Intrusion Detection System (IDS) Using Machine Learning

Abdulmannan Ajabnoor
Elect. and Comp. Eng. Depart.
Georgia Institute of Technology
Atlanta, United States
aajabnoor3@gatech.edu

Mohammad Ahmad
Elect. and Comp. Eng. Depart.
Georgia Institute of Technology
Atlanta, United States
mohdahmad@gatech.edu

Jiaying Li
School of Music
Georgia Institute of Technology
Atlanta, United States
jli3269@gatech.edu

***Abstract*— Detecting and preventing cyberattacks on the network is very crucial to maintain security and defending against any malicious attempt. This paper is mainly focusing on the detection of six different cyberattacks by using four different machine learning algorithms, which include k-nearest neighbors (KNN), random forest (RF), Extreme Gradient Boosting (XGBoost), and logistic regression (LR). These algorithms are trained and tested on the CICIDS 2017 dataset. Before evaluating the model using a jupyter notebook, three different unbalanced techniques are utilized, including Synthetic Minority Oversampling Technique (SMOTE), borderline-SMOTE, and Adaptive Synthetic (ADASYN). In addition, the ANOVA test is used for feature filtering, while Principal Component Analysis (PCA) is employed for feature extraction. An evaluation according to the F1 score is provided.**

Keywords— Machine learning, unbalanced techniques, features extraction, and Cyberattacks

I. INTRODUCTION

Today, computer networks are used everywhere and it is crucial in many different applications. Computer networks can connect two or more devices at the same time. In some applications, these devices may need to communicate with each other to increase the performance or efficiency of the system. For example, [1] used the network to enable communication between different vehicles to enhance safety and driving efficiency. If the attacker can compromise the network and able to control the vehicle, safety will be affected. As a result, cybersecurity is very important to ensure the integrity, confidentiality, and availability of the network. IDS is used to detect network cyberattacks. Then, prevention methods are utilized to defend the network against any malware attempts. This paper is only focusing on detecting cyberattacks.

IDS can be signature-based and anomaly-based [2] and [3]. Signature-based can not detect unknown attacks because it matches the network attack with a previously known attack. On the other hand, anomaly-based can detect new attacks because any deviation from the normal operation of the network will be considered an attack. However, it has a risk of false detection.

One of the methods used for detecting cyberattacks is by using machine learning. This method requires data to train, and test the model. To ensure better performance of the classification, majority and minority classes should have the same amount of data. Otherwise, unbalanced techniques should be used to balance the amount of data. The dataset used in this paper is CICIDS2017. [1] used the CICIDS2017 dataset. However, SMOTE was the only unbalanced technique that is utilized. One class in the dataset has more than 2 million samples and some have only 36 samples. This paper does not show the amount of data in each class after applying SMOTE. It is not practical to increase the number of samples from 36 to more than 2 million. [3] used CICIDS2017 but no imbalanced techniques were employed. The machine learning models were tested without any training on abnormal samples. One drawback of using only benign samples to train the model is the high risk of false detection. In this paper, three imbalanced techniques are employed, SMOTE, Borderline-SMOTE, and ADASYN. However, these techniques are not utilized to maintain an equal number of samples between majority and minority classes. It is used to increase the number of minority class regardless of the number of majority class samples because there is a massive difference between them (ex. 2,271,352 samples of majority class and 36 samples for minority class)

In machine learning problems, feature selection is always an important step for preprocessing. Feature selection can eliminate irrelevant and redundant features, thereby achieving the purpose of reducing the number of features, improving model accuracy, and reducing running time. On the other hand, by selecting the relevant features, the model can be simplified, and the overfitting risk can be reduced. In this paper, both feature filtering, (ANOVA test) and feature extraction (Principal Component Analysis), are used.

In this paper, the classifications of normal and different types of attacks are achieved in two stages. The first stage is to apply a binary classification between normal and attack data. If the data is classified as an attack in the first stage, the second stage will be applied. The second stage classifies the type of attack. Four different ML models are applied and compared to each other in terms of accuracy and F1 score.

The rest of this paper is as follows: section II shows the description of the CICIDS2017 dataset. Section III describes the preprocessing techniques used before training the machine learning models. Section IV shows the experiment. Section V

discusses the results and provides a comparison between different machine learning models. Section VI concludes the paper.

