IoT Bonet and Network Intrusion Detection using Dimensionality Reduction and Supervised Machine Learning

Madhuri Gurunathrao Desai
College of Computing and Software Engineering
Kennesaw State University
Atlanta, USA
mdesai9@students.kennesaw.edu

Yong Shi
College of Computing and Software Engineering
Kennesaw State University
Atlanta, USA
yshi5@kennesaw.edu

Kun Suo
College of Computing and Software Engineering
Kennesaw State University
Atlanta, USA
ksuo@kennesaw.edu

Abstract — **In a present-day world there are different types of attacks being launched on computing devices. World is experiencing more and more cyber-attacks and types of attacks are also increasing. For example, an IoT device in a home network can act as a botnet attacking other devices or there could be man in the middle attack. As time goes by more and more devices are being connected within any given network. All these devices will be vulnerable to attacks if any one of the devices is compromised within the network. This complicates Intrusion Detection in any given network. Manual detection and intervention is nearly impossible. Hence it is quintessential to detect different types of attacks with more confidence with less computation complexity and time. A lot of research work has already been done in this area where the attacks have been studied separately. In this paper we focus on detecting intrusions including IoT botnet attacks and other types of network attacks. To achieve this, we build a multiclass classification using supervised learning models along with the dimensionality reduction technique. Numerous studies on ML-based IDS have been using KDD or the upgraded versions of KDD dataset. In this study we used a new dataset, IoT network Intrusion Detection dataset.**

Keywords—Supervised Learning, Multiclass Classification, Dimensionality Reduction, types of attacks, IoT network Intrusion dataset

I. INTRODUCTION

With the advancement in technology, more and more devices are being connected to the internet, with the majority of them being IoT devices. Internet of Things (IoT) is one of the emerging technologies used in various fields. These IoT devices are vulnerable to attacks and can be weaponized as botnets. One of such famous botnets is Mirai which is used to attack on all devices connected in the network where the affected device is in. Networks are also susceptible to other attacks such as Man in the Middle and DDoS attacks. Securing networks from intrusions or attacks is becoming harder as the network technologies are rapidly growing.

It is convenient when smart devices like TVs or watches are connected to the Internet and receive/send data. However, before reaching their final destinations data pass through all four steps of TCP/IP model, and they are exposed to all well-known risks. So, in the network with the IoT devices there is not only a single Mirai attack but also many other security threats involved, and they must be detected, managed and handled by IT experts.

In this paper we apply the standard workflow which consists of data preparation, feature reduction, model building, model training and validation, on a new dataset "IoT network Intrusion Detection dataset". For feature reduction we use PCA and the reduced features allow us to reduce the computational speed with very little compromise in model accuracy. For the model building and training we use Decision Tree, Random Forest and SVM which are supervised learning algorithms to detect the attacks. Here we use multiclass classification to detect the attacks and categorize the attacks into four different types: Benign, Mirai, Man-in-the middle and Scan attacks. This categorization helps the security analyst take the required actions on these attacks. We also use different metrics such as accuracy, confusion matrix, recall and precision to validate and check the results of the models being built. Main contribution of this study is using a single model for detecting different attacks with special focus on IoT botnet attacks. Another highlight of this study is reducing the features which helps in turn reduce the computational complexity while training the data set.

Section II discusses different types of attacks and Machine Learning approaches applied to Intrusion Detection. Section III covers details on the dataset used in this study. Supervised learning methods application and initial results of their application along with some detailed methods are also covered. Section IV details feature extraction, implementation of dimensionality reduction and other detailed results. Section V concludes with interpretation of the results.

II. RELATED WORK

In this section we discuss the background of the researches in Intrusion Detection. Extensive research work has been done in detecting anomalies in network. Signature-based Intrusion Detection has been widely popular and has been deployed in many systems. It has been extensively studied as well. However, given dynamic nature of systems new signatures keep on adding to the list. Hence, there is need for dynamic systems which can detect the attacks without dependency on its signature.

There are many supervised and unsupervised Machine Learning algorithm used in detecting these anomalies. Feedforward Neural Network has been used in [8] to detect anomalies like DoS and Port scanning. This does not include IoT botnets. In some studies K-nearest neighbor has shown to yield better accuracy for detecting intrusions [7] but accuracy takes a hit as the data size increases. PCA has been used in combination with SoftMax Regression to do multi class classification for different types of attacks [2], but it does not contain botnet attacks and it uses KDD cup 99 which is not originally derived from IoT networks.

