positive for faster detection by using feature selection. The Random Forest came up with an accuracy of 94.47 percent, which performed much better than other deep learning and machine learning models and, thus, can be measured as a suitable explanation to effectually sense the IoT botnet with a lesser detection rate.

Keywords: Machine Learning, Deep Learning, dense neural network, random forest, KNN, feature selection, dimensionality reduction, Internet of Things (IoT), botnet, IoT botnet, AdaBoost, IoT botnet detection.

1. INTRODUCTION

Several smart devices like computers, mobile phones, coffee makers, video surveillance cameras, and home thermostats [20] and [21] are allied to the Internet and the sum is increasing every day. As the applications of Internet of Things has widely increased in the fields like office automation, home automation, automobiles automation, energy management, healthcare sector[1] and many more, the concept IoT technology is evolving constantly. This is accelerating towards infinite devices connecting to the internet daily.

The poor sophisticated architectural structure of the security embedded in IoT devices is easily targetable by attackers. The cyber-attacks on these devices are drastically increasing, as the scope to enhance the security of these devices is less. The detection of botnets timely is critical to minimize the related risks. Identification of IoT botnet with the help of information from network traffic by employing machine-learning techniques is the main aim of this research. Figure 1 demonstrates the idea of my research plan of action.

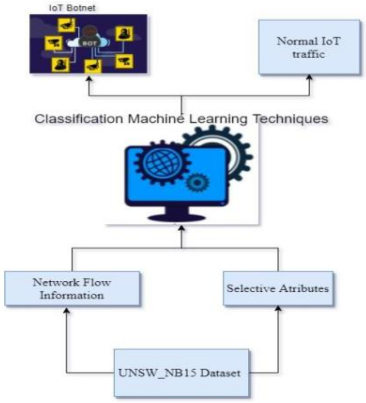


Figure 1: A block map to signify machine learning practices castoff for IoT botnet classification

1.1 Objective of research

The objectives to perform this research work are mentioned as:

- Employ feature engineering to retrieve the critical feature and thus reduce the dataset's dimensionality.
- Incorporating the correlation method of Spearman and injecting these choosy features into the classification model to foresee the IoT botnet results in feature selection.
- On the feature-engineered data, the dense neural network model, K-nearest neighbours (KNN) model, Adaboost model, and the Random Forest classifier model are used.
- To successfully complete assessment and rapid prediction, classification models are cross validated, optimized, and compared.
- The work is systematized by the literature survey by portraying the related work done in machine learning to detect IoT botnets,

1.2 Paper organization

This investigation effort is organized as: Section 2- describes the method used in this study, Section 3- explains the three-stage design workflow incorporated for this study, Section 4-discusses the various machine learning and deep learning models used, Section 5 displays the results and evaluations of each model, and Section 6 concludes the research work.

2. LITERATURE SURVEY

The Cisco’s survey summary reports that by the end of 2020, the count of machines coupled to the internet have crossed 50 billion and 1 trillion by 2022. Figure 2 illustrates the growth in usage of IoT devices.

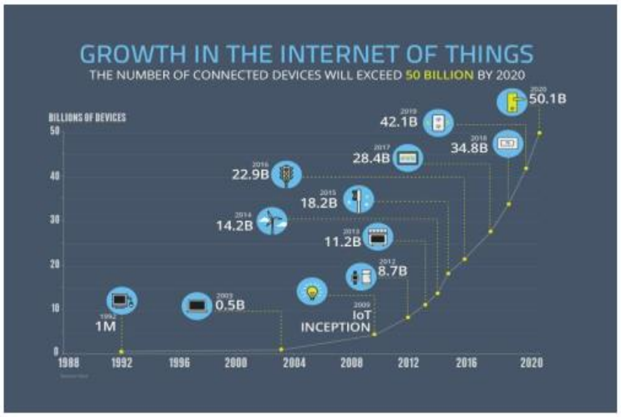


Figure 2: Illustration to show the gigantic progress of IoT devices [18]

The usage of IoT devices is increasing rapidly, and the constant attacks on those devices are also increasing, creating an ever-challenge for security professionals. The American Consumer Institute survey reports that nearly 10 routers are exposed to exploitation by hacking because their credentials are not being more assertive. The available IoT devices' computational power is used by most of the malware for implementing denial-of-service attacks on a large scale. Therefore, identifying and detecting the presence of IoT botnets can cut the opportunities for DoS attacks drastically.

2.1 Dataset

Earlier researchers [2] faced the problem of the unattainability of the current datasets for network traffic. Then, these researchers have come up with UNSW NB15 data. This is compared to the prevailing ones, which have already set a benchmark but are considered obsolete datasets. In contrast, NB15 performed better with high benefits than the datasets from other network intrusion detection systems like NSLKDD, KDDCUP99, and KDD98. The NIDS datasets do not end with acceptable outcomes when evaluated. Hence, the proposed effort by [3] addressed the topics, demonstrated feature correlation and statistical analysis, and applied several classifiers to assess the dataset's intricacy. The observations of NB15 are equated with KDD99 and concluded that KDD99 has several drawbacks.

The characteristics of the classifier are discussed in the other part. Among several classifiers, logistic regression and decision tree with an accuracy of 81 and 86 percent, respectively, exceeded artificial neural network, expectation maximum, and naïve Bayes algorithms. To generate better results than other classifiers, the work [4] implemented a 2-stage random forest on UNSW NB15. The UNSW NB15 tends to have the most significant features for the detection of IoT botnets, as it comprises real-time IoT network traffic details.

In the study [5] UNSW NB 15 was used to exhibit anomaly detection on massive neural networks using trapezoidal area estimation. By using principal component analysis (PCA) [5] mined 13 features and gained 92.8 percentage of accuracy with 5.1 percent of false positives, but few features such as source destination loads, time-to-live, transaction's duration, transaction rate weren’t taken into consideration [5], thus completely not using the UNSW NB15.

