Classification of Attack Types for Intrusion Detection Systems using a Machine Learning Algorithm

Prof. Shraddha Chourasia[1], Kaushal Potphode[2], Apoorva Ghagare[3], Ayush Indurkar[4], Dikshant Gaikwad[5], Varun Sayam[6]
[1,2,3,4,5,6]Jhulelal Institute of Technology, Nagpur, India

***Abstract*— On this paper, we present the outcomes of our experiments to evaluate the performance of detecting special Sorts of attacks (e.g., IDS, Malware, and Shellcode). We examine the recognition performance by making use of the Random Forest algorithm to the numerous datasets which are evaluated from the Kyoto 2006+ dataset, which is the recent network packet information accumulated for developing Intrusion Detection Systems. And we conclude with discussions and future studies projects.**

***Keywords*— Machine Learning, Supervised Machine Learning, Kyoto2006+, Labelling, Intrusion Detection System, Classification, Attacks.**

I. INTRODUCTION

The IDS (Intrusion Detection device) is a protection towards Attacks attempting to thieve statistics saved on numerous structures together with servers and personal computers. In the case of widely recognized attacks, it is straightforward for the administrator to judge and process it immediately, however it is doubtful to choose unknown abnormal data, and the cost of recovery increases because the handling is not on time [8].

Machine learning techniques are extensively utilized in IDS because of its potential to categorize regular/attack network packets with the aid of learning patterns based on the accumulated records. There are many consequences for classification of regular/attack, but, it is required to do some work analysing different attack types.

We took Kyoto 2006+ as a dataset (learning) [1]. Kyoto 2006+ dataset incorporates network traffic records accrued from November, 2006 to December, 2015. The dataset is additionally to be had for big data evaluation, of which length is 19.683 Gigabytes. Similarly, the most commonly used dataset for IDS research is KDD Cup99 [7]. However, the dataset was accumulated in 1999, and might not incorporate the present day network intrusion patterns. Further, this dataset is gathered from a virtual network environment, which makes it exclusive from the styles determined in real network structures.

On this paper, we describe our work. The section 2 describes the related work and background knowledge, And the section 3 details the evaluation processes that consists of the construction of various datasets for testing the algorithm and the evaluation configuration. The section 4 presents the consequences of our experiments to assess the performance of detecting IDS, Malware and Shellcode. In the end, we finish with discussions and future work.

II. BACKGROUNRD AND RELATED WORK

A. Machine Learning for IDS and Classification

There are some of works of intrusion detection system that applied machine learning algorithms to Kyoto 2006+ dataset [1]. Song et al. proposed an intrusion detection method based on correlation of the outcomes acquired from the two one-class SVM models, one version trained with raw traffic data and the next version trained with Snort indicators, respectively [2]. In[3], Sallay et al. proposed a real time intrusion detection alert classifier based on online self-trained support vector machines on the way to perceive actual attacks efficaciously even when a high ratio of false positives exists in the intrusion alerts. In [4], Chitrakar et al. evolved an intrusion detection device based totally on a candidate support vector based incremental SVM algorithms, and developed the IDS the use of the Kyoto 2006+ dataset. In [5], Ishida et al. Proposed an anomaly primarily based intrusion detection approach that mixes OptiGrid clustering and a cluster labelling algorithm using grids to extract the feature of traffic data and discover the attack traffic correctly. In [6], Ambusaidi et al. proposed a mutual

information primarily based algorithm to select the most advantageous set of capabilities for category from high dimensional traffic data and a least rectangular aid vector system primarily based IDS integrated with the ultimate function selection algorithm.

B. Evaluation Metrics

Deciding on assessment indicator that could examine overall performance objectively is vital for evaluating performance between numerous methods or one-of-a-kind datasets. IDS overall performance evaluation normally uses accuracy mainly. Accuracy is computed because the wide variety of efficaciously categorised records over the entire range of facts. True positive (TP) and True Negative (TN) way the variety of efficiently labelled as Positive or Negative. False Positive (FP) way that a Negative example is anticipated as positive, and False Negative (FN) way the other (expected negative while the example is positive).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

Although accuracy is an intuitive dimension, it can supply when the information is imbalanced. As an example, a dataset of one thousand times includes 990 positive and 10 negative instances. For a excessive accuracy dimension, one could ignore all of the negative cases and predict an enter as high-quality. This results in a high accuracy of 0.99.