II. Dataset Description

This dataset was published in 2017 and recorded for 5 consecutive days. In addition, it includes 14 different attacks. Table I summarize the types of attack that are recorded each day [4]. Table II shows the total amount of data for each type of class (benign and 14 attacks). The total number of instances in the dataset is 2,830,743.

It can be noticed from Table II that there are massive differences in the number of instances for each attack. As a result, [5] merged a few minority classes into one class to form a new label. This was done to mitigate the issue of an unbalanced dataset. The merged classes have the same main classification of attacks. For example, Web Attack Brute Force, Web Attack Sql Injection, and Web Attack XSS are merged to form a new class of attack, which is Web Attack. Table III describes the new class name after merging minority classes, and the new number of instances. It can be seen that the updated data set has only 7 classes (benign and 6 attacks). This data set has 78 features.

III. Preprocessing Dataset

A. Dataset Cleaning

After analyzing the dataset, it has been found that some instances had some missing data. As a result, it has been decided to remove these data because there were only 2,867 missing data. Most of these data were in benign class. Originally, it has 2,273,097 instances. It is important to say that this data set has 307,078 duplicate instances, which are removed. The total samples after cleaning the dataset are 2,520,798. Table IV demonstrates the updated number of instances for each class.

TABLE I. Type of attack found each day

Day	Attacks
1	Benign (Normal)
2	Benign, FTP-Patator, and SSH-Patator
3	Benign, DoS GoldenEye, DoS Hulk, DoS Slowhttptest, DoS slowloris, Heartbleed
4	Benign, Infiltration, Web Attack Brute Force, Web Attack Sql Injection, and Web Attack XSS

TABLE II. Total amount of data for each type of class

Attacks	Number of Instances
Benign	2,273,097
FTP-Patator	7,938
SSH-Patator	5,897
DoS GoldenEye	10,293
DoS Hulk	231,073
DoS Slowhttptest	5,499
DoS slowloris	5,796
Heartbleed	11
Infiltration	36
Web Attack Brute Force	1,507
Web Attack Sql Injection	21
Web Attack XSS	652
DDoS	128,027
PortScan	158930
Bot	1966

TABLE III. New class label

Merged Classes	New Label	Number of Instances
Benign	Benign	2,273,097
Bot	Botnet Areas	1,966
FTP-Patator, and SSH-Patator	Brute Force	13,834
DDoS, DoS GoldenEye, DoS Hulk, DoS Slow-httptest, DoS slowloris, and Heartbleed	DoS	380,700
Infiltration	Infiltration	36
Port Scan	Port Scan	158,930
Web Attack Brute Force, Web Attack Sql Injection, and Web Attack XSS	Web Attack	2,180

B. Training and Test Data

The dataset has been split by 80% for training data and 20% for test data. After the model is trained, it is tested using the test data. Table V shows the number of instances for training, and test data.

C. Imbalanced Techniques

From Table IV and V, there is a massive difference in the number of training data between majority and minority classes. These differences may result in a low anomaly detection rate. As a result, there are different techniques for an imbalanced dataset including, under-sampling, oversampling, and hybrid sampling are used by researchers [6]. The under-sampling technique is used to delete data from the majority class (ex. Benign). On the other hand, oversampling is utilized to generate data in the minority class (ex. Infiltration). Lastly, hybrid sampling employs both undersampling and oversampling. In this paper, an oversampling technique including SMOTE, Borderline-SMOTE, and ADAYSN have been used.

SMOTE is KNN based [7]. After identifying the k-nearest neighbors for each instance, new data in the minority class is

TABLE IV. New instances after cleaning the dataset

Label	Number of Instances
Benign	2,095,057
Botnet Areas	1,948
Brute Force	9,150
DoS	321,770
Infiltration	36
Port Scan	90,694
Web Attack	2,143

TABLE V. Training, and test data

Total data	2,520,798
Training Data (80%)	2,016,638
Test Data (20%)	504,160

generated. The identifying process for each instance in the minority class is done randomly. Typically, after this process, the minority class should be equal to the majority class.

In some cases, the KNN for a minority sample is a majority class data. As a result, Borderline-SMOTE is used to detect the borderline samples and then generates minority instances that may have the chance of being classified as a majority class (ex. benign) [7]. This method is useful for making the classifier discriminate between observations in borderline.