2.2 Techniques in Machine learning

In contrast to the traditional benchmark-based detection techniques, the evolution of modern IoT botnet detection techniques is necessary. Many researchers have given theirs contributed to tackling the issue by using various machine-learning methods to accomplish network forensics. In sensing the untruthful happenings of the botnets, the learning [6] illustrated using machine learning techniques such as ANN, ARM, naïve Bayes, and decision tree on UNSW NB15. The experiment resulted in the decision tree, in combination with protocols and source or destination IP addresses, effectually detecting the botnet.

In the process mentioned in [7] to offer imposition activity logs, honeynet was installed. This study extracted pcap files, which contain logs and network traffic dump. The collected data from honeynets was used by machine learning techniques to detect the botnet. This result marked that the random forest classifier was performing consistently better than other machine learning models. The limited terminuses and consistent time intermissions among packets, along with the technique of feature selection [8], showed that IoT-specific networks behaved with higher accuracy. Though linear SVM, random forest, KNN, ANN, and decision tree were cast off to categorize among standard IoT packets and DoS attack packets with 99.99-percentage accuracy, random forest outperformed with a consistent output. The study [9] worked to improve the prediction of accuracy on 84,030 different occurrences; the random forest was implemented and showed an accuracy of 99.7 percent.

In the paper [10], a random forest-based PSI-rooted sub graph-based feature for IoT botnet detection is discussed, which resulted better than other classification models with 98 percent true positives and the best ROC-AUC value of 97 percent. The hyperactive parameters [4], [22] used in the random forest are the number of features, the extreme depth of trees, and the number of features. The random forest can handle the larger databases by reducing the over fitting problem with fewer model parameters [11].

A model briefed in [12] can handle DNS query flows on a large scale using a random forest algorithm and achieved a 0 percentage of false positive rate and a 4.36 percentage of false negative rate. This made the model a perfect model suitable for real-time detection, which resulted in higher accuracy and high precision with the UNSW NB15 dataset. It is evident that the random forest algorithm performed better than other models. Thus, the random forest was chosen in our research work to detect the IoT botnet traffic effectively with less training and less recognition time. The dense neural network is also used, which results in improved accuracy than the neural networks, as noticed in [13], [14], [15].

2.3 Collection of feature

The technique of removing the input variables when developing a predictive model is feature selection. The classification highly desires dimensionality data reduction to improve the model's predictive analysis. The dimensionality reduction not only improves predictive analysis but also overcomes the computational cost of the model. To improve the accuracy of the IoT detection model, the feature selection technique plays a vital role by ignoring unwanted features. Out of 45 total features in the UNSW NB15 dataset, only 17 features were considered important and taken into count.

The work [6] selected one of the simplest feature selection methods, Information Gain, and used ten features for experimental purposes. The approach [1] lessened the number of features by illustrating the dimensionality reduction required to detect the IoT botnet. Using the model of auto encoder in the study [15], dimensionality reduction was accomplished, and the compression of the code layer among encoders and decoders resulted in informative data. A distributed approach to network anomaly was proposed by Palmieri [16], and the approach is based on the autonomous component analysis in mining the unseen features from multiple variety data, which carries information by components independently. Bashi and Nomm [1] used the unsupervised model to present IoT botnet detection. The feature selection was given importance in [1] by training a distinct model on all IoT devices to attain resource optimization rather than a dedicated model for every IoT device.

Our proposed model worked on several feature selection methods like entropy-based, Hopkins statistics, variance-based, and entropy-based methods. As the original dataset is unbalanced, a sampling technique was applied to obtain a balanced dataset. On the whole, the accuracy performance of SVM was better on the unbalanced dataset, and isolation forest in combination with entropy-based was top notch. In the arena of cyber security identification of threats is the primary aspect of a detection model. Thus, in our research, along with several machine-learning techniques, feature selection is also employed in order to attain quicker and more accurate detection, which makes the machine-learning model accomplish capably too.

3. RESEARCH METHODOLOGY

This part describes the methodologies used in expecting the occurrence of IoT botnets using several machine-learning techniques on the UNSW NB15 dataset. The work is based on KDD data mining.

Data Acquisition removes noisy and unwanted data, which helps in preparing meaningful data that is used in the detection of IoT botnets. The process of cleaning on UNSW NB15 dataset helps the model in better prediction as missing and noisy data performs weakly during prediction. Here XL tool is used to perform cleaning.

The collection of data from various sources to form one dataset is referred to as data integration. The training and testing csv files of UNSW_NB15 are merged to do my research. The training set size is 82332x45, testing set size is 175341x45 together, and they form 257673x45 with 44 features and the final target column, which holds the value

1 and 0 for the presence of IoT botnet and normal traffic, respectively. Finaldataset.csv has these 257673 rows x 45 columns. The selection of important attributes plays a vibrant part in the performance of the model, and hence the third step is data selection. An inbuilt class “feature importance” is implemented in the classifier to extract the important features. The Extra Tress Classifier is used in my work to suite few arbitrary decision trees into the dataset and does not over fit the data. Finaldataset.csv used wrapper-based feature selection to recognize the chief features required for the prototype. The selected were uploaded into a data frame and using the sea born library, the correlation matrix was designed. The Spearman correlation matrix was merged to depict the relationships among the extracted features from the previous part.

In the Feature engineering part, Extra Tree Classifier yielded 10 important network features such as spkts, dttl, dload, dur, swin, rate, sbytes, sttl, sinpkt. The variables obtained using the Spearman correlation technique are used to illustrate a heatmap that shows the correlation between the variables. Based on the correlation matrix, only 17 out of 40-network traffic features were selected whose count is more than in [6], which were selected by the process of Information Gain. These 17 are strongly correlated, and the remaining is weakly correlated.

To extract the network traffic feature, several data mining techniques were implemented to classify the traffic of the IoT botnet from the normal accurately. The classification was employed in machine learning to predict a categorical variable from the data by 1 and 0 for the presence of the IoT botnet traffic and the presence of normal traffic, respectively. The same dataset is implemented using dense neural network, kNN, and AdaBoost classifier, random forest for quick and reliable prediction. Each classifier resulted in different outputs, and I also noticed that the model, which yielded better results, was the Random Forest. The evaluations are discussed in the next sections.