Precision indicates the part of data effectively expected positive over the variety of data predicted as positive.

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

Recall (i.e., sensitivity, detection rate) indicates the share of effectively expected positive instances out of the number of actual positive times. It serves as a first-rate overall performance indicator when it's important to come across all positive data, inclusive of IDS.

$$Recall = \frac{TP}{FN+TP} \quad (3)$$

To increase the precision of a certain class, you simplest want to expect that a data instance belongs to a sure class simplest when the possibility of belonging to the class could be very high. At the equal time, the overall performance of the recall becomes decreased due to the fact there are times belonging to the class are in all likelihood to be excluded due to the excessive threshold.

Precision and Recall are each important indicators and using best certainly one of them is not enough to evaluate the IDS overall performance. F1-score is the harmonic suggest of the two, which considers Precision and Recall together. With unbalanced binary category datasets, F1-score may be a higher indicator than accuracy. F1-score may be acquired the use of the subsequent equation that computes Fβ-score, via substituting β with 1.

$$F_\beta = \frac{(1+\beta 2)(PXR)}{(\beta P+R)} \quad (4)$$

F1 is a metric that considers precision and recall similarly, even as F2 considers consider two times extra important than precision. F0.5 considers precision twice as crucial as recall. Seeing that IDS needs to hit upon all the attacks as a great deal as possible, the usage of F2 as the primary metric is suitable. On this take a look at, all the essential metrics (accuracy, precision, recall, and F-score) are offered for comparison within the evaluation segment.

III. EVALUATION

This segment describes the experimental manner and the assessment results that we received.

A. Data Preparation

The Kyoto 2006+ dataset is a big 19.78GB of data collected during the last 9 years and two months from November 2006 to December 2015. The data used within the experiments were selected in keeping with the subsequent criteria. We initially built a dataset between 3 and 7 days as recommended by Dr. Song, one of the researchers who accumulated the Kyoto 2006+ dataset and an author of [1]. The bigger the data set, the greater noise is in all likelihood to be present, in an effort to be learned by means of machine learning knowledge of algorithms. However, we discovered that the range of data sets in line with class within the dataset is reduced considerably when the attack class is subdivided into unique attack types (e.g., IDS, Malware, and Shellcode). Therefore, we set the duration of data accumulated a month, so that the dataset consists of all distinctive attack types and has enough data consistent with attack type. Figure 1 indicates the system of constructing numerous datasets. First, the month-to-month facts for the six labels are obtained and pick out the appropriate month for the reason of observe. In the end, subdivide labels in the system of creating training and check datasets.