978-1-7281-9656-5/20/$31.00 ©2020 IEEE

There are some papers which studied multiclass classification with different types of attacks. In [2] the authors classify attacks into 4 categories (DOS, U2R, R2L and Probing), and they have used only one method SVM for the Intrusion Detection using KDD cup dataset.

In recent years many papers have been proposed for IoT botnet Intrusion Detection, but the research does not extend to cover other type of attacks along with IoT botnets. Decision tree has been used along with feature selection for detecting botnet in [1], where the authors classified the attacks into either benign or botnet attack. They have used decision tree and F-score to build and train their model. This yields better accuracy for classification with concentration only detecting botnets without considering other types of attacks.

Similarly, different supervised learning algorithms like Support Vector Machine (SVM), KNN, Neural Networks, Decision Tree and Random Forest have been applied successfully on the dataset which was generated in [5]. However, this only provides generic classification between normal and attack traffics. [6] implements IoT and IoBT malware detection with help of Deep Eigenspace and sequences from devices and OpCodes sequence used as a feature for classification. Deep auto encoders have also been revised to specifically detect IoT Botnet Attacks [10].

In [4] the authors used kyoto 2006+ dataset, and using Random Forest Classifier they have first refined the original 3 classes (i.e., normal, known attack unknown attack) into 6 classes (i.e., normal, unknown, shellcode, IDS+shellcode, malware, IDS). However, performance varies depending upon the dataset.

All these papers have addressed the issue related to either IoT botnet intrusion or network Intrusion Detection. But when there is information being exchanged in the network with the connected devices, there will be not only a botnet attack but also many other different network intrusion (like Scan, protocol specific attacks etc.) Given the ability to differentiate the type of attacks, the system administrators can quickly isolate the devices and take necessary actions. Hence it is necessary to detect all kinds of attacks and one of the possible ways is through multi class classification.

III. Our Appoach

A. Dataset

The KDD CUP 1999 dataset (KDD) has been used in most of the research work involving Intrusion Detection. This dataset is developed by Defense Advanced Research Projects Agency (DARPA) and is the most used dataset for IDS evaluation [3]. KDD classifies attacks into four categories, including DoS, User to Root (U2R), Remote to Local (R2L) and Probing. KDD was generated by injecting these kinds of attacks into each category. Most of these studies perform binary classification that classifies the entire KDD into attack and benign. They also carry out multiclass classification to classify the KDD into the four categories, but this dataset is not derived from IoT networks [1].

In this study we use a new dataset "IoT network Intrusion Detection dataset" [9]. As per our knowledge this dataset has not used in any Intrusion Detection papers so far. This dataset is downloaded from HCRL (Hacking and countermeasure research lab) which was added on September 20, 2019. To download the dataset, we had submitted the application form with all the required details like purpose for download, email id etc. This dataset is created with various types of network attacks involving different types of devices in the network. Two smart home devices (SKT NUGU known as NU 100 and EZVIZ Wi-Fi Camera known as C2C Mini O Plus 1080P) were used, and there are also laptops and smartphones which are connected to the same wireless network. This dataset consists of 36 raw network packet files (pcap) at different time points. All packets except Mirai Botnet attacks are captured while simulating attacks using the Nmap tool. In the case of Mirai Botnet category, the attack packets were generated on a laptop and then manipulated to make it appear as if it originated from the IoT device. Mirai is very well-known malware which can make IoT devices as botnets. IoT network Intrusion Detection dataset is categorized into 3 different types of attacks: Mitm (Man in the middle), Mirai and Scan. The total data present in each of the attacks including benign are as follows: Benign:137,396; Mirai: 2,202,225;Man-in-the-middle(Mitm):194,184; Scan:310,480

B. Algorithm Description

We use three types of supervised learning algorithms: Decision Tree (DT), Random Forest classifier (RFC), and Support Vector Machines (SVM). In these methods the dataset is split into training (80%) and testing (20%). In these algorithms, the labels are created for multiclass classification to classify the data into Benign, Mirai, Mitm and Scan. As discussed in the Feature Extraction (FE) which is detailed in the experiment section, we have considered 115 features for training all the different models. Each model was trained on different sets of data, starting from equal distribution of data in all the classes then, training on benign skewed data and attacks skewed data later. Different models are trained on the chosen data to get the training and testing accuracy curve along with the estimation of training time.