The evaluation is illustrated in the form of a confusion matrix, which describes the classification as the overall figure of instances the prototype correctly or incorrectly known as a subset of IoT botnet class as true positive or false positive and the overall figure of instances the model incorrectly/correctly recognized as not a subset of the IoT botnet class as false negative or true negative respectively.

3.1 Implementation

To implement the proposed work in figure 3, we have chosen a design that comprises the following:

a) Data preparation stage, which extracts, cleans, and merges the data, including feature selection using the spearman correlation technique. The source csv file is opened in ipynb using python.

b) Modelling stage has various techniques of deep learning and machine learning employed on the UNSW NB15. The model’s optimization is performed by means of AdaBoost and Stochastic Gradient Descent. The chosen evaluation metrics are accuracy, recall, precision, AUC-ROC, F1 score and confusion matrix.

c) The obtained results are visualized in graphs, plots.

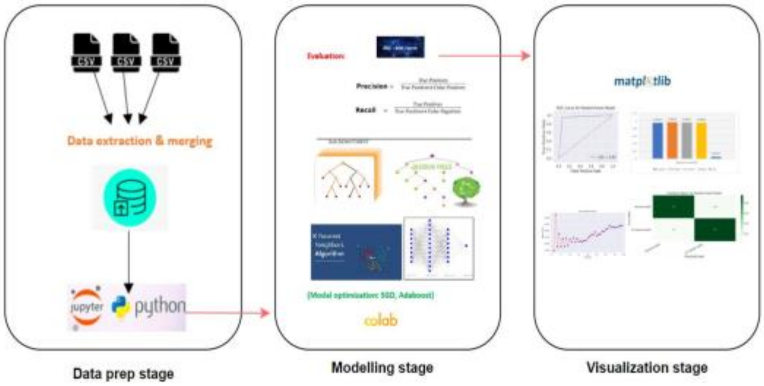


Figure 3: The design pattern to identify the IoT botnet.

The execution in detecting the IoT botnet worked in three phases (i) the preparation of data, (ii) the process of cross-validation, and (iii) the prototype's refinement using hyperactive parameters. The use of hyperactive parameters enhances the prediction rate of the proposed model. The results of diverse detection models are related to analyze the proper system.

(i) Data preparation: As mentioned earlier using various feature engineering techniques were applied to transform 49 featured dataset to 17 featured dataset and thus at the modelling phase, the dataset is prepared to have 257672 rows and 18 columns. The 17 featured dataset is loaded into a and b variables where a holds the only the information about the network traffic without target and b holds the information about the both. The a and b are spliited into 70% and 30% as training and testing sets respectively. The training set was trained using several classifiers. Using spearman's correlation technique, dimensionality reduction was realized which measures the strength of the relationship among a and b, and is defined as rs = −1 ≤ rs ≤ 1. This correlation technique achieved in obtaining 17 features that was enough for successfully predicting the traffic of IoT botnet. The below figure 4 illustrates the information about the network information for better visualization.

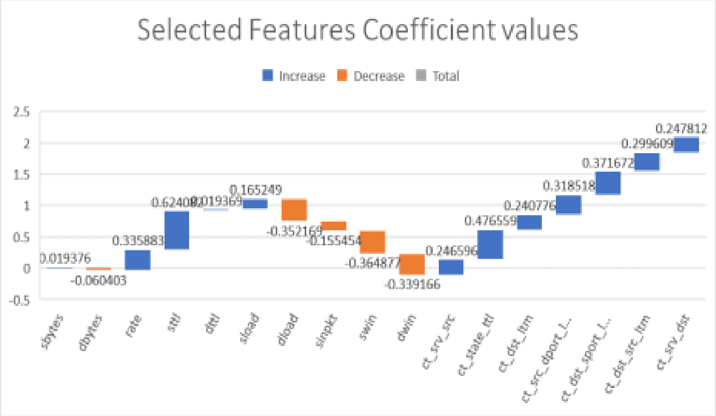


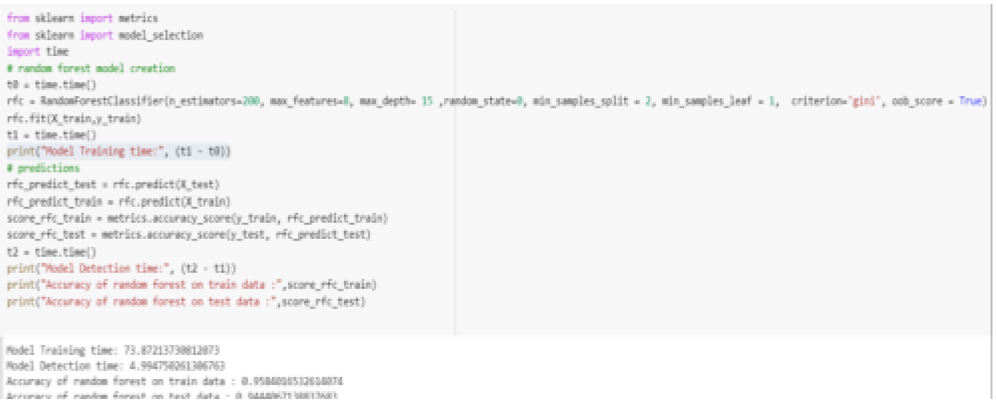
Figure 4: Plot illustrating extracted features coefficient values

The Table 1 below describes the significant features with their correlation coefficient values using tree based feature selection. Nevertheless, more perceptions were done using spearman’s correlation feature selection method by selecting 17 features that were used for accurate and faster detection of IoT botnets.

