Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 127 (2018) 1–6

Procedia Computer Science

www.elsevier.com/locate/procedia

The First International Conference On Intelligent Computing in Data Sciences

Performance evaluation of intrusion detection based on machine learning using Apache Spark

Mustapha Belouch[a,*], Salah El Hadaj[a], Mohamed Idhammad[b]

[a]*Faculty of Science and Technics, Cadi Ayyad University, Marrakech, Morocco*
[b]*Faculty of Science, Ibn Zohr University, Agadir, Morocco*

Abstract

Nowadays, network intrusion is considered as one of the major concerns in network communications. Thus, the developed network intrusion detection systems aim to identify attacks or malicious activities in a network environment. Various methods have been already proposed for finding an effective and efficient solution to detect and prevent intrusion in the network, ensuring network security and privacy. Machine learning is an effective analysis framework to detect any anomalous events occurred in the network traffic flow. Based on this framework, the paper in hand evaluates the performance of four well-known classification algorithms; SVM, Naïve Bayes, Decision Tree and Random Forest using Apache Spark, a big data processing tool for intrusion detection in network traffic. The overall performance comparison is evaluated in terms of detection accuracy, building time and prediction time. Experimental results on UNSW-NB15, a recent public dataset for network intrusion detection, show an important advantage for Random Forest classifier among other well-known classifiers in terms of detection accuracy and prediction time, using the complete dataset with all 42 features.

Keywords: Intrusion Detection; Machine Learning; Apache Spark.

1. Introduction

Intrusion is defined as a set of activities that violates security objectives. In general, an intrusion detection system (IDS) must analyze the network traffic and immediately warn of potential threats. When any malicious intrusion or attack is manifest in a network, computers and information systems may suffer serious consequences when defined computer security policies are violated. Various security strategies have been employed over the years to safeguard networks. The firewall is considered as a basic packet filter; however, it has been proved that is not sufficient in providing a secure network environment. Intrusion detection working in conjunction with a firewall may provide an improved and safer network. In the literature, many IDSs have been developed via the implementation of various techniques derived from different disciplines including statistical methods, AI techniques, and more. Some IDSs are based on single-classification techniques, while others (hybrid/ensemble) include more than one classification techniques. Ensemble based IDSs present many advantages over the single classification IDS. Many works have

* Corresponding author. Tel.: +212672220196.
E-mail address: mbelouch@gmail.com (M. Belouch)., elhadajs@yahoo.fr (S. Elhadaj)., idhammad.mohamed@gmail.com (M. Idhammad).

10.1016/j.procs.2018.01.091

proposed different ensembles for ID, primarily through the exploitation of different characteristics of weak classifiers and datasets.

IDS are categorized by misuse and anomaly strategies. The misuse approach seeks known attacks referred to as attack signatures, while the anomaly approach is based on normalcy models, where any significant deviation from these reference models indicates a potential threat. However, both approaches suffer from a number of weaknesses. Misuse detection requires frequent updates of signatures to ensure ample detection, while anomaly detection presents a high false positive rate. Thus, the current challenge is to tackle these two shortcomings toward the provision of a solution with characteristics such as superior accuracy with low false positive rates.

The solution of using multiple classifiers has been widely employed to solve various classification problems, including IDS [1, 2, 3]. These methods are based on feature representation, with an accurate voting system and weighting assignment that can improve the classification rate. Furthermore, in domains with huge data volumes, such as network traffic, available resources and computational time are greatly impacted.

The purpose of this paper is to address the issue of low accuracy and prediction time in IDS. We employed multiple classifiers with different learning paradigms to evaluate different classifier model. The organization of this paper is as follows: The majority of Section 2 discusses the background and related works. Section 3 presents the various classification techniques and dataset employed in this work, and Section 4 provides experimental results and a discussion on the findings. Finally, Section 5 concludes the paper.

2. Related Work

Several research papers describing the use of machine learning methods for anomaly detection have reported the attainment of a very high detection rate of 98%, while the false positive rate was less than 1% [4]. However, when surveying state-of-art IDS solutions and commercial tools, there is no evidence for the utilization of anomaly detection approaches, presumably because experts still consider anomaly detection to be an immature technology. To discover the reason for this contrast [5] concentrated studies on the details of the research done in anomaly detection were considered from different angles, for example learning and detection approaches, training data sets, testing data sets, and evaluation methods. These studies indicated that there are some issues in the KDDCUP'99 data set [6], which is broadly used as one of the rare publicly accessible data sets for network based anomaly detection systems.