Following is the Pseudocode of the algorithm:

```
Data Preparation
{
    For all pcap files
    {
        Extract features;
        Save the features into respective CSV file;
    }
}

Model Training and Validation
{
     Read features from CSV file:
     Create class for each category of attack and benign:
     IF PCA enabled
     {
          Reduce the features from 115 features to x Dimensions
     }
     Split data in train and test
     Build the model
     Train the model / fit the data
    Test the model on training data and record training metrics
   Test the model on testing data and record the testing metrics
        IF graph enabled
        {
                Plot the learning curve and timing curve
        }
}
```

1) *Decision Tree (DT)*

DT builds classification models in the form of tree structure. It breaks down a data set into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. Here the flow of the DT classifier is that the data which is chosen is divided into training and testing data. Later the DT model is trained on different depths as 3,5,7,9,11 and we have used 'Entropy' as the criteria. Depth is one of the main parameters of DT which affects the accuracy of the model to great extent. If the sample is homogeneous, Entropy is 0; else if the sample is equally divided entropy is maximum 1. Here is the formula

$$\sum_{i=1}^{n} Pi * log(p_i) \quad (1)$$

The following learning graph Fig 1 explains the training and testing accuracy of DT models for different decision tree depths. 29K records of each type – Mirai, Mitm, Scan and Benign are considered for training.

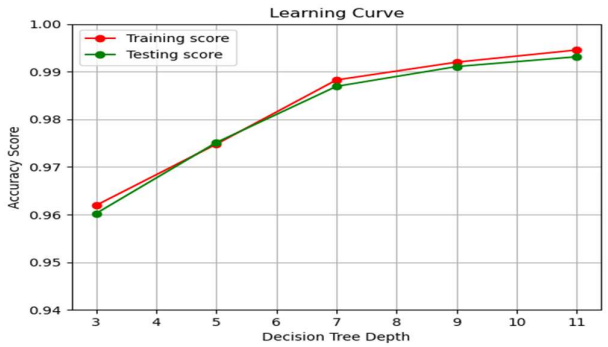

Fig 1: Learning curve graph of DT describing the accuracy at different depths (equal distribution of data 116000 with 115 features)

From Fig 1 we can see that the accuracy increases with the increase in the number of depths. Here, we consider *depth = 11* for our further calculations and analysis as the accuracy obtained is highest as

Training Accuracy: 99.45
Testing Accuracy: 99.31

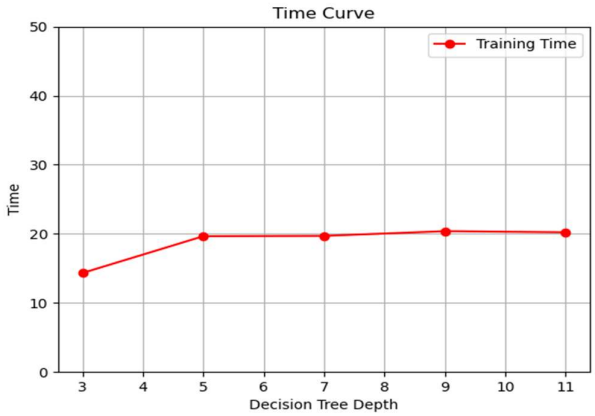

Fig 2: Time curve graph of DT describing the time taken by the model to train the data at different depths (equal distribution of data 116000 each with 115 features)

Fig 2 shows the training time taken for different depths. we can see that time taken by the DT model on all the depths is less than *25 seconds*. Since we are interested in the *depth =11* the time taken by model to train at this depth is *16 seconds.*

2) *Random Forest Claasifier (RFC)*

RFC adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. Here in RFC model the importance is given to the "N-estimators". N-estimator fits a given number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. We have trained model using different estimator values from 50, 80, 100, 130 to 150. Along with that we have also used other parameters such as class_weight = 'balanced' and random_state = 5 and analyzed the training and testing accuracy with the learning curve graph. 29K records of each type – Mirai, Mitm, Scan and Benign are considered for training.