Table 1

S. No	Significant features	Correlation coefficient	Description
1	Sbytesd	0.019376	Amount of data transported between the source and the destination in bytes
2	Dbytess	-0.060403	Amount of data transported between the source and the destination in bytes
3	Trate	0.335883	The rate of transmission
4	Sttlv	0.624082	The original time remaining value
5	Dttlv	0.019369	The worth of the intended time to life
6	Loads	0.165249	Number of bits/second at the source end
7	Loadd	-0.352169	Number of bits/second at the destination end
8	Sintpktt	-0.155454	Interpacket arrival time in milliseconds at the source end
9	Swinv	-0.364877	TCP window value at the source
10	Dwint	-0.339166	TCP window value at the destination
11	Ncnt_srca_srv	0.246596	The number of connections with the same service and source address
12	Ncnt_st_ttlv	0.476559	The count of links between states and the worth of time to live
13	Ncnt_dst	0.240776	The figure of acquaintances in the same location has
14	Ncnt_src_dstn	0.318518	The figure of acquaintances to the destination port coming from the same source address.
15	Nct_dstn_src	0.371672	The count of acquaintances to the source port coming from the same destination address.
16	Ncnt_ssrc_sdstn	0.299609	The count of acquaintances coming from the same source to the destination address.
17	Ncnt_ssrv_dstn	0.247812	A measure of how many connections have the same service and destinations.

According to the name of the Random forest classifier, the model aggregates the results of the various decision trees and predicts the target’s ultimate class. This classifier by tuning the hyper parameters was included in the work for identifying the IoT botnet effectively. By using the time function the training time for the model and the detection time by the model is also taken into consideration. Figure 5 illustrates a sample of the tuned hyper parameters.

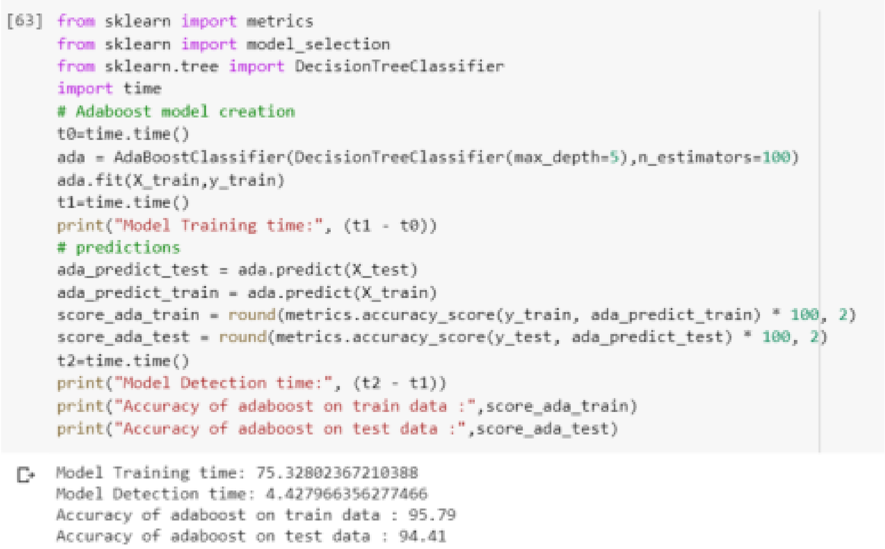


```
from sklearn import metrics
from sklearn import model_selection
import time
# random forest model creation
t0 = time.time()
rfc = RandomForestClassifier(n_estimators=200, max_features=8, max_depth= 15 ,random_state=0, min_samples_split = 2, min_samples_leaf = 1,  criterion="gini", oob_score = True)
rfc.fit(X_train,y_train)
t1 = time.time()
print("Model Training time:", (t1 - t0))
# predictions
rfc_predict_test = rfc.predict(X_test)
rfc_predict_train = rfc.predict(X_train)
score_rfc_train = metrics.accuracy_score(y_train, rfc_predict_train)
score_rfc_test = metrics.accuracy_score(y_test, rfc_predict_test)
t2 = time.time()
print("Model Detection time:", (t2 - t1))
print("Accuracy of random forest on train data :",score_rfc_train)
print("Accuracy of random forest on test data :",score_rfc_test)
```

```
Model Training time: 73.87213730812073
Model Detection time: 4.994750261306763
Accuracy of random forest on train data : 0.9584016532614074
Accuracy of random forest on test data : 0.9444067138883681
```

Figure 5: Random Forest model's snippet code

Ada Boost is an ensemble binary classifier, which converts a group of weaker classifiers into stronger one. Ada Boost boosts the presentation of any algorithm but performs best while working with the decision tree. The below figure 6 shows the implemented simple Ada Boost model with implemented hyper parameters.

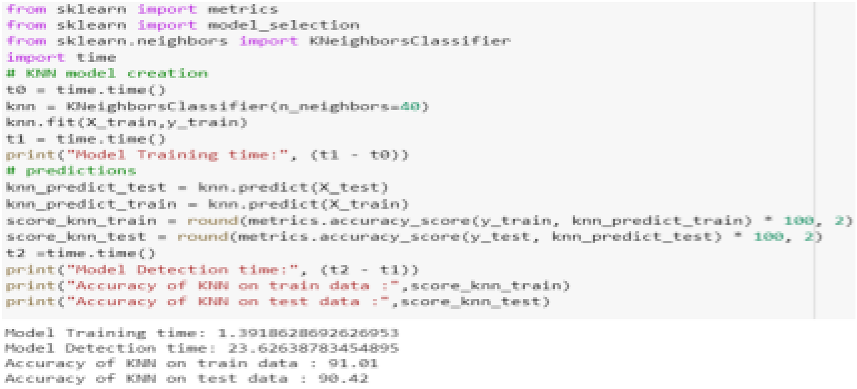


```
[63] from sklearn import metrics
     from sklearn import model_selection
     from sklearn.tree import DecisionTreeClassifier
     import time
     # Adaboost model creation
     t0=time.time()
     ada = AdaBoostClassifier(DecisionTreeClassifier(max_depth=5),n_estimators=100)
     ada.fit(X_train,y_train)
     t1=time.time()
     print("Model Training time:", (t1 - t0))
     # predictions
     ada_predict_test = ada.predict(X_test)
     ada_predict_train = ada.predict(X_train)
     score_ada_train = round(metrics.accuracy_score(y_train, ada_predict_train) * 100, 2)
     score_ada_test = round(metrics.accuracy_score(y_test, ada_predict_test) * 100, 2)
     t2=time.time()
     print("Model Detection time:", (t2 - t1))
     print("Accuracy of adaboost on train data :",score_ada_train)
     print("Accuracy of adaboost on test data :",score_ada_test)
```

```
     Model Training time: 75.32802367210388
     Model Detection time: 4.427966356277466
     Accuracy of adaboost on train data : 95.79
     Accuracy of adaboost on test data : 94.41
```

Figure 6: Ada boost classifier model's snippet code

The k-nearest neighbour classifier model and also a regression predictive model. kNN is also renowned as a lethargic classifier as much effort is not required during training and utilises complete dataset for training and classification. The value of k was initialised to 40 and later reinitialized to 15 depending on the plot of mean error rate. Figure 7 illustrates a sample model for kNN classifier.

