Intrusion Detection System: A Robust Machine Learning Approach

Md Rakibul Islam
Computer Science Department
University of New Orleans
LA, USA
e-mail: mislam3@uno.edu

Aayush Nagpal
Computer Science Department
University of New Orleans
LA, USA
e-mail : anagpal@my.uno.edu

***Abstract*—In this study we aim to develop a robust approach for intrusion detection system (IDS) using machine-learning techniques. The approach can work on different datasets with high accuracy, which is an important demand for IDS in recent. Unfortunately, the current literature short in this aspect.**

Keywords—Intrusion; Anomaly; Machine learning; KDD;

I. Introduction

The term intrusion refers to any unauthorized access that attempt to compromise confidentiality, integrity and availability of information resources. In general, we can say that any malicious use or misuse of any entity in a network can be referred as an intrusion. The intruder tries to find the vulnerability in the security system and then prepare for attack. Intrusion detection is the process of fast detection of unwanted violation in system's normal behavior due to attacks performed by the malicious user. Intrusion detection system (IDS) deals with detection of such type of attacks in just in time or real time and report, alert or countermeasure on the attack to the administrator.

Now a days, artificial Intelligence, data mining and machine learning algorithms have been subjected to extensive research in intrusion detection with emphasis on improving the accuracy of detection and make an immune model for Intrusion Detection System (IDS) to tackle zero day attacks or novel attacks. While many researches [5][6] focused on the accuracy and timely detection of attacks in the network, none of the studies aims at the robustness of their proposed approach. Here, robustness of an approach indicates how effective it is to detect anomaly in different datasets. To the best knowledge, we, for the first time, investigate in this direction and proposed an approach that can achieve high performance to detect anomaly in different datasets.

In particular we ask following two research questions to conduct the project-

RQ1: Can we develop a robust holistic approach using machine-learning (ML) algorithms for IDS?

- Such approach will be very useful to detect in a broader range of attack, which, in turns, will minimize the requirement of having multiple systems to detect anomaly and ensure more secured environment.

RQ2: Can we identify the reason(s) why different ML algorithms perform in different levels of accuracies?

- Such revelation of reason(s) will help the research community to understand the pros and cons of ML algorithms to develop their own IDS.

II. Data Collection

As we aim to find a robust approach of IDS using ML algorithms that can work different datasets, so we collect three datasets, e.g., KDDCup99 [1][4], NSL-KDD [1] and GureKDD [2]. While NSL-KDD is processed version of existing KDDCup99 dataset, hence those are same dataset, which does not suit with our goal to use different datasets for testing and training. Descriptions of these datasets are given below-

A. *KDDCup99*

The KDD dataset was used in the UCI KDD1999 competition. The objective of the competition is to develop intrusion detection system models to detect attack categories i.e. DOS, PROBE, R2L and U2R. The KDD Cup99 dataset available in three different files such as KDD full dataset, which contains 489,8431, instances, KDD Cup 10% dataset, which contains 494,021 instances, KDD Corrected dataset which contains 311,029 instances. We use the full dataset for our study.

B. *GureKDD*

GureKDDcup [6] dataset contains connections of kddcup99 (database of UCI repository) but it adds its payload (content of network packets) to each of the connections. It will permit to extract information directly from the payload of each connection to be used in machine learning processes.

The GureKDDCup capture team follows the same steps followed to generate kddcup99. They processed tcpdump files with bro-ids [17] and got each connection with its attributes. Finally, the dataset is labeled each connection based on the connections-class files (tcpdump.list) that MIT provides. The size of the dataset is 9.3 GB and 6 percent dataset's size is 4.2 GB. We use the 6 percent of the dataset for our study.

III. Methodology

After collecting the datasets, we take KDDcup99 as training and GureKDD as testing. Then we select the algorithms for our study. Before feeding the data in the ML algorithms we preprocess the datasets. We also perform manual investigation and modification to make the datasets compatible to run the algorithms. In the following sections we detail describe the steps we carry out to perform our jobs. Figure 1 also depicts our study procedure.

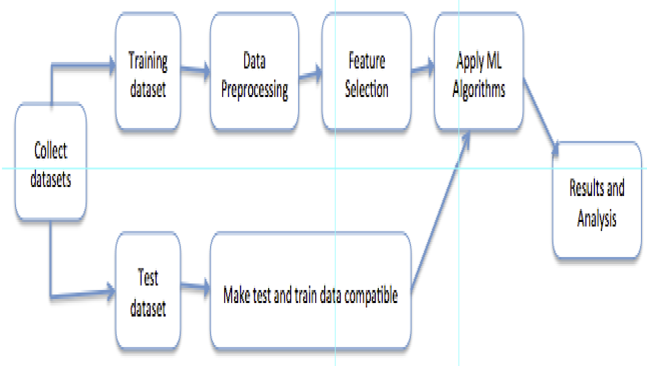

Figure 1- The Study Procedure

A. *Selecting the algorithms*

We select four ML algorithms e.g., Logistic regression [3], SMO [3], J48 [3] and Random Forest (RF) [3]. The first two algorithms are function base, whereas, the later two algorithms are tree base. All those algorithms are chosen wisely as those are popularly used in the literature for binary classification, which suit in our problem domain as well.