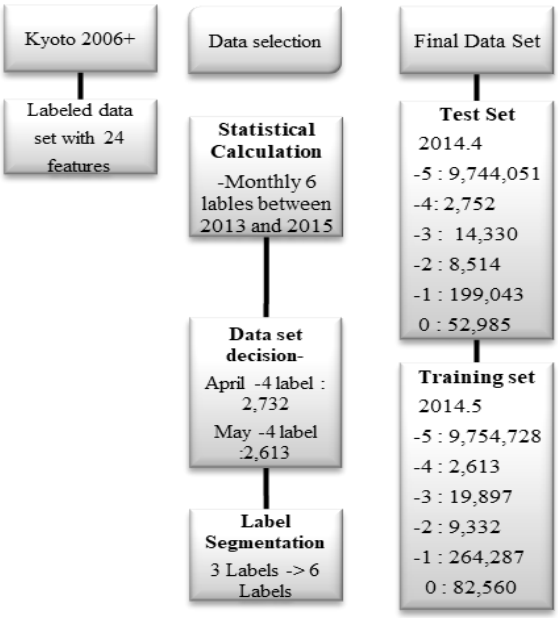

Figure 1 The process of building training and test datasets for evaluation

The Kyoto 2006+ dataset contains 3 class types: -1 (attack), -2 (Shellcode), and 1 (normal). For the -1 and -2 labels, there are three attack types (e.g., IDS, Malware, Shellcode). As shown in Table 1, a single packet may be detected by means of several IDSs. As an example, the packet within the fourth row become detected by using IDS and Shellcode on the equal time, and its unique label is -1. However, we discovered that a few packets that have been detected via IDSs are categorized as ordinary (e.g., 1). We eliminated those data, considering them as errors. Extra information about the data collection process is stated [1].

Table 1 Label Segmentation Table 1 Label Segmentation

IDS	Malware	Shellcode	Label	New Class	New Label
0	0	0	1	Normal	0
0	0	0	-1	Unknown	-5
0	0	1	-2	Shellcode	-4
1	0	1	-1	IDS+Shellcode	-3
1	1	1	-1	IDS+Shellcode	
0	1	0	-1	Malware	-2
0	1	1	-1	Malware	
1	1	0	-1	Malware	
1	0	0	-1	IDS	-1

Table 1 shows the new label assignment criteria. We set the class name as the detected attack field name, and within the case of detecting malware down with different attack types, we use malware. Due to the fact the detection of malware is more critical than the detection of the other two threats. Table 2. indicates the wide variety of instances for every class. It is able to be seen that the unknown attack occupies the largest range of statistics, and the sizes of the ultimate classes are less than 1% besides IDS class.

Table 2 Statistic of training and test data. The number in parentheses denotes the percentage of the label in the set.

Label	Training (2014.5)	Test (2014.4)
-5	9,754,728 (96.26%)	9,744,051 (97.23%)
-4	2,613 (0.03%)	2,752 (0.03%)
-3	19,897 (0.2%)	14,330 (0.14%)
-2	9,332 (0.09%)	8,514 (0.08%)
-1	264,287 (2.61%)	199,043 (1.99%)
0	82,560 (0.81%)	52,985 (0.53%)

B. Selection of Evaluation Metrics and Machine Learning Algorithm

In network intrusion detection systems, it is crucial to recognize how properly the intrusion is detected, so the recall is more crucial than the precision. Therefore, both F1-score and F2-score are supplied in this, have a look at.

The Random Forest algorithm become implemented the usage of the Scikit-learn 0.19.0 version [], a machine learning library written in Python 3.6 version. The PC specification used within the evaluation changed into Intel Core i5 3.1 GHz, 8GB memory, and Mac OS. The parameter values of the algorithm are set to the default values.

C. Learning Features

We used 17 functions for our experiment, which includes 14 objects that are selected as crucial for intrusion detection in KDD Cup 99 and 3 of 10 newly created objects in Kyoto 2006+. Since the 10 newly created features in Kyoto 2006+ (see Table 3) were created for verification and include critical details for attack detection, we excluded the three detection fields (IDS, Malware, Shellcode) at some stage in the training session which can improve the evaluation performance. Then, all of the statistics were normalized.

Table 3 New Features of Kyoto2006+

Feature	Meaning	Note
Duration	The length of the connection.	Used
Service	The connection's service type, e.q., http, telnet, etc.	Used
Source Bytes	The number of data bytes sent by the source IP address.	Used
Destination Bytes	The number of data bytes sent by the destination IP address.	Used
Count	The number of connections whosesource IP address and destination IPaddress are	Used