ADASYN is KNN based similar to SMOTE. However, it generates synthetic data samples adaptively based on the minority data distribution [7].

Using python (Jupiter notebook), the three minority classes, Botnet Areas, BruteForce, Web Attack, and Infiltration have been oversampled to 5,000, 10,000, 5000, and 500 samples, respectively. The aim of using these three oversampling techniques is to increase the number of minority class samples. The goal is not to maintain an equal number between majority and minority classes due to the huge difference between them. Table VI demonstrates the new amount of training data after using three imbalanced techniques.

D. Feature Extraction

After loading the original data, we found that there were 78 features in total. To simplify our model and speed up the training process, we need to do the feature selection.

We first applied the ANOVA test to the original data set. An ANOVA test is always used to find whether the experiment results are significant. Basically, we can use the ANOVA test to test groups to see if there's a difference between them.

In past research, researchers selected 30 to 45 features for their models. However, we still find that their models easily got overfitting. In feature filtering methods, we decided to select 20 features in total.

We used SelectKBest and f_classif functions to implement this step. The f_classif function can compute the ANOVA F value for the provided samples. Therefore, by comparing the ANOVA F values between each feature, we can select the top 20 highest scores to select the 20 most important features. The result of the selected features and their corresponding scores is shown in Table VII.

The principal component analysis (PCA) is another way for feature selection. It's different from the feature filtering method because it will generate new features instead of selecting features from the original features.

TABLE VI. NUMBER OF INSTANCES AFTER USING AN IMBALANCED TECHNIQUE IN TRAINING DATA

Label	Number of Instances
Benign	1,676,045
Botnet Areas	5,000
Brute Force	10,000
DoS	257,416
Infiltration	500
Port Scan	72,555
Web Attack	5,000

TABLE VI. 20 FEATURES SELECTED BY F VALUE IN ANOVA TEST

Feature Name	Score	Feature Name	Score
Bwd Packet Length Std	621886.79	Idle Max	331069.96
Bwd Packet Length Max	570757.59	Flow IAT Max	324819.65
Bwd Packet Length Mean	556456.87	Fwd IAT Max	324435.57
Avg Bwd Segment Size	556456.87	Idle Mean	323439.34
Packet Length Std	510653.75	Idle Min	304482.61
Max Packet Length	467789.87	Flow IAT Std	235691.57
Packet Length Variance	456555.07	Min Packet Length	155809.38

PCA is a commonly used dimensionality reduction method. The principal components are constructed in descending order for the variance in the data set. In our research, we decided to keep 99% information from the dataset. After calculation, we only need to keep 4 principal components.

The PCA is implemented by the function in the sklearn library., which provides a method to implement linear dimensionality reduction using Singular Value Decomposition (SVD) of the data to project it to a lower dimensional space.

The dimension of the training set with each feature selection method is shown in Table VIII.

During the PCA process, we are able to calculate the features with the highest weight in the principal components. The result is shown in Table IX.

IV. EXPERIMENT

After pre-processing and feature selection for experimentation, we created different datasets to experiment with. As stated in the above section, we used three techniques to solve the undersampling problem, and for feature representation, we employed two techniques feature selection and PCA, for feature deduction. With the above methods, we created a dataset for each technique and trained them.

In this paper, we divided the problem into two different models; first, we trained a binary classifier that classifies the input features into Benign or Attack; second, we trained a

TABLE VIII. THE FINAL DIMENSION FOR THE TRAINING SET AFTER FEATURE EXTRACTION

Class Label	Origin Data	Feature Filtering	PCA
Benign1	(1000000, 78)	(1000000, 20)	(1000000, 4)
Benign2	(676045, 78)	(676045, 20)	(676045, 4)
Botnet Areas	(4931, 78)	(4931, 20)	(4931, 20)
Brute Force	(9993, 78)	(9993, 20)	(9993, 4)
DoS	(257416, 78)	(257416, 20)	(257416, 4)
Infiltration	(500, 78)	(500, 20)	(500, 4)
Port Scan	(72555, 78)	(72555, 20)	(72555, 4)

TABLE IX. FEATURE WITH HIGHEST WEIGHT

Class Label	Feature with highest weight	Weight
Benign1	URG Flag Count	55.2%
Benign2	Active Max	22.9%
Botnet Areas	Fwd IAT Total	43.5%
Brute Force	Fwd IAT Total	43.5%
DoS	Idle Mean	83.3%
Infiltration	Average Packet Size	48.1%
Port Scan	Bwd IAT Total	37.1%

multi-class classifier that classifies the input features into different types of attacks. This way, we could tackle the problem more efficiently since 80% of the dataset was Benign and some classes were less than 1% of the dataset; because of this, the model was unable to learn the features of the undersampled classes even after using oversampling techniques. Dividing the problem into two different models made the outcome better.