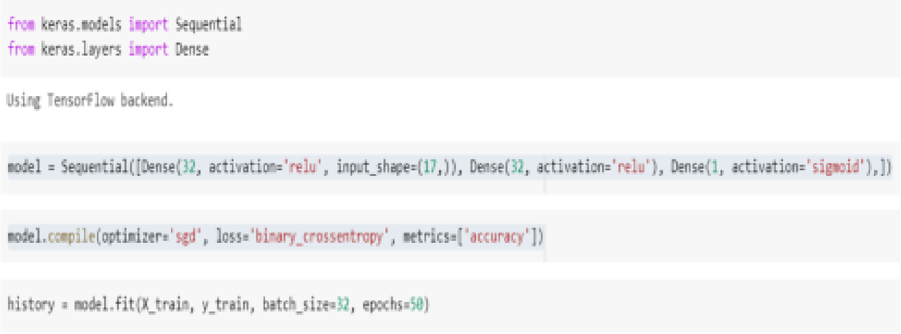


```
from sklearn import metrics
from sklearn import model_selection
from sklearn.neighbors import KNeighborsClassifier
import time
# KNN model creation
t0 = time.time()
knn = KNeighborsClassifier(n_neighbors=40)
knn.fit(X_train,y_train)
t1 = time.time()
print("Model Training time:", (t1 - t0))
# predictions
knn_predict_test = knn.predict(X_test)
knn_predict_train = knn.predict(X_train)
score_knn_train = round(metrics.accuracy_score(y_train, knn_predict_train) * 100, 2)
score_knn_test = round(metrics.accuracy_score(y_test, knn_predict_test) * 100, 2)
t2 =time.time()
print("Model Detection time:", (t2 - t1))
print("Accuracy of KNN on train data :",score_knn_train)
print("Accuracy of KNN on test data :",score_knn_test)
```

```
Model Training time: 1.3918628692626953
Model Detection time: 23.62638783454895
Accuracy of KNN on train data : 91.01
Accuracy of KNN on test data : 90.42
```

Figure 7: KNN classifier model's snippet code

The network in which neurons of a layer are related to the neurons of the next layer forms dense neural networks. The dense layers provide combine all the features from the previous layer. If the model has exactly a single input tensor and a single tensor, a sequential model is used. In this work, the sequential model was developed with 2 dense layers and one dense layer with ReLu and sigmoid activation functions. The figure 8 illustrates the snippet code of the dense neural network.



```
from keras.models import Sequential
from keras.layers import Dense
```

```
Using TensorFlow backend.
```

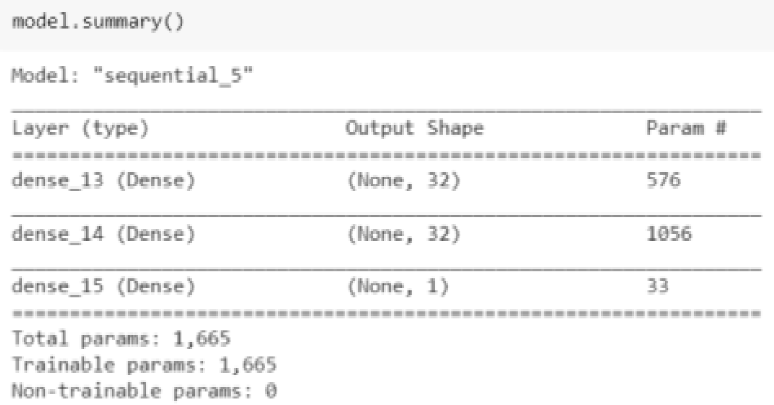
```
model = Sequential([Dense(32, activation='relu', input_shape=(17,)), Dense(32, activation='relu'), Dense(1, activation='sigmoid'),])
```

```
model.compile(optimizer='sgd', loss='binary_crossentropy', metrics=['accuracy'])
```

```
history = model.fit(X_train, y_train, batch_size=32, epochs=50)
```

Figure 8: Dense Neural network's snippet code

Keras classification metric was used to define the accuracy of IoT botnet classification. The binary cross entropy was mentioned to comprehend the performance of the defined sequential model. To analyze the error of the prototype and also to appraise the weights, stochastic gradient descent (SGD) optimizer was used. The size of the batch and the number of iterations were initialized to 32 and 50, respectively, to produce better and quicker results. The chosen characteristics in this study of our dense layers are illustrated in figure 9.



```
model.summary()

Model: "sequential_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_13 (Dense)             (None, 32)                576
_________________________________________________________________
dense_14 (Dense)             (None, 32)                1056
_________________________________________________________________
dense_15 (Dense)             (None, 1)                 33
=================================================================
Total params: 1,665
Trainable params: 1,665
Non-trainable params: 0
```

Figure 9: The model structure is demonstrated

4. THE PROCESS OF EVALUATION

The evaluation phase shows several trials steered to evaluate the act in terms of accurateness and to evaluate the performance of the classifiers used for our research. AdaBoost, Random Forest, and kNN classifiers were employed to identify the IoT botnet accurately. Their outputs are compared and visualized using ROC AUC curves and the confusion matrix.

When the random forest model was used, the performance is detailed in table 2 and in figures 10, 11, 12.

