Building a System to Detect the Intrusions over Network using Machine Learning Algorithms

[1]Raghu Bhaskar Ganteela,[2]D Srinivas

[1]Student,[2]Assistant Professor

[1,2]Department of CSE, KIET, Korangi, AndhraPradesh.

Abstract : Recently, the huge amounts of data and its incremental increase have changed the importance of information security and data analysis systems for Big Data. Intrusion detection system (IDS) is a system that monitors and analyzes data to detect any intrusion in the system or network. High volume, variety and high speed of data generated in the network have made the data analysis process to detect attacks by traditional techniques very difficult. Big Data techniques are used in IDS to deal with Big Data for accurate and efficient data analysis process. Most techniques used in today's IDS are not able to deal with the dynamic and complex nature of cyber-attacks on computer networks. Hence, efficient adaptive methods like various techniques of machine learning can result in higher detection rates, lower false alarm rates and reasonable computation and communication costs. Using a set of benchmark data from a KDD (knowledge discovery and data mining) competition designed by DARPA, we demonstrate that efficient and accurate classifiers can be built to detect intrusions. We compare the performance of Decision Tree based, Random Forest based and support vector machine based, systems for intrusion detection.

Keywords **- Intrusion Detection, Big Data, Apache Spark, Support Vector Machine (SVM), NSL-KDD.**

I. INTRODUCTION

The profusion of the e-devices and extensive reliance on web based applications for both, our regular activities, as well as high-profile use has led to ever increasing network/internet communication. This has led to the generation of a tremendous amount of traffic data at a very fast rate, posing some serious challenges for safe and reliable use of the internet by individuals and institutions. As per the available data, over the years there has been a tremendous increase in network traffic and a corresponding rise in network intrusion or cyber-attacks. Based on the Cisco reports, the size of the global internet traffic will reach zettabytes (1021) by the year 2016 and twice by the end of the year 2019.

Cyber-attack or network intrusion is an activity which tries to compromise the normal functioning of a computer network. To neutralize cyber-attacks, we have to develop a mechanism called intrusion detection, which is a method to mitigate or report these intrusions. However, it becomes difficult to monitor and identify intrusions at very high network speed and moreover, in the event of an outbreak of Distributed Denial of Service (DDoS) attacks, these issues aggravate exponentially. Therefore, it becomes imperative on the part of organizations to equip themselves against imminent network attacks. With legacy intrusion detection methods, we have struggled to keep a watch on the networks efficiently. To overcome these challenges, in the recent years, there have been various attempts to propose efficient Intrusion Detection System (IDS). IDS is an application t hat monitors, detects, and prevents the network or the system against any suspicious activity of harming network's Confidentiality, Integrity, and Availability (CIA) properties.

Big Data is the data that are difficult to store, manage, and analyze using traditional database and software techniques. Big Data includes high volume and velocity, and also variety of data that needs for new techniques to deal with it. Intrusion detection system (IDS) is hardware or software monitor that analyzes data to detect any attack toward a system or a network. Traditional intrusion detection system techniques make the system more complex and less efficient when dealing with Big Data, because its analysis properties process is complex and take a long time. The long time it takes to analyze the data makes the system prone to harms for some period of time before getting any alert [1, 2]. Therefore, using Big Data tools and techniques to analyze and store data in intrusion detection system can reduce computation and training time.

The IDS has three methods for detecting attacks; Signature-based detection, Anomaly-based detection, and Hybrid-based detection. The signature-based detection is designed to detect known attacks by using signatures of those attacks. It is an effective method of detecting known attacks that are preloaded in the IDS database. Therefore, it is often considered to be much more accurate at identifying an intrusion attempt of known attack [3]. However, new types of attack cannot be detected as its signature is not presented; the databases are frequently updated in order to increase their effectiveness of detections [4]. To overcome this problem Anomaly-based detection that compares current user activities against predefined profiles is used to detect abnormal behaviors that might be intrusions. Anomaly-based detection is effective against unknown attacks or zero-day attacks without any updates to the system. However, this method usually has high false positive rates [5, 6]. Hybrid-based detection is a combination of two or more methods of intrusion detection in order to overcome the disadvantages in the single method used and obtain the advantages of two or more methods that are used. Many researches proposed machine learning algorithm for intrusion detection to reduce false positive rates and produce accurate IDS. However, to deal with Big Data, the machine learning traditional techniques take a long time in learning and classifying data. Using Big Data techniques and machine learning for IDS can solve many challenges such as speed and computational time and develop accurate IDS. The objective of this paper is to introduce Spark Big Data techniques that deal with Big Data in IDS in order to reduce computation time and achieve effective

classification. For this purpose, we propose an IDS classification method named Spark-SVM. Firstly, a preprocessing method is used to convert the categorical data to numerical data and then the dataset is standardization for the purpose of improving the classification efficiency. Secondly, SVM is used for the data classification. More specifically, we introduce comparison between SVM classifier, Decision Tree and Random Forest classifier on Apache Spark Big Data platform based on area under curve (AUROC), Area Under Precision-Recall curve (AUPR) and time metrics. The KDDCUP99 are tested in this study.