The primary critical deficiency in the KDD data set was the enormous number of redundant records. Approximately 78% and 75% of the train and test set records, respectively, are duplicated in KDD [5]. These redundant records in the train set cause learning algorithms to be one sided toward the more frequent records, and prevent it from learning infrequent records, which are typically more damaging. On the other hand, the presence of these redundant records in the test set causes the evaluation results to be biased by the methods that have better detection rates on the frequent records [5]. New NSL-KDD datasets were generated in order to solve the issues of the original KDD dataset [5], with new train and test sets (KDDTrain+ and KDDTest) consisting of selected records of the complete KDD data set. This new version of the KDD data set (NSL-KDD) is publicly available for researchers [7]. According to Tavallaee et al. and McHugh, the major disadvantage of NSL-KDD is that it does not represent actual existing networks and associated attack scenarios [8, 5].

On the other hand, the combination of classifiers for IDS has been an effective research area for several years, and many research studies have concentrated on dealing with improving the accuracy of the proposed model. The most popular classifiers were the meta classifiers: Boosting, Bagging, and Stacking. Several classification techniques and machine learning algorithms have been tested by Choudhury and Bhowal to categorize network traffic, and out of several classifiers they concluded that Random Forest and BayesNet were suitable for intrusion detection, particularly when using the Boosting method [9]. A network anomaly detection strategy using an ensemble of three-based classifiers (C4.5, Random Forest, and CART) and Particle Swarm Optimization (PSO) for feature selection was proposed, which showed promising detection accuracy and a lower positive rate in contrast to existing ensemble techniques [10].

In 2015, Moustafa et al. generated a new dataset UNSW-15 in order to counter the unavailability of network benchmark data set challenges [11]. This data set contained a fusion of actual modern normal network traffic and contemporary synthesized attack activities thereof. The authors were able to find only one paper that described the use of different existing machine learning classifiers to evaluate complexities in terms of accuracy and false positive rate algorithms on the UNSW-15 dataset [12]. The results indicated that the Decision Tree classifier accomplished the

highest accuracy of 85.56% and the lowest FAR at 15.78% compared to other classifiers (e.g., Logistic Regression, Naive Bayes, Artificial Neural Network, Expectation-Maximization Clustering). However, the UNSW-NB15 dataset was considered as complex and may be used to evaluate existing and novel methods of Network Intrusion Detection Systems [10].

3. Description of Dataset and Classification Techniques

3.1. UNSW-NB15 Data Set

As mentioned earlier, the UNSW-NB15 data set was created at the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) using the AXIA Perfect Storm tool to create a hybrid of modern normal and abnormal network traffic [10]. This data set included 49 features and nine attack families: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms.

The UNSW-NB15 dataset was decomposed into two partitions the training and the testing sets, including 175,340 and 82,000 records respectively. The two partitions are available online [13] for research purposes.

3.2. Support Vector Machine

The Support Vector Machine (SVM) is a classification method introduced in 1992 by Boser, Guyon, and Vapnik [23]. The SVM classifier is widely used due to its high accuracy, ability to deal with high-dimensional data. the SVM training algorithm aims to find a hyperplane that separates the dataset into a discrete predefined number of classes in a fashion consistent with the training examples. The term optimal separation hyperplane is used to refer to the decision boundary that minimizes misclassifications, obtained in the training step [24].

3.3. Random Forest & decision tree

Decision trees are developed by recursive partitioning [17, 20]. A univariate split is selected for the root of the tree according to some criterion, and the process repeats recursively. This process known as pruning, which decreases the tree size, is performed once a full tree has been built [16]. The most popular representative of decision trees is C4.5.

The random forest is an ensemble learning method for unpruned classification, regression or other tasks, that consists of building multiple decision trees [18]. A bootstrap sample of the original data subsets is utilized to construct multiple decision trees (Forest). Each tree in the forest gives a decision about the class of the new object that needs to be classified. The class which obtains the most votes for the object is selected by the forest.

3.4. Naïve Bayes

The Naive Bayes algorithm is an intuitive method that uses Bayes rule to compute the probabilities of each attribute belonging to each class to make a prediction [14, 15]. It simplifies the calculation of probabilities by assuming that the attributes are independent, given the label of all other attributes. Numerous studies have shown that Naive Bayes algorithms were unexpectedly accurate for classification tasks, albeit only with small databases. For some larger databases, the accuracy of decision trees was better than Naive Bayes [16].