J48 and Random forest and decision based The decision tree is built by maximizing the information gain at each variable split, resulting in a natural variable ranking or feature selection. When building the decision tree, at each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of examples into subsets.

The advantages of decision trees are intuitive knowledge expression, high classification accuracy, and simple implementation. The main disadvantage is that for data including categorical variables with a different number of levels, information gain values are biased in favor of features with more levels. Smaller trees are obtained from bigger ones by pruning. Larger trees often have high classification accuracy but not very good generalization capabilities. By pruning larger trees, smaller trees are obtained that often have better generalization capabilities

Another algorithm we are using in this study is SMO which is Sequential minimal optimization (SMO) is an algorithm for solving the quadratic programming (QP) problem that arises during the training of support vector machines. The SVM is a classifier based on finding a separating hyper plane in the feature space between two classes in such a way that the distance between the hyper plane and the closest data points of each class is maximized. The approach is based on a minimized classification risk rather than on optimal classification.

Motivation for keeping four different ML algorithms is to select a model, which would prove to be robust keeping problems such as over fitting and under fitting in consideration. Out of four Ml algorithm, Logistic and SMO is linear model, which deals efficiently with over fitting. J48 and Random forest are based on decision tree and ensemble respectively, which are prone to over fit at times.

B. *Data preprocessing*

Data preprocessing is the most time consuming and complex task of preparing for subsequent analysis as per requirement for IDS model. If the dataset contains duplicate records then the clustering or classification algorithms take more time and also provides inefficient results. To achieve more accurate and efficient model your dataset should be free from noise and redundancy samples. To do data preprocessing, we perform following steps

a) Remove duplicate records:

b) Removing outliers:

c) Perform numeric transformation (e.g., log scale)

d) Normalize the dataset.

C. *Feature Selection*

Feature selection is an important data processing step. Selecting the appropriate features can increase the efficiency of ML algorithms; at the same time it reduces the curse of data dimensionality. There are many techniques for features selection among which information gain (IG) and principal component analysis are the two most popular approaches in the literature.

Due to use two different datasets PCA is not applicable in our study, so we opt for IG technique. Using that technique we select 12 features in addition to the class feature.

Table 1- Selected 12 features

SL	Name	Description
01	diff_srv_rate	The percentage of connections that were to different services
02	same_srv_rate	The percentage of connections that were to the same service
03	Count	Sum of connections to the same destination IP address
04	Service	http, ftp, smtp, telnet etc.
05	Flag	Connection status.
06	src_bytes	Bytes sent in one connection
07	dst_bytes	Bytes received in one connection
08	dst_host_diff_srv_rate	The percentage of connections that were to different services
09	dst_host_same_srv_rate	The percentage of connections that were to the same service
10	dst_host_srv_count	Sum of connections to the same destination port number
11	dst_host_srv_serror_rate	The percent of connections that have activated the flag
12	dst_host_serror_rate	The percentage of connections that have activated the flag

IV. Results and Analysis

There are several criteria by which the ML/DM methods for cyber could be compared:

- Accuracy
- Time for training a model
- Understandability of the final solution (classification)
- Time for classifying an unknown instance with a trained model

Even though above parameters do matter in selecting best suitable algorithm for data mining problems. In this study we are primarily focusing on accuracy of the model above all other parameters. Analysis was performed in an incremental fashion. Every step motivated us to perform further investigation and provided us more insight about the data and the behavior of its features. Note that we use two different datasets namely KDDCup99 and GureKDD and make those compatible and use train as KDDCup99 and test as GureKDD.

We run the ML algorithms with raw and preprocessed datasets. In addition, we also perform the analysis with minor preprocessing that we call process dataset. In the following we describe how we run the ML algorithms using those datasets.

RUN 1- we trained all the models using four different ML algorithms on KDDCup99 without preprocessing and tested it with GureKDD. Results are provided in the Table 2 for this run.

RUN 2- here we perform preprocessing as discussed in the above section on both the datasets (test & train) and observe the performance given in Table 3.

RUN 3- Again, we observed that two of the 12 features in the selected have high ranges and the distribution is also right skewed as observed in the Weka visualization. Those are needed to be preprocessing by removing outliers, extreme values and performing normalization. We preprocessed those two features and trained and tested our model and observed the results in Table 4.

RUN 4- results from RUN 3 provided more feature specific insight and thus motivated us to balance out the dataset as suggested in literature review. Results in this RUN 4 are much more satisfactory and represents true accuracies, which are presented in Table 5 and Table 6.