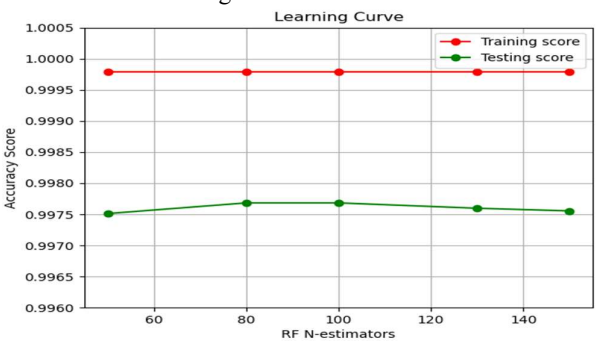

Fig 3: Learning curve graph RFC describing the accuracy at different N-estimators (equal distribution of data 116000 each with 115 features)

From fig 3 we can see that the accuracy is almost same for all the n-estimators value. Here we consider *estimator = 50* for all our further calculations as it has the low estimator value and better accuracy as

Training Accuracy: 99.97
Testing Accuracy: 99.75

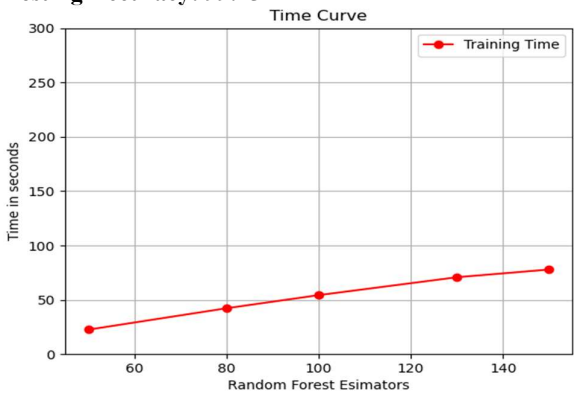

Fig 4: Time curve graph of RFC describing the time taken by the model to train the data at different estimators (equal distribution of data 116000 each with 115 features)

From fig 4 we can see that time taken by the RFC model on all the estimators is less than *1 minute*. Also, as we can see that as the number of estimators increases the time also increases. Since we are interested in the *estimator = 50* the time taken by model to train for this estimator is 24 *seconds.*

3) Support Vector Machine (SVM)

Support Vector Machine (SVM) is one of the most used supervised algorithms because of its ability to separate data of different classes from each other. This algorithm uses the classification algorithm for two group classification but, here in our case we have used it as a multiclass classification. The same splitting of the data is done as the previous methods.

C parameter in SVM is Penalty parameter of the error term considered as degree of optimization the SVM must meet. For greater values of C, there is impossible that SVM optimizer can misclassify any single point. Here we wanted to try out the values of 'C' 0.1, 0.5, 0.8, 1.0. with other parameters kernel = 'rbf', degree = 3, gamma = 'auto'. However, the SVM model takes too long time to fit the 116000 records with 115 features. Also, it gives lower testing accuracy. Here is an example of the run where we executed the SVM for 64K records of data and all the features.

Training Accuracy Score: 0.9995
Testing Accuracy score: 0.58593
Execution time: 2:30:7

Given the range of values differ for different value, it seems that SVM may not be the right model to use for this network dataset. Going forward in this paper SVM will not be considered and further experimentation is carried out on Decision Tree and RFC.

4) Principal Component Analysis (PCA)

PCA is often used as a dimensionality-reduction technique which explains the variance - covariance structure of a set of variables through linear combinations. The purpose of using the dimensionality reduction technique is to reduce the training time of the model. For the original data the time taken to train the model is very long. So one of the is by reducing the dimensions, while maintaining the accuracy as much as possible.

We applied PCA on dataset and later trained Random Forest Classifier on reduced data with parameter n-estimator as 50, along with other parameters mentioned in the RFC method. We tested the PCA on different dimension 3, 5, 10, 15, 20, 50, 100 on the same 116K dataset. Here is the formula.

$$cov(x,y) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) \quad (2)$$

Below is the training and testing accuracy learning curve graph and training time graph of RFC which uses PCA for reducing data. Here the data is equally distributed between all the classes, 29K records of each type of packets – Benign, Mirai, Mimt, Scan are considered for training.