3.5. Apache Spark

Apache Spark is a cluster computing platform designed to be fast [21], Spark extends the popular MapReduce model to efficiently support more types of computations, including interactive queries and stream processing. Speed is important in processing large datasets, as it means the difference between exploring data interactively and waiting minutes or hours. One of the main features Spark offers for speed is the ability to run computations in memory, but the system is also more efficient than MapReduce for complex applications running on disk. Spark is also designed to cover a wide range of workloads that previously required separate distributed systems, including batch applications, iterative algorithms, interactive queries, and streaming. By supporting these workloads in the same engine, Spark makes it easy and inexpensive to combine different processing types, which is often necessary in production data

analysis pipelines. In addition, it reduces the management burden of maintaining separate tools.

4. Experiments and Result Analysis

The latest Big Data Processing Tool: Apache Spark and its MLlib [22] library are used for experiment and result analysis. In the experiments, four well-known machine learning algorithms are used, namely Naïve Bayes, Decision Tree, and Random Forest are used for performance evaluation.

The classification measures are four elements: TP, TN, FP and FN. First, TP (true positive) is the number of correctly classified attacks. Second, TN (true negative) is the number of correctly classified normal records. Third, FP (false positive) is the number of misclassified attacks. Finally, FN (false negative) denotes the number of misclassified normal records. The accuracy is the percentage of the correctly classified records over all the rows of the data set, whether correctly or incorrectly classified, as reflected in the following Equation:

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

Sensitivity: It is also called true positive rate. It is used to measures the proportion of positives that are correctly identified as such.

$$Sensitivity = \frac{TP}{TP+FN} \quad (2)$$

Specificity: It is also called true negative rate. It is used to measures the proportion of negatives that are correctly identified as such.

$$Specificity = \frac{TN}{TN+FP} \quad (3)$$

4.1. Result Analysis

In this section, we discuss the result analysis of the classification based intrusion detection schemes. Performance is evaluated using UNSW-NB15 dataset.

In this experiment, we have compared the performance of four intrusion detection schemes which is designed based on well known classifiers: Naïve Bayes, Support vector machines, Decision Tree and Random forest in order to detect the attack in network traffic. Results are listed in Table 1. It is observed from table 1 that Random Forest classifier perform better than all the remaining classifiers in terms of sensitivity. This is due to fact that it gets 93.53% sensitivity followed by Decision Tree with 92.52%. Naïve Bayes and SVM have almost same sensitivity with values 92.46% and 92.13%.

Tableau 1. COMPARISON OF DIFFERENT INTRUSION DETECTION METHODS using UNSW-NB15 dataset

Methods	Accuracy	Sensitivity	Specificity	Training Time	Prediction Time
SVM	92.28	92.13	91.15	38.91	0.20
Naïve Bayes	74.19	92.16	67.82	2.25	0.18
Decision Tree	95.82	92.52	97.10	4.80	0.13
Random Forest	97.49	93.53	97.75	5.69	0.08

From the Table 1, it is observed that specificity for the Random Forest and Decision Tree based schemes are almost same with 97.75% and 97.10% respectively. However, specificity for SVM based scheme is about 91.15%. Naïve Bayes is the least ranked amongst all the classifiers in terms of Specificity. However, Naïve Bayes based scheme perform poor in terms of specificity amongst all the schemes. Among the all classifiers, Random Forest perform better in terms of accuracy with 97.49% and the accuracy of the Naïve Bayes based scheme is lower among the all schemes with 74.19%.

Naïve Bayes is fastest of all classifier methods to train, with a training time of just 2.25 seconds. On the other hand,

SVM is highest classification method used among all the four. SVM took a total of 38.91 seconds to train. Decision Tree is second most time efficient method as it took 4.80 seconds followed by Random Forest took almost 5.69 seconds.

Random Forest took the least time approximately 0.08 seconds and thus is the fastest detection scheme of all. Whereas SVM took highest time to predict with the maximum time of 0.20 seconds and thus become the slowest scheme of all the detection methods. Decision Tree ranks the second fastest scheme with approximately 0.13 seconds to predict, followed by Naïve Bayes with 0.18 seconds.