Table 2- Performance measurement of ML algorithms using raw dataset.

ML Algorithm	Class	Precision	Recall	F-Measure	ROC Area
J48 - Raw	Normal	0.984	0.993	0.988	0.579
	Anomaly	0.458	0.265	0.336	0.579
SMO- Raw	Normal	0.992	0.646	0.783	0.705
	Anomaly	0.046	0.764	0.088	0.705
RFRaw	Normal	0.984	0.997	0.99	0.669
	Anomaly	0.649	0.283	0.394	0.669
Logistic- Raw	Normal	0.992	0.702	0.822	0.636
	Anomaly	0.054	0.76	0.101	0.669

Table 3- Performance measurement of ML algorithms using process dataset.

ML Algorithm	Class	Precision	Recall	F-Measure	ROC Area
J48- Process	Normal	0.984	0.989	0.986	0.588
	Anomaly	0.357	0.292	0.315	0.606
SMO- Process	Normal	0.981	0.721	0.831	0.55
	Anomaly	0.03	0.38	0.055	0.55
RF-Process	Normal	0.984	0.97	0.977	0.717
	Anomaly	0.175	0.284	0.216	0.751
Logistic-Process	Normal	0.976	0.793	0.875	0.623
	Anomaly	0.012	0.12	0.023	0.592

Table 4- Performance measurement of ML algorithms using preprocess dataset.

ML Algorithm	Class	Precision	Recall	F-Measure	ROC Area
j48 -PreProcess	Normal	0.988	0.804	0.886	0.402
	Anomaly	0.005	0.085	0.009	0.402
SMO _PreProcess	Normal	0.989	0.954	0.967	0.989
	Anomaly	0.012	0.065	0.021	0.011
RF-PreProcess	Normal	0.989	0.96	0.975	0.535
	Anomaly	0.011	0.04	0.017	0.431
Logistic-PreProcess	Normal	0.989	0.911	0.948	0.467
	Anomaly	0.005	0.042	0.009	0.381

Table 5- Performance measurement of ML algorithms using raw balanced dataset.

ML Algorithm	Class	Precision	Recall	F-Measure	ROC Area
J48_Raw_Balanced	Normal	0.45	0.898	0.6	0.508
	Anomaly	0.596	0.12	0.2	0.029
SMO_Raw_Balamced	Normal	0.513	0.785	0.621	0.594
	Anomaly	0.7	0.403	0.511	0.594
RF_Raw_Balanced	Normal	0.554	0.95	0.7	0.726
	Anomaly	0.907	0.386	0.542	0.726
Logistic_Raw_Balanced	Normal	0.526	0.881	0.658	0.559
	Anomaly	0.791	0.362	0.497	0.573

Table 6- Performance measurement of ML algorithms using preprocessed balanced dataset.

ML Algorithm	Class	Precision	Recall	F-Measure	ROC Area
J48_Balanced_Prerocess	Normal	0.877	0.963	0.918	0.541
	Anomaly	0.527	0.232	0.322	0.563
SMO_Balanced_PreProcess	Normal	0.864	0.993	0.924	0.553
	Anomaly	0.113	0.196	0.254	0.553
RF_Balanced_PreProcess	Normal	0.878	0.994	0.932	0.403
	Anomaly	0.853	0.214	0.342	0.392
Logistic_Balanced_PreProcess	Normal	0.863	0.933	0.897	0.527
	Anomaly	0.295	0.16	0.207	0.695

V. Conclusion

In this study we have used different ML algorithms on different datasets to build a robust approach for IDS. We have found Random Forest (RF) has performed the best among the selected algorithms along with the preprocessing steps. We would recommend usage of decision tree base (e.g., RF) for a robust anomaly base IDS.

References

[1] http://kdd.ics.uci.edu/databases/kddcup99/task.html

[2] http://www.aldapa.eus/res/gureKddcup/README.pdf

[3] Anna L. Buczak, "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection," in IEEE COMMUNICATIONS SURVEYS & TUTORIALS, VOL. 18, NO. 2, SECOND QUARTER 2016, pp. 1153-1174.

[4] M.Tavallaee,E.Bagheri,W.Lu,andA.Ghorbani,"Adetailedanalysis of the KDD Cup 1999 data set," in Proc. 2nd IEEE Symp. Comput. Intell. Secur. Defense Appl., 2009, pp. 1–6..

[5] A. Bivens, C. Palagiri, R. Smith, B. Szymanski, and M. Embrechts, "Network-based intrusion detection using neural networks," Intell. Eng. Syst. Artif. Neural Netw., vol. 12, no. 1, pp. 579–584, 2002.

[6] Y. Zhang, L. Wenke, and Y.-A. Huang, "Intrusion detection techniques for mobile wireless networks," Wireless Netw., vol. 9, no. 5, pp. 545–556, 2003.