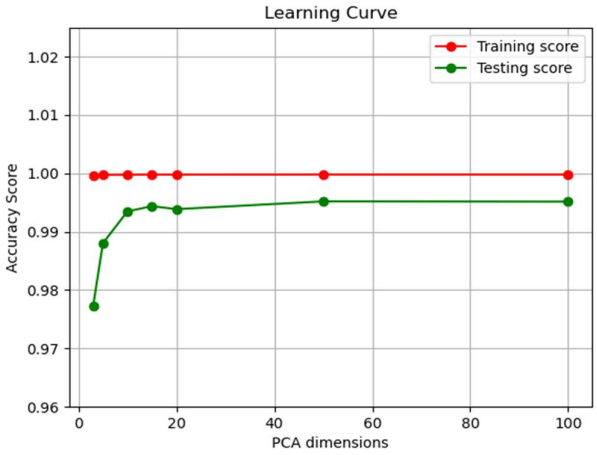

Fig 5: Accuracy Learning curve graph using different PCA dimensions run on RFC model (equal distribution of data 116000 each with 115 features)

From fig 5 we can see that the accuracy of the training and testing scores are maintained even when the dimensions are reduced. We see that accuracy is increasing with increase in dimensions. With 3 dimension it has low accuracy but with the dimension 10, 15, 20 the accuracy has increased and is maintained.

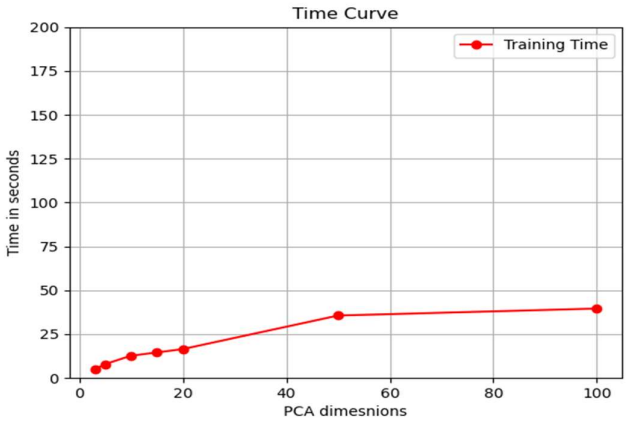

Fig 6: Training time curve graph using different PCA dimensions run on RFC model (equal distribution of data 116000 each with 115 features)

From fig 6 we can see that with the decrease in dimensions the training time also has reduced. We compare the accuracy and time using the table below.

Dimension	**Train Accuracy**	**Test Accuracy**	**Train time**
10	99.97	99.80	10 sec
15	99.97	99.43	10sec
20	99.97	99.55	8sec

Table 1: Accuracy and training time using different PCA dimensions run on RFC model

From table 1 we can observe that all three dimensions have higher accuracy with low execution time. According to the above table, dimension =10 seems to be the better choice for PCA. However, this is only looking at one metric. We tested with confusion metric and precision and recall for each category to get more insight.

Dim	Class	Confusion Metrics		Precision	Recall
		Correct	Wrong		
10	Benign	5868	42	0.992	0.991
	Mirai	5654	46	0.995	0.993
	Mitm	5836	36	0.993	0.994
	Scan	5798	28	0.992	0.995
15	Benign	5894	36	0.993	0.993
	Mirai	5660	40	0.997	0.993
	Mitm	5840	32	0.994	0.994
	Scan	5803	23	0.996	0.994

Table 2: Confusion metric, Precision and Recall metrics using different PCA dimensions run on RFC model

From table 3 we can see that dim=15 has high precision and low false positive data compared to dim=10 . We have also tried on different combination of dataset set which is shown in the experiment section.

IV. EXPERIMENTS

A. Feature Extraction

The downloaded dataset ("IoT network Intrusion Detection dataset") was in pcap (wire shark) format which are considered as raw data files, and these had to be converted to (.csv) format to make them readable. For this we leveraged the feature extraction framework [11] and wrote code to extract the data. We have 115 features from each record. As discussed above in the dataset part, we know that the dataset consists of both IoT network data and different kinds of attacks data. First, we analyzed what the IP-addresses of each of the target IoT devices are, in order to know the amount of incoming and outgoing traffic from each of these devices. Next, with the help of [1] we were able to make different analysis, for example, we apply the method where recent history of the stream is captured in various time statistics such as L5 (100ms), L3 (500ms), L1 (1.5sec), L0.1 (10sec) and L0.01 (1min). The entire traffic summary is divided into five feature categories: Source-IP(H), Source-MAC&IP(MI), channel (HH), channel-jitter (HH_Jit) and socket (HpHp). The detailed information is shown in table 4. Here is an example of how the features are extracted from the packets : Host IP-100ms- weight(Pkt Count)" corresponds to the feature that is computed by the packet count of the Source-IP category at interval 100ms. This is carried out for all the 5-time intervals to get different stats.