5. Conclusion

In this paper, we evaluated the performance of different classification algorithms in order to find the classifier model with the best classification accuracy and at the same time with less execution time. The UNSW-NB15 dataset, a recent public dataset for Network Intrusion Detection Systems, was employed to evaluate the performance of different detection algorithms using Apache Spark. The task of the detection algorithm was to classify whether the incoming network traffic was normal or an attack, based on all features that described every pattern of network traffic. We concluded that Random Forest classifier gave the best performance in term of accuracy, sensitivity, specificity and execution time. Followed by Decision Tree and Naïve Bayes gave at the worst detection accuracy with 74.19%. In future work we aim to reduce the training and detection time, by applying feature selection techniques in order to reduce the number of features used for detection. We might also develop a detection algorithm to classify the incoming network traffic into one of the nine attack categories.

References

1. S. Peddabachigari, A. Abraham, C. Grosan & J. Thomas, Modeling intrusion de- tection system using hybrid intelligent systems, in Journal of network and computer applications, 30 (2007), p. 114-132.
2. G. Giacinto, R. Perdisci, M. Del Rio & F. Roli, Intrusion detection in computer networks by a modular ensemble of one-class classifiers, Information Fusion, 9 (2008), p. 69–82.
3. S. Chebrolu, A. Abraham & J. P. Thomas, Feature deduction and ensemble design of intrusion detection systems, in Computers & security, 24 (2005), p. 295–307.
4. S.X.Wu&W.Banzhaf, The use of computational intelligence in intrusion detection systems: A review, in Applied Soft Computing, 10 (2010) 1–35.
5. M. Tavallaee, E. Bagheri, W. Lu & A. A. Ghorbani, A detailed analysis of the KDD CUP 99 data set, in Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium, (2009) 1–6.
6. KDD CUP 1999 Data Set Available on: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
7. NSL-KDD Data Set for network-based intrusion detection systems, Available on: http://iscx.ca/NSL-KDD/, 2009.
8. McHugh and John, Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory, in ACM Transactions on Information and System Security (TISSEC), 3 (2000) 262–294.
9. S. Choudhury & A. Bhowal, Comparative analysis of machine learning algorithms along with classifiers for network intrusion detection, Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM), 2015 International Conference, (2015) 89–95.
10. B. A. Tama & K. H. Rhee, A combination of PSO-based feature selection and tree-based classifiers ensemble for intrusion detection systems, in Advances in Computer Science and Ubiquitous Computing, (2015) 489–495.
11. N. Moustafa and J. Slay, UNSW-NB15: a comprehensive dataset for network intrusion detection systems (UNSW-NB15 network data set), in Military Communications and Information Systems Conference, 2015.
12. N. Moustafa & 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, in Information Security Journal: A Global Perspective, 25 (2016) p.18–31.
13. UNSW-NB15 data set. Available on: http://www.cybersecurity.unsw.adfa.edu.au/ADFA%20NB15%20Datasets/, 2015.
14. I. J. Good, I. Hacking, R. C. Jeffrey & H. Trnebohm, The estimation of probabilities: an essay on modern Bayesian methods, MIT Press, 1965.
15. P. Langley, W. Iba & K. Thompson, Ananalysis of Bayesian classifier, in Proceedings of the 10th national Conference on Artificial

Intelligence, 1992, 223–228.
16. R. Kohavi, Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid, KDD, 96 (1996) p. 202–207.
17. L. Breiman, J. H. Friedman, R. A. Olshen & C. J. Stone, Classification and regression trees. Wadsworth & Brooks, Monterey, CA, 1984.
18. L. Breiman, Random forests, Machine learning, 45, (2001) p. 5–32.
19. A. Liaw & M. Wiener, Classification and regression by randomForest, R news, 2 (2002).
20. J. R. Quinlan, C4. 5: programs for machine learning, 2014.
21. H. Karau, A. Konwinski, P. Wendell & M. Zaharia. Learning spark: lightning-fast big data analysis. " O'Reilly Media, Inc.". (2015)
22. Apache Spark MLlib, http://spark.apache.org/docs/latest/mllib-guide.html
23. Boser, Bernhard E., Isabelle M. Guyon, and Vladimir N. Vapnik. "A training algorithm for optimal margin classifiers." Proceedings of the fifth annual workshop on Computational learning theory. ACM, 1992.
24. Zhu, Guobin, and Dan G. Blumberg. "Classification using ASTER data and SVM algorithms;: The case study of Beer Sheva, Israel." Remote sensing of Environment 80.2 (2002): 233-240.