	the same to those of the current connection in the past two seconds.	
Same_srv_rate	Percentage of connections to the same service in Count feature.	Used
Serror_rate	Percentage of connections that have "SYN" errors in Count feature.	Used
Srv_serror_rate	Percentage of connections that have "SYN" errors in Srv_count feature.	Used
Dst_host_count	Among the past 100 connections whose destination IP address is the same to that of the current connection, the number of connections whose source IP address is also the same to that of the current connection.	Used
Dst_host_srv_count	Among the past 100 connections whose destination IP address is the same to that of the current connection, the number of connections whose service type is also the same to that of the current connection.	Used
Dst_host_same_src_port_rate	Percentage of connections whose source port is the same to that of the current connections in Dst_host_count feature.	Used
Dst_host_serror_rate	Percentage of connections that have "SYN" errors in Dst_host_srv_count feature.	Used
Dst_host_srv_serror_rate	Percentage of connections that "SYN" errors in Dst_host_srv_count feature.	Used
Flag	The state of the connection at the time the summary was written.	Used
IDS	It indicates whether IDS(Intrusion Detection System) triggered an alert for the connection; '0' means any alerts were not triggered, and an arabic numeral(except '0') means the different kinds of the alerts. Parenthesis indicates the number of the same alert observed during the connection.	Unused
Malware	It indicates whether malware, also known as malicious software, was observed in the connection; '0' means no malware was observed, and a string indicates the corresponding malware observed at the connection.	Unused
Shellcode	It indicates whether shellcode; '0' means no shellcodes and exploit codes were observed, and an arabic numeral (except '0') means the different kinds of the shellcodes or exploit codes. Parenthesis indicates the number of the same shellcode or exploit code observed during the connection.	Unused
Label	It indicates whether the session was attack or not; '1' means the session was normal, '-1' means known attack was observed in the session, and '-2' means unknown attack was observed in the session.	Used (Training)
Source IP Address	It indicates the source IP address used in the session. Also, the same private IP addresses are only valid in the same month.	Unused
Source Port Number	It indicates the source port number used in the session.	Used
Destination IP Address	indicates the source IP address used in the session. Also, the same private IP addresses are only valid in the same month.	Unused
Destination Port Number	It indicates the destination port number used in the session.	Used
Start Time	It indicates when the session was started.	Unused
Protocol	It indicates the protocol type.	Used

IV. RESULTS

This section describes the outcomes of making use of the Random Forest algorithm to the chosen training and test datasets we built. First, we experimented on predicting six classes using a dataset that have been accrued between April and May, 2014. Subsequent, we built a further dataset to tackle the data imbalance trouble, and then we performed the experiments on the new set once more.

A. Prediction Results

Table 4 indicates the overall performance of six class predicted through Random Forest algorithm in the order of precision, Recall, F1-score, F2-score, and accuracy, when the data collected in May, 2014 is used for training and the statistics collected in April, 2014 become used for testing. The result value shown in the end is the weighted average, averaging the overall performance values weighted through its class size.

The general performance (i.e., weighted average) is right enough to attain 0.99 for all evaluation metrics. But, the overall performance of predicting every attack type differs greatly. While the detection rate for unknown attack is excessive (F1 score of 0.99), the detection of Shellcode attack (-4 label) suggests a bad performance as low as 0.16 of F1 score. The IDS attack and regular classes (-1 and 0 labels) did now not show properly outcomes both. Because the information is thoroughly unbalanced, the weighted average overall performance can be excessive when the primary class detection suggests an acceptable result, neglecting the overall performance of classes containing small plenty less data.

Table 4 Performance comparison of the Random Forest algorithm for Dataset A

	Precision	Recall	F_1**Score**	F_2 **score**	Accuracy	Number of instances
-5	0.99	1.00	0.99	1.00		9744051
-4	0.31	0.11	0.16	0.13		2752
-3	0.82	0.91	0.86	0.89		14330
-2	0.98	0.99	0.98	0.99		8514
-1	0.89	0.70	0.78			181125
0	0.69	0.74	0.72	0.73		52985
Result	0.99	0.99	0.99	0.99	0.99	10003757

B. Prediction Results of Under-Sampled Datasets

Inside the previous dataset, the label -4 (Shellcode) contained the smallest range of instances. To address the data imbalance trouble, we randomly below-sampled the training data, putting the quantity of all the instructions to the range of class -4, which is 2,613. Therefore, the variety of times of all the training

become arbitrarily adjusted to 2,613. The equal test set (April, 2014) became once more used for evaluation.
Table 5 suggests the results of this experimentation. The overall performance has dropped drastically and the performance for every class has additionally reduced. We consider that the 2,613 instances according to class have been no longer enough for the machine learning algorithm. The overall performance of the class -4 (Shellcode) become even lower than that of the previous data set.