Feature Categories	Description	Extracted features Stat	No: of features from each category
Source-IP(H)	Stats summarizing the recent traffic from pkt Source (IP)	**Weight** (pkt count), **Mean**, **Variance**	**15**
Source-MAC&IP(MI)	Stats summarizing the recent traffic from pkt Source (IP + MAC)	**Weight** (pkt count), **Mean, Variance**	**15**
Channel (HH)	Stats summarizing the recent traffic going from this pkt source (IP) to pkt destination.	**Weight** (pkt count), **Mean, Std, Magnitude**, **Radius** (variances), **co-variance, Pcc**	**35**
Channel-jitter (HH_Jit	Stats summarizing the jitter of the traffic going from this pkt source (IP) to pkt destination.	**Weight** (pkt count), **Mean**, **Variance**	**15**
Socket (HpHp)	Stats summarizing the recent traffic going from this pkt source+port (IP) to the pkt destination source+port	**Weight** (pkt count), **Mean, Std, Magnitude, Radius** (variances), **co-variance, Pcc**	**35**

Table 3: Attribute and Features information

B. Model Evaluation & Results

In this section we discuss how the Random Forest classifier (RFC) is better when compared to the other methods in doing the multiclass classification, i.e., classifying the given data into Benign, Mirai, Man In the Middle and Scan attacks. We have evaluated the performance of the PCA using confusion matrix, precision, and recall. We have tested the working of the RFC model on PCA with different combination of the dataset - benign skewed, attacks skewed, equal distribution and complete 2 million plus dataset.

Parameters deduced in previous section are used for Decision Tree and Random Forest for following experiments which yield higher accuracy and lower training time.

	Method	Train Acc	Test Acc	Est. Time
Skewed to Benign	DT	99.54	99.36	30 sec
	RFC	99.98	99.71	30 sec
Skewed to Attack	DT	99.56	99.35	17 sec
	RFC	99.54	99.36	30 sec
All Data	DT	97.48	97.43	23 min
	RFC	99.97	99.76	35 min

Table 4: Accuracy and training time of Benign Skewed, Attack Skewed (Unbalanced Distribution) and All data Distribution

1) All data

Firstly, we ran these above models on all the data which contains 2844285 records with 115 features. We can see from table 4 that DT estimation time is less when compared to the RFC but the accuracy obtained by the RFC model is higher than the DT model. To get the better clarity on this we conducted some other different experiments on the data.

2) Skewed to Benign (unbalanced data)

In this experiment we considered the dataset which has more benign data compared to attack dataset. In other words, Benign (50000 data, 115 features) and all other attacks (Mirai, Mitm, Scan) (29000 data, 115 features). From table 4 we can see that both accuracy and training time are same. So, In order to get the better clarity on which model is better we also included different matrices, Confusion Matrix, Precision Recall.

Method	Class	Confusion Metrix		Precision	Recall
		Correct	Wrong		
DT	Benign	9997	31	0.993	0.996
	Mirai	5820	42	0.994	0.992
	Mitm	5751	43	0.992	0.992
	Scan	5733	58	0.994	0.989
RFC	Benign	10012	16	0.999	0.998
	Mirai	5845	17	0.999	0.997
	Mitm	5773	21	0.998	0.996
	Scan	5768	23	0.997	0.996

Table 5: Precision, Recall, Confusion Matrix of Benign Skewed (unbalanced data) (Benign-50000 attacks-29000 each for other attacks)

3) Skewed to Attack (Unbalanced Data)

In this experiment we considered the dataset which has more Attack data when compared to Benign data. In other words, Benign (10000 data, 115 features) and all other attacks (Mirai, Mitm, Scan) (29000 data, 115 features). We evaluated using confusion matrix, precision and recall matrices on both DT and RFC models.

Method	Class	Confusion Metrix		Precision	Recall
		Correct	Wrong		
DT	Benign	2049	17	0.989	0.991
	Mirai	5788	35	0.996	0.993
	Mitm	5762	24	0.990	0.995
	Scan	5756	50	0.995	0.991
RFC	Benign	2057	9	0.997	0.995
	Mirai	5814	9	0.998	0.997
	Mitm	5768	18	0.998	0.996
	Scan	5791	15	0.996	0.997

Table 6: Precision, Recall, Confusion Matrix of Attacks Skewed (unbalanced data) (Benign-10000 attacks-29000)

From table 5 & 6 we can observe that the number of False-Positives predicted by the DT model is more when compared to the RFC. Also, looking at the precision and recall metrics we see that the percentage of total relevant results correctly classified by RFC algorithm is more when compared to DT.