Table2

Classifier	Error	F1 score	Accuracy	Recall	Precision
Random forest	0.054	0.957	0.9455	0.9511	0.963

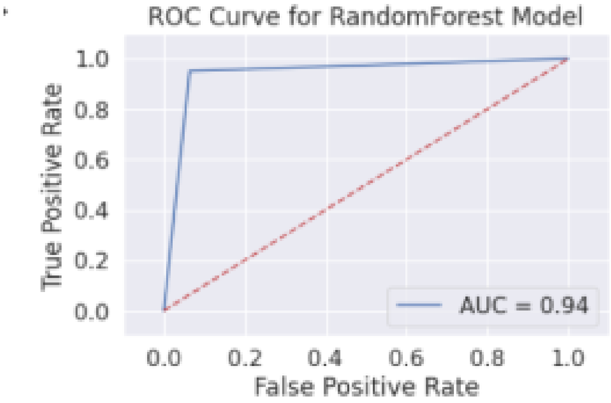


Figure 10: ROC Curve generated by Random Forest.

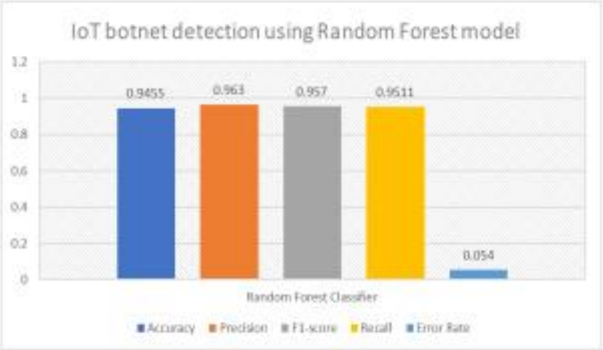


Figure 11: the evaluation graph generated by Random Forest Model

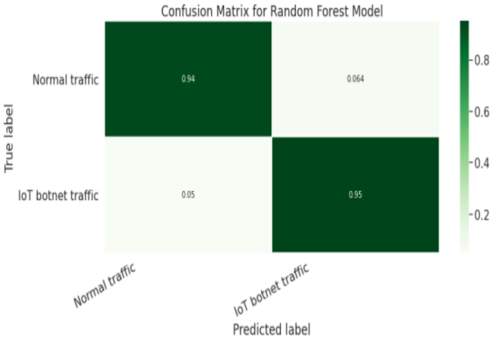


Figure 12: The confusion matrix plotted by the Random Forest.

When Ada Boost model was implemented, the performance is detailed in table 3 and in the figures 13, 14 and 15

Table3

Classifier	Error	F1 score	Accuracy	Recall	Precision
AdaBoost	0.057	0.9551	0.9429	0.9528	0.9576

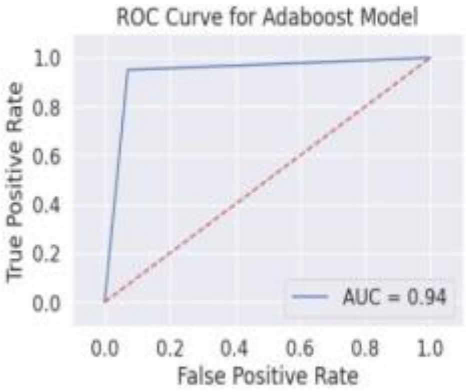


Figure 13: The ROC curve generated by AdaBoost

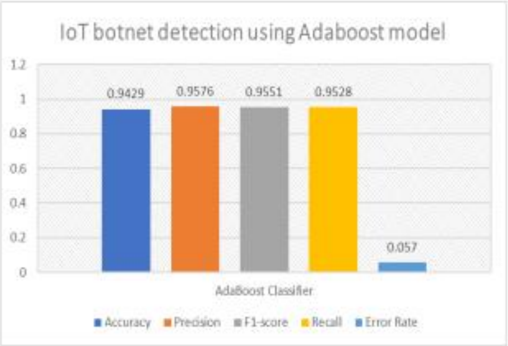


Figure 14: The evaluation graph generated by AdaBoost model

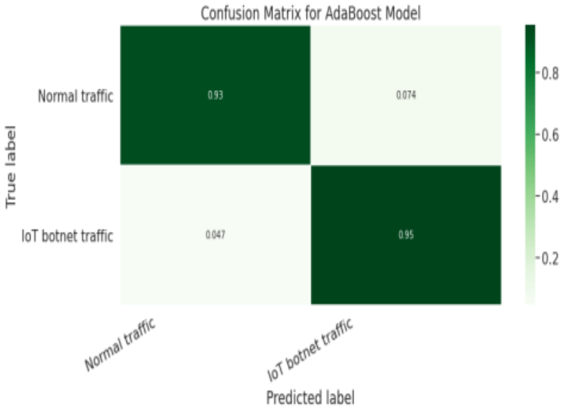


Figure 15: the Confusion matrix plotted by the AdaBoost

kNN classifier when executed, the performance is represented in table 4 and also visualized in figure 16, 17 and 18.

Table4

Classifier	Error	F1 score	Accuracy	Recall	Precision
kNN	0.0872	0.9318	0.9127	0.9361	0.9277