For the training process, we used four different machine-learning algorithms; Logistic regression, KNN, Random Forest, and XGboost. First, we trained the above-described dataset for binary classification using four machine-learning algorithms.

For KNN, we experimented with two different values of k, k=n/5, and k=sqrt(n), where n is the number of data points. Since we had around 3 million data points, the model with k=n/5 was not converging and was taking to a longer period of training time. So we trained the KNN with k=sqrt(n), which converged in around 20 mins. For other algorithms, we used default parameters.

I. RESULTS

After training the dataset with three different oversampling techniques, We found that the ADASYN technique for oversampling worked better than others by a small margin. We also trained our model with different feature selection and feature reduction techniques. With the best 20 features obtained from the confusion matrix, we expected to get a reduction in training time with a small reduction in overall accuracy, but after experimentation, we found that we had a very small reduction in training time, and accuracy degraded significantly. Similarly, with features we get after PCA, the training time difference with all feature models was insignificant, and accuracy was degrading significantly. These results led to continuing with the original feature, the experimental result of the all-feature model is given in Table X. From Table X, we can infer that for binary classification, both random forest and XGboost have similar results and work better than other algorithms.

Similarly, for attack classification, we conducted a similar experiment as binary classification. With similar results, we concluded that for attack classification, all feature model is giving the best results. The experimental result of all four machine learning models for attack classification is given in Table XI. From the table, we can infer the random forest and XGboost are giving comparable F1 scores.

TABLE X. BINARY CLASSIFICATION

Model	Accuracy	Recall	Precision	F1 Score
LR	0.9262	0.8124	0.9117	0.8513
KNN	0.9683	0.9455	0.9422	0.9439
RF	0.9988	0.9981	0.9976	0.9979
XGBoost	0.9990	0.9992	0.9974	0.9983

TABLE XI. ATTACK CLASSIFICATION

Model	Accuracy	Recall	Precision	F1 Score
LR	0.9685	0.8146	0.8909	0.8094
KNN	0.9920	0.7523	0.7646	0.7563
RF	0.9997	0.9986	0.9975	0.9981
XGBoost	0.9997	0.9990	0.9980	0.9985

II. CONCLUSION

In conclusion, having two classification stages helps in tackling the oversampling problem. We divided our problem into two different sub-problems, which were binary classification and multi-class classification. With experimentation, we concluded for both sub-problems, random forest and XGboost gave better results. In the future, we need to dive deep into the highly imbalanced data problem between normal and attack network traffic. In addition, how we can merge the models or add prior to mitigate the massive difference in the number of samples between benign and attack classes and to increase the accuracy of the model.

REFERENCES

[1] L. Yang, A. Moubayed, I. Hamieh, and A. Shami, "Tree-Based Intelligent Intrusion Detection System in Internet of Vehicles," 2019 IEEE Global Communications Conference (GLOBECOM), 2019.

[2] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," 2015 Military Communications and Information Systems Conference (MilCIS), 2015.

[3] T. Elmasri, N. Samir, M. Mashaly, and Y. Atef, "Evaluation of CICIDS2017 with Qualitative Comparison of Machine Learning Algorithm," 2020 IEEE Cloud Summit, 2020.

[4] C. I. f. Cybersecurity. https://www.unb.ca/cic/datasets/ids-2017.html.

[5] R. Panigrahi and S. Borah, "A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems," International Journal of Engineering & Technology, 2018.

[6] S. Yadav and G. P. Bhole, "Handling Imbalanced Dataset Classification in Machine Learning," 2020 IEEE Pune Section International Conference (PuneCon), 2020.

[7] M. Khalil, E. Fantino, and P. Liatsis, "Evaluation of Oversampling Strategies in Machine Learning for Space Debris Detection," 2019 IEEE International Conference on Imaging Systems and Techniques (IST), 2019.