4) PCA – RFC on complete datset

In this section PCA is applied on the entire dataset and later RFC is used for classification.

Dataset/Dim	Train Acc	Test Acc	Train Time
Unreduced (2844285 data, 115 features)	99.97	99.76	35 mins
Reduced (2844285 data, 10 features)	99.97	98.41	7 min 24 sec
Reduced (2844285 data, 15 features)	99.97	99.07	6 min 33 sec

Table 7: comparison table- training and test accuracy with train time on complete dataset with 115 feature and reduced features with PCA (10, 15)

The above table explains about the comparison of training and testing accuracy along with the execution time between the dataset with 115 features and dataset with reduced features to 10 and 15 using PCA. We can say that using PCA before RFC has reduced the training time and as well maintained the accuracy high.

	Dataset/Dim	Train Acc	Test Acc	Est. Time
Skewed to Benign	167396 data 10 features	99.97	99.51	14 sec
	167396 data 15 features	99.97	99.89	16sec
Skewed to Attack	145000 data 10 features	99.97	99.11	14 sec
	145000 data 15 features	99.97	99.95	16 sec

Table 8: comparison table- training and test accuracy with train time on Benign skewed dataset and Attack Skewed data with PCA reduced dimensions

5) PCA-RFC on Benign skewed dataset

This experiment is on how the PCA with reduced 10, 15 features works with Benign skewed dataset. Here we have considerd benign (167396 data, 10 and 15 feature) other attacks(30000 data, 10 and 15 feature)

From table 8 it seems that dimension = 10 and dimension =15 both have almost similar accuracy and training time. But this is not enough to say dimension =10 is better choice as it has. To say which of two dimensions is better from all angle we have the below tables

Dim	Class	Confusion Metrix		Precision	Recall
		Correct	Wrong		
10	Benign	9994	36	0.989	0.991
	Mirai	5830	32	0.996	0.993
	Mitm	5763	31	0.990	0.995
	Scan	5737	54	0.995	0.991
15	Benign	9996	34	0.997	0.995
	Mirai	5831	31	0.998	0.997
	Mitm	5763	31	0.998	0.996
	Scan	5742	49	0.996	0.997

Table 9: Benign skewed confusion matric and precision and recall based on each class with 10 and 15 dimensions

6) PCA-RFC on Attack skewed dataset

This experiment is on how the PCA with reduced 10, 15 features works with Attack skewed dataset. Here we have considerd benign (10000 data, 10 and 15 feature) other attacks (135000 data, 10 and 15 feature).

Dim	Class	Confusion Metrix		Precision	Recall
		Correct	Wrong		
10	Benign	2030	36	0.995	0.985
	Mirai	5800	23	0.996	0.995
	Mitm	5760	26	0.995	0.995
	Scan	5768	38	0.992	0.991
15	Benign	2037	36	0.995	0.987
	Mirai	5800	23	0.996	0.995
	Mitm	5762	24	0.995	0.995
	Scan	5778	28	0.995	0.991

Table 10: Attack skewed confusion matric and precision and recall based on each class with 10 and 15 dimensions

From the above table 9 and 10 we can say that the number of false positives detected are more in dimension =10 than dimension =15 and also the precision is high on dimension =15. So PCA -RFC with dimension =15 is better choice with above mentioned condition

V. CONCLUSION

Our study has revealed that it is very important to have a model that is able to detect different types of attacks with special focus on IoT botnet attacks. This paper has demonstrated that supervised learning method of Random Forest is able to identify and classify benign and different types of attacks (Mirai, Mitm, Scan) with highest testing accuracy of 99.97%. Multiclass classifier model built using Random Forest has shown better results compared to other models like Decision Tree and SVM. The model has been tested with different ratio of attack to normal data and has consistently shown great results with very less false positive. Also, our research has shown that Principal Component Analysis does better job of reducing the features with little to no compromise in the quality of detection rate and greatly reducing the computational resources required for building the model.