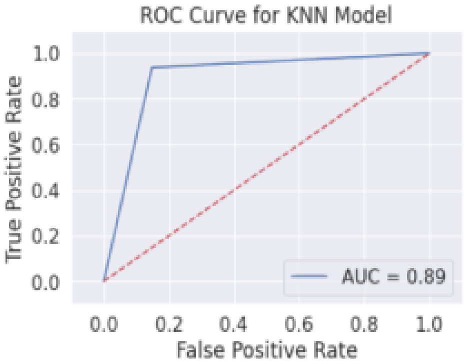


Figure 16: The ROC Curve generated by kNN

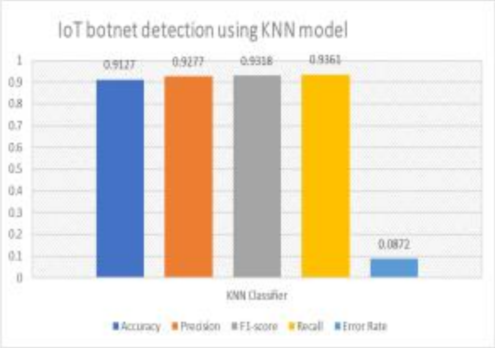


Figure 17: The evaluation graph generated by kNN model

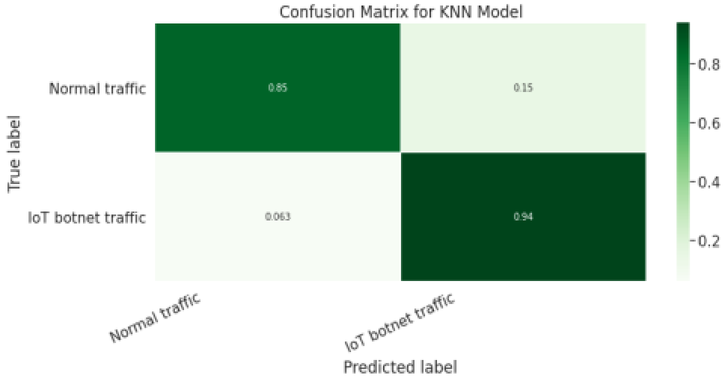


Figure 18: The Confusion matrix plotted by the kNN

The complete model implemented was dense neural network and the results are shown in table 5 and plot is visualized in figure 19.

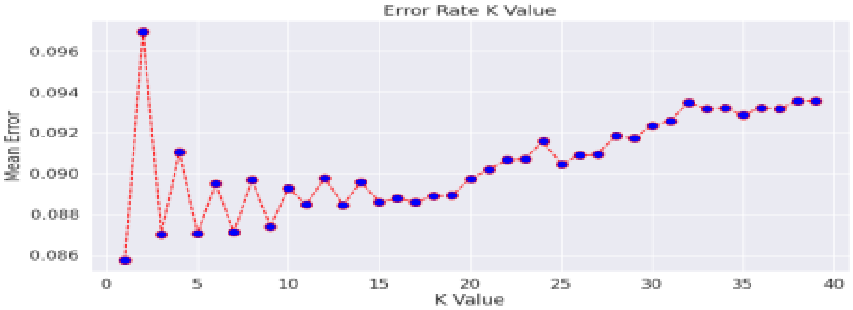


Figure 19: The comparative analysis of error rate to the k-value

Table 5

Model	Number of epochs	Validation loss	Training Accuracy	Testing accuracy	Testing loss
Dense neural network	50	0.1758	0.9114	0.9080-0.9170	0.1758

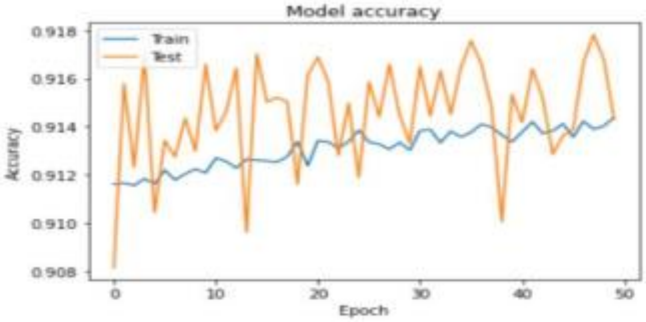
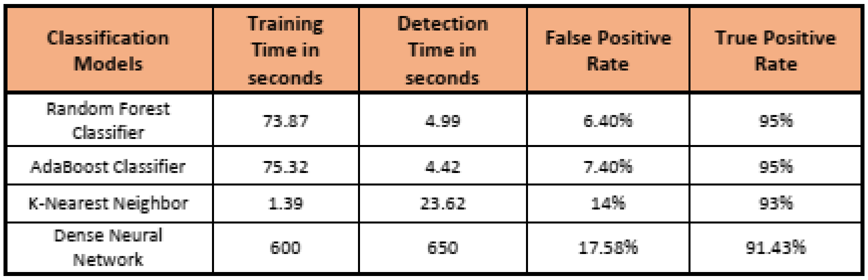


Figure 20: The accuracy plot generated by Dense Neural Networks

5. DISCUSSION

Our proposed model verified the role of identifying the IoT botnet with the help of network traffic information. A pragmatic analysis was done on the dataset using AdaBoost, KNN, random forest, and dense neural networks. After analyzing the considered metrics, the results exposed that the random classifier has potentially outperformed well than the other techniques of the machine and deep learning. To accomplish the task of feature selection, spearman's correlation technique was used to implement the dimensionality reduction and to condense the rate of false positive and detection time. It is observed that all the mentioned four techniques resulted in good accuracy and explicit performance by random forest with less detection time. It is also noticed from the dense network that, this model was over fitting for the selected data. The table 6 demonstrates the comparative report in overall among the work we have experimented.

Table 6 - Comparison Of Tables



Classification Models	Training Time in seconds	Detection Time in seconds	False Positive Rate	True Positive Rate
Random Forest Classifier	73.87	4.99	6.40%	95%
AdaBoost Classifier	75.32	4.42	7.40%	95%
K-Nearest Neighbor	1.39	23.62	14%	93%
Dense Neural Network	600	650	17.58%	91.43%

6. CONCLUSION

Throughout our research work, random forest classifier performance was assessed in the identification and detection of the IoT botnet. The training and the detection duration for other machine learning algorithms were also gaged. The time taken for random forest model to execute was 4.99 seconds and attained an accuracy of 94.47 percent with 96.28 percent precision. The count of computational resources used to instrument this work is low and thus an appropriate option in the IoT environment.

Limitations:

Though we have the ability to generate our own IoT dataset using an IoT network, we were unable to do so due to a few constraints. However, only a few open-source simulators support IoT networking. More research on a few simulators, such as Mininet, Cup carbon, and the IoT if y simulator, was directed to comprehend the functionality of IoT devices. Meanwhile, because the UNSW NB15 dataset was used, obtaining network details was impossible. In the future, open source simulators may assist us in extracting information from IoT network traffic as well as data to train models to successfully detect the IoT botnet.