The rest of this work is organized as follows: A review of relevant works is conducted in "Related works" section. In "Methods" section, we introduced the proposed method. Also, each step in this method is described. Results and experiment settings are mentioned in "Result and discussion" section. Finally, we conclude our work and describe the future work in "Conclusion" section.

Description of NSL-KDD

The KDD 1999 dataset was developed by the MIT Lincoln Labs [10] and was extensively used by researchers during the last decade. The entire dataset is very large in size and contains many attributes variables. Therefore to improve the machine learning computation, 10 % of it was extracted and adopted as training dataset in the intrusion detection process. However, some inherent drawback was made about this dataset. The KDD 99 contains important quantities of redundant records which has as consequence to prevent the learning algorithm to perform well. In addition, duplicate records found in the test dataset cause the evaluation result to be biased by the method used during the detection rates results. To resolve some issues found in the previous KDD 99, an improved version was created, the NSL KDD dataset which can be available at [11]. The reason behind the use of this dataset has been reported at [9] among them the following are relevant to mention:

• Elimination of redundant records in the training set will help our classifier to be unbiased towards more frequent records.
• No presence of duplicate records in the test set, therefore, the classifier performance will not be biased by the techniques which have better detection rates on the frequent records.
• The training and test set contains both a reasonable numbers of instances which is affordable for experiments on the entire set without the need to randomly choose a small portion.

The NSL KDD dataset contains four main files as describe in the Table 2

TABLE 1 NSL KDD Dataset Description

Name of the Files Description	Description
KDDTrain+.TXT	It is the full training set including attack-type labels and difficulty level in csv Format
KDDTest+.TXT	It is the full test set including attack-type labels and difficulty level in csv format
KDDTrain+_20	Percent.TXT20% subset of the KDDTrain+.txt
KDDTest-21.TXT	A subset of the KDDTest+.txt file which does not include records with difficulty level of 21 out of 21

In this paper, the KDDTain+.TXT and the KDDTest+.TXT which consists of 126,620and 22,850records respectively were used. The training and test set contain both 41 features labeled as normal traffic or specific attack types, all these features are subdivided in 4 categories [12][13]: basic features, time-based traffic features, content features and host-based traffic features. All categories are described below: Basic features: It contains all features which derived from TCP/IP connection such as Protocol_type, Service, duration and etc. Time-based traffic features: It is used to capture those features which are mature over a 2 second temporal window (e.g. count, srv_count, Rerror_rate and etc.) Content features: Those features use domain knowledge to access the payload of the original TCP packets(e.g. hot, num_root, is_guest_login and etc.) Host-based traffic features: all attacks which span longer than 2 second intervals that have the same destination host as the current connection are access using these features (e.g. dst_host_count, dst_host_srv_count and etc.) The classes or labels in the NSL KDD dataset are divided into four categories which represent the attack class and one as normal traffic [12]:

1) Denial of Service (DoS): This attack aims to block or restrict a computer system or network resources or services.
2) Probe: here the intruder aims to scan for information or vulnerabilities in a network or computer system which later on will be used to launch attacks.
3) Remote to Local (R2L): Here the intruder gain remotely unauthorized access to a computer system over a network by sending data packet to that system.
4) User to Root (U2R): Here the intruder gains access to a user with normal privilege and later on try to access a user with administrator or root privilege.

TABLE 2 KDD99 Dataset Attributes

No	Attribute name	No	Attribute name
1	Duration	22	Is_guest_login
2	Protocol_type	23	Count
3	Service	24	Serror_rate
4	Src_bytes	25	Rerror_rate
5	Dst_bytes	26	Same_srv_rate
6	Flag	27	Diff_srv_rate
7	Land	28	Srv_count
8	Wrong_fragment	29	Srv_serror_rate
9	Urgent	30	Srv_rerror_rate
10	Hot	31	Srv_diff_host_rate
11	Num_failed_logins	32	Dst_host_count
12	Logged_in	33	Dst_host_srv_count
13	Num_compromised	34	Dst_host_same_srv_rate
14	Root_shell	35	Dst_host_diff_srv_rate
15	Su_attempted	36	Dst_host_same_src_port_rate
16	Num_root	37	Dst_host_srv_diff_host_rate
17	Num_file_creations	38	Dst_host_serror_rate
18	Num_shells	39	Dst_host_srv_serror_rate
19	Num_access_files	40	Dst_host_rerror_rate
20	Num_outbound_cmds	41	Dst_host_srv_rerror_rate
21	Is_hot_login	42	Class