Table 5 Performance comparison of the Random Forest algorithm for under-sampled dataset

	Precision	Recall	F_1Score	F_2 score	Accuracy	Number of instances
-5	0.99	0.85	0.92	0.87		9744051
-4	0.02	0.09	0.03	0.05		2752
-3	0.11	0.41	0.18	0.27		14330
-2	0.20	0.81	0.32	0.50		8514
-1	0.13	0.76	0.22	0.39		181125
0	0.06	0.55	0.10	0.21		52985
Result	0.97	0.84	0.90	0.86	0.84	10003757

V. CONCLUSION AND FUTURE WORK

In this have a look at, we analzed class-particular detection of Kyoto 2006+ datasets, the usage of the Random Forest algorithm, an efficient supervised machine learning algorithm for IDS. We first refined the authentic 3 classes (i.e., normal, well recognised attack unknown attack) into 6 classes (i.e., normal, unknown, shellcode, IDS+shellcode, malware, IDS). Next, we built one test dataset and the two training datasets that vary inside the length among classes for comparing the performance of detecting the attack types. Although we obtained a high average detection performance while trained with the primary training set (0.99 of precision, recall, F1-score, and F2-score), we located that the performance for every class differs substantially (as low as 0.16 of F1-score for shellcode attack). Because of this, we constructed the second training set through random below-sampling to set the size of all the class equal to the range of times of the smallest class (i.e., shellcode). The assessment ended in lot lower performances for all the classes, which was disappointing. We consider that the size of data changed into not sufficient, and training with the same size class may not be best for the machine learning strategies. We additionally notice that the unknown attack class still suggests an excellent overall performance, F1-score of 0.90, which suggests that there is a good pattern for the unknown attack.

In the future, we will similarly check out the overall performance of detecting distinctive attack types using the Kyoto 2006+ dataset various the training situations.

ACKNOLEGEMENT

This research changed into supported by Basic Science Research Program through the country wide studies basis of National Research Foundation of Korea(NRF) funded via the Ministry of education. (2016R1D1A1B03933002). This studies changed into supported by using the MISP(Ministry of Science, ICT & Future Planning), Korea, below the National Program for Excellence in SW(R2215-16-1005) supervised by the IITP(Institute for Information & communications Technology Promotion). This research changed into supported with the aid of the MSIT(Ministry of Science and ICT), Korea, beneath the ITRC(Information Technology Research Center) help Program (IITP-2017-0-01642) supervised through the IITP(Institute for Information & communications Technology Promotion).

REFERENCE

[1] Song, Jungsuk, et al. "Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation." Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security. ACM, 2011.

[2] Song, Jungsuk, et al. "Correlation analysis between honeypot data and IDS alerts using one-class SVM." Intrusion Detection Systems. InTech, 2011.

[3] Sallay, Hassen, and Sami Bourouis. "Intrusion detection alert management for high-speed networks: current researches and applications." Security and Communication Networks 8.18 (2015): 4362-4372.

[4] Chitrakar, Roshan, and Chuanhe Huang. "Selection of Candidate Support Vectors in incremental SVM for network intrusion detection." computers & security 45 (2014): 231-241.

[5] Ishida, Moriteru, Hiroki Takakura, and Yasuo Okabe. "High-performance intrusion detection using optigrid clustering and grid-based labelling." Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on. IEEE, 2011.

[6] Ambusaidi, Mohammed A., et al. "Building an intrusion detection system using a filter-based feature selection algorithm." IEEE transactions on computers 65.10 (2016): 2986-2998.
[7] KDD Cup 1999. Available on: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, Ocotber 2007.
[8] http://www.ddaily.co.kr/news/article.html?no=157416)