REFERENCES

- S. Nomm and H. Bahsi, "Unsupervised Anomaly Based Botnet Detection in IoT Networks," in Proceedings - 17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018,Jan. 2019, pp. 1048–1053, doi: 10.1109/ICMLA.2018.00171.
- N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Data set for Network Intrusion Detection systems (UNSW-NB15 Network Data Set)." [Online]. Available: https://cve.mitre.org/.
- N. Moustafa and J. Slay, "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," Information Security Journal,vol. 25, no. 1–3, pp. 18–31, Apr. 2016, doi: 10.1080/19393555.2015.1125974.
- Gadde Ramesh and Dr. Suresh Pabboju, "BOTRANSACK to detect botnets using Machine learning" in Design Engineering (Toronto) @ 2021, ISSN: 0011-9342, Issue-9, pp: 10279-10292, http://thedesignengineering .com/index.php/DE/article/view/8145.
- W. Zong, Y. W. Chow, and W. Susilo, "A two-stage classifier approach for network intrusion detection,"in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligenceand Lecture Notes in Bioinformatics), 2018, vol. 11125 LNCS, pp. 329–340, doi: 10.1007/978-3-319-99807-7_20.
- N. Moustafa, J. Slay, and G. Creech, "Novel Geometric Area Analysis Technique for Anomaly Detection Using Trapezoidal Area Estimation on Large-Scale Networks," IEEE Transactions on Big Data, vol. 5, no.4, pp. 481–494, Jun. 2017, doi: 10.1109/tbdata.2017.2715166.
- N. Koroniotis, N. Moustafa, E. Sitnikova, and J. Slay, "Towards Developing Network forensic mechanism for Botnet Activities in the IoT based on Machine Learning Techniques".
- Gadde Ramesh and Dr. Suresh Pabboju, ―An overview on architecture of botnet and techniques towards the botnet detectionǁ in International Journal of Research and Applications (Impact Factor - 2.643) July – September © 2019 Transactions, eISSN: 2349-0020 &pISSN: 2394-4544 Vol-6, Issue-23, pp: 1307-1313, DOI: 10.17812/IJRA.6.23 (2)2019. , http://ijraonline.com/pdffiles/cimg144213_6 (23)1307-1313.pdf.
- R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning DDoS detection for consumer internet of things devices," in Proceedings - 2018 IEEE Symposium on Security and Privacy Workshops, SPW 2018,Aug. 2018, pp. 29–35, doi: 10.1109/SPW.2018.00013.
- K. Singh, S. C. Guntuku, A. Thakur, and C. Hota, "Big Data Analytics framework for Peer-to-Peer Botnet detection using Random Forests," Information Sciences, vol. 278, pp. 488–497, Sep. 2014, doi: 10.1016/j.ins.2014.03.066.

❖ H. T. Nguyen, Q. D. Ngo, D. H. Nguyen, and V. H. Le, “PSI-rooted subgraph: A novel feature for IoT botnet detection using classifier algorithms,” ICT Express, Jun. 2020, doi: 10.1016/j.icte.2019.12.001.

❖ L. Buczak and E. Guven, “A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection,” IEEE Communications Surveys and Tutorials, vol. 18, no. 2, pp. 1153–1176, Apr. 2016, doi: 10.1109/COMST.2015.2494502.

❖ L. Chen, Y. Zhang, Q. Zhao, G. Geng, and Z. Yan, “Detection of DNS DDoS Attacks with Random Forest
Algorithm on Spark,” in Procedia Computer Science, 2018, vol. 134, pp. 310–315, doi: 10.1016/j.procs.2018.07.177.

❖ C. Yin, Y. Zhu, J. Fei, and X. He, “A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks,” IEEE Access, vol. 5, pp. 21954–21961, Oct. 2017, doi: 10.1109/ACCESS.2017.2762418.

❖ B. Ingre and A. Yadav, “Performance Analysis of NSL-KDD dataset using ANN,” 2015. D. Breitenbacher and Y. Elovici, “N-BaIoT-Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders.” [Online]. Available: http://archive.ics.uci.edu/ml/datasets/detec-. F. Palmieri, U. Fiore, and A. Castiglione, “A distributed approach to network anomaly detection based on independent component analysis,” Concurrency Computation Practice and Experience, vol. 26, no. 5, pp. 1113–1129, Apr. 2014, doi: 10.1002/cpe.3061.

❖ Gadde Ramesh, and Dr. Suresh Pabboju,”A Host–Based P2P Host Identification Approach with Flow–Based in Detection of P2P Bots” in Tianjin DaxueXuebao (ZiranKexueyu Gongcheng Jishu Ban) Journal of Tianjin University Science and Technology @ 2022, ISSN: 0493-2137, Vol: 55, Issue: 01:2022, pp: 1-17, https://tianjindaxuexuebao.com/details.php?id=DOI:10.17605/OSF.IO/UE28D.

❖ "HuffPost is now a part of Verizon Media", Huffpost.com, 2020. [Online]. Available: https://www.huffpost.com/entry/cisco-enterprises-are-leading-the-internet-of-things. [Accessed: 16- Aug- 2020]

❖ M. Mayo, "The Data Science Process, Rediscovered - KDnuggets", KDnuggets, 2020. [Online]. Available: https://www.kdnuggets.com/2016/03/data-science-process-rediscovered.html. [Accessed: 16- Aug-2020]

❖ "DDoS Hackers Using IoT Devices to Launch Attacks - Corero", Corero, 2020. [Online]. Available: https://www.corero.com/blog/ddos-hackers-using-iot-devices-to-launch-attacks/. [Accessed: 16- Aug- 2020]

❖ T. Sureda Riera, J. Bermejo Higuera, J. Bermejo Higuera, J. Martínez Herraiz and J. Sicilia Montalvo, "Prevention and Fighting against Web Attacks through Anomaly Detection Technology. A Systematic Review", Sustainability, vol. 12, no. 12, p. 4945, 2020.

❖ D. Burgio, "Reduction of False Positives in Intrusion Detection Based on Extreme Learning Machine with
Situation Awareness", NSUWorks, 2020. [Online]. Available: https://nsuworks.nova.edu/gscis_etd/1093/. [Accessed: 16- Aug- 2020].