II. RELATED WORKS

There are many types of researches introduced for intrusion detection system. With emerge of Big Data, the traditional techniques become more complex to deal with Big Data. Therefore, many researchers intend to use Big Data techniques to produce high speed and accurate intrusion detection system. In this section, we show some researchers that used machine learning Big Data techniques for intrusion detection to deal with Big Data.

Ferhat et al. [7] used cluster machine learning technique. The authors used k-Means method in the machine learning libraries on Spark to determine whether the network trafc is an attack or a normal one. In the proposed method, the KDD Cup 1999 is used for training and testing. In this proposed method the authors didn't use feature selection technique to select the related features.

Peng et al. [8] proposed a clustering method for IDS based on Mini Batch K-means combined with principal component analysis (PCA). The principal component analysis method is used to reduce the dimension of the processed dataset and then mini batch K-means++ method is used for data clustering. Full KDDCup1999 dataset has been used to test the proposed model.

Peng et al. [9] used classifcation machine learning technique. The authors proposed an IDS system based on decision tree over Big Data in Fog Environment. In this proposed method, the researchers introduced preprocessing algorithm to fgure the strings in the given dataset and then normalize the data to ensure the quality of the input data so as to improve the efciency of detection. They used decision tree method for IDS and compared this method with Naïve Bayesian method as well as KNN method. Te experimental results on KDDCUP99 dataset showed that this proposed method is efective and precise.

Belouch et al. [10] evaluated the performance of SVM, Naïve Bayes, Decision Tree and Random Forest classifcation algorithms of IDS using Apache Spark. The overall performance comparison is evaluated on UNSW-NB15 dataset in terms of accuracy, training time and prediction time.

Also, Manzoor and Morgan [11] proposed real-time intrusion detection system based on SVM and used Apache Storm framework. The authors used libSVM and C-SVM classifcation for intrusion detection. The proposed approach was trained and evaluated on KDD 99 dataset. In addition, Features selection techniques were used in a lot of researches.

PCA Features selection technique implemented in some proposed IDSs like Vimalkumar and Randhika [12] proposed Big Data framework for intrusion detection in smart grid by using various algorithms like a Neural Network, SVM, DT, Naïve Bayes and Random Forest. In this approach, a correlation-based method is used for feature selection and PCA is used for dimensionality reduction. The proposed approach aimed to minimize the time of predicting attack and also to increase the accuracy of the classifcation task. This approach used Synchrophasor dataset for training and evaluation. The results of this proposed approach are compared by accuracy rate, FPR, Recall and specificity evaluation metrics.

Dahiya and Srivastava [13] proposed a framework for fast and accurate detection of intrusion using Spark. In the proposed framework was used Canonical Correlation Analysis (CCA) and Linear Discriminant Analysis (LDA) algorithms for feature reduction, and seven classifcation algorithms(Naïve Bayes, REP TREE, Random Tree, Random Forest, Random Committee, Bagging and Randomizable Filtered). In the proposed work the two sets of UNSW-NB 15 dataset was used to evaluate the performance of all classifiers. The experiment result of the proposed method found the LDA and random tree algorithm approach is more effective and fast. The Results showed that AUROC = 99.1 for dataset1 and 97.4 for dataset2. In our model, we obtained the results of AUROC = 99.55. Terefore, our model is more effective and fast.

Hongbing Wang et al. [14] proposed a parallel principal component analysis (PCA) combined with parallel support vector machine (SVM) algorithm based on the Spark platform (SP-PCA-SVM). PCA is used for analyzing data and feature extract for dimensionality reduction based on Bagging. Te proposed approach used KDD99 for training and evaluation.

Natesan et al. [15] proposed optimization algorithm for feature selection. The authors proposed Hadoop based parallel Binary Bat algorithm method for intrusion detection. In this approach, the authors used parallel Binary Bat algorithm for efficient feature selection and optimized detection rate. The MapReduce of Hadoop is used to improve computational complexity and parallel Naïve Bayes provides a cost-effective classification. The proposed approach was trained and evaluated on KDD99 dataset. The proposed approach displayed that the detection rate is improved and the detection time is reduced.