REFERENCES

[1] Hayretdin Bahsi, Sven N˜omm and Fabio Benedetto La Torre, "Dimensionality Reduction for Machine Learning Based IoT Botnet Detection", 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV) Singapore, November 18-21, 2018.

[2] S. Zhao, W. Li, T. Zia and A. Y. Zomaya, "A Dimension Reduction Model and Classifier for Anomaly-Based Intrusion Detection in Internet of Things," 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech), Orlando, FL, 2017, pp. 836-843, doi: 10.1109/DASC-PICom-DataCom-CyberSciTec.2017.141.

[3] "Cup, K. D. D. "Dataset." ," http://kdd. ics. uci. edu/databases/kddcup99/kddcup99. html , 1999. [Online]. Available: http://kdd. ics. uci. edu/databases/kddcup99/kddcup99.

[4] K. Park, Y. Song and Y. Cheong, "Classification of Attack Types for Intrusion Detection Systems Using a Machine Learning Algorithm," 2018 IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService), Bamberg, 2018, pp. 282-286, doi: 10.1109/BigDataService.2018.00050.

[5] R. Doshi, N. Apthorpe and N. Feamster, "Machine Learning DDoS Detection for Consumer Internet of Things Devices," *2018 IEEE Security and Privacy Workshops (SPW)*, San Francisco, CA, 2018, pp. 29-35, doi: 10.1109/SPW.2018.00013.

[6] A. Azmoodeh, A. Dehghantanha and K. R. Choo, "Robust Malware Detection for Internet of (Battlefield) Things Devices Using Deep Eigenspace Learning," in IEEE Transactions on Sustainable Computing, vol. 4, no. 1, pp. 88-95, 1 Jan.-March 2019, doi: 10.1109/TSUSC.2018.2809665.

[7] S. Malhotra, V. Bali and K. K. Paliwal, "Genetic programming and K-nearest neighbour classifier based Intrusion Detection model," 2017 7th International Conference on Cloud Computing, Data Science & Engineering - Confluence, Noida, pp. 42-46, 2017.

[8] S. B. Wankhede, "Anomaly Detection using Machine Learning Techniques," *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*, Bombay, India, 2019, pp. 1-3, doi: 10.1109/I2CT45611.2019.9033532.

[9] " IOT Network Intrusion Dataset", "http://ocslab.hksecurity.net/Datasets/iot-network-intrusion-dataset",2019.[Online].Avaliable:http://ocslab.hksecurity.net/Datasets/iot-network-intrusion-dataset. html [Accessed: 14-March-2020].

[10] Y. Meidan et al., "N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders," in IEEE Pervasive Computing, vol. 17, no. 3, pp. 12-22, Jul.-Sep. 2018, doi: 10.1109/MPRV.2018.03367731.

[11] Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai, " Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection", arXiv:1802.09089 [cs], Feb.2018.

[12] Zoran Petar Cekerevac, Zdenek Dvorak, L. Prigoda, Petar Čekerevac "INTERNET OF THINGS AND THE MAN-IN-THE-MIDDLE ATTACKS – SECURITY AND ECONOMIC RISKS" 2017.

[13] A. A. Korba, M. Nafaa and Y. Ghamri-Doudane, "Anomaly-based Intrusion Detection system for ad hoc networks," *2016 7th International Conference on the Network of the Future (NOF)*, Buzios, 2016, pp. 1-3, doi: 10.1109/NOF.2016.7810132.

[14] K. Park, Y. Song and Y. Cheong, "Classification of Attack Types for Intrusion Detection Systems Using a Machine Learning Algorithm," *2018 IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService)*, Bamberg, 2018, pp. 282-286, doi: 10.1109/BigDataService.2018.00050.

[15] M. O. Miah, S. Shahriar Khan, S. Shatabda and D. M. Farid, "Improving Detection Accuracy for Imbalanced Network Intrusion Classification using Cluster-based Under-sampling with Random Forests," 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), Dhaka, Bangladesh, 2019, pp. 1-5, doi: 10.1109/ICASERT.2019.8934495.

[16] A. Azmoodeh, A. Dehghantanha and K. R. Choo, "Robust Malware Detection for Internet of (Battlefield) Things Devices Using Deep Eigenspace Learning," in IEEE Transactions on Sustainable Computing, vol. 4, no. 1, pp. 88-95, 1 Jan.-March 2019, doi: 10.1109/TSUSC.2018.2809665.