Table 1 shows differences between related works based on the Big Data tool that were used for developing the work and the machine learning algorithm that were used as a classifier in the work and the dataset that has been used to train and evaluate. The researchers are still seeking to find an efective way to detect the intrusions with high performance, high speed and a low of false positive alarms rate. The main objective of this paper is to improve the performance and speed of intrusion detection within Big Data environment. In this method, the researchers used Apache Spark Big Data tools because it is 100 times faster than Hadoop [16], the feature selection that takes the amount of computation time, and this time can be reduced when using SVM on KDD datasets [17]. Therefore, we used SVM algorithm and compared it with Decision Trees and Random Forest classifier based on area under curve (ROC), Area Under Precision Recall Curve and time metrics.

TABLE 3 Related Work Comparative

Related work	Big Data tool	Algorithm	Dataset
[7]	Apache Spark	K-Means	KDD 99
[8]	Anaconda	K-Means++	KDD 99
[9]	Anaconda	Decision tree	KDD 99
[10]	Apache Spark	SVM, Naïve Bayes, Decision Tree and Random Forest	UNSW-NB 15
[11]	Apache Storm	C-SVM	KDD 99
[12]	Apache Spark	Neural network, SVM, DT, Naïve Bayes and Random forest	Synchrophasor
[13]	Apache Spark	Naïve Bayes, REP TREE, Random Tree, Random Forest, Random Committee,Bagging.	UNSW-NB 15
[14]	Apache Spark	SVM	KDD 99
[15]	Hadoop	Parallel Naïve Bayes	KDD 99

III. METHOD

Spark Chi SVM Proposed Model

In this section, the researchers describe the proposed model and the tools and techniques used in the proposed method. Figure (1) shows Spark-Chi-SVM model. The steps of the proposed model can be summarized as follows:

1 Load dataset and export it into Resilient Distributed Datasets (RDD) and Data Frame in Apache Spark.
2 Data preprocessing
3 Feature selection
4 Train Spark-Chi-SVM with the training dataset.
5 Test and evaluate the model with the KDD dataset.

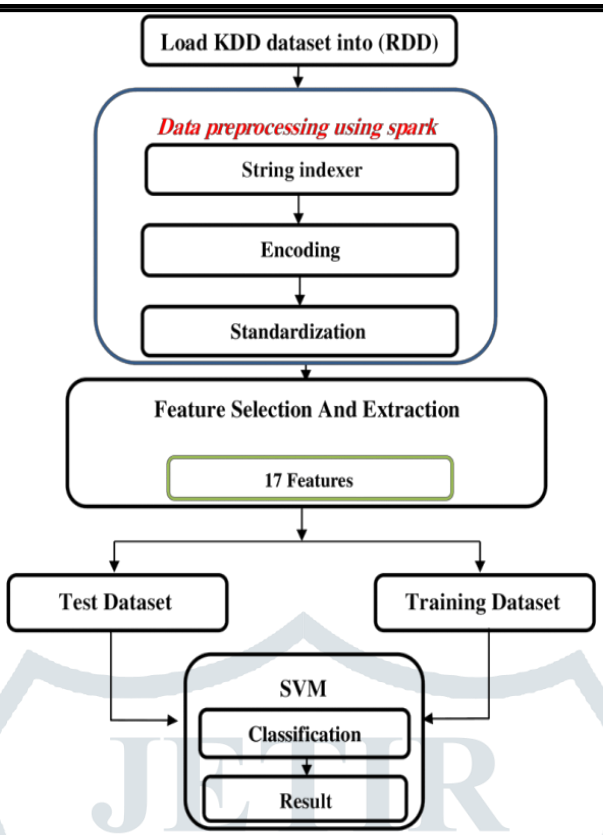

Fig. 1 Spark-Machine Learning model. The sequence of steps that in Spark-Machine Learning model

Apache Spark

Spark [16] is a fast and general-purpose cluster computing system for large-scale in-memory data processing. Spark has a similar programming model to Map-Reduce but extends it with a data-sharing abstraction called Resilient Distributed Datasets or RDD [18]. A Spark was designed to be fast for iterative algorithms, support for in-memory storage and efficient fault recovery. Spark Core consists of two APIs which are the unstructured and structured APIs [19]. The unstructured API is RDDs, Accumulators, and Broadcast variables. The structured API consists of Data-Frames, Datasets, Spark SQL, and it is the interface that most users should use. In this work the dataframe structure and RDD are used. Dataframe used to load and store the dataset, then it converted to RDD for processing by other process. Spark runs up to 100 times faster than Hadoop in certain environments [18]. Spark can be run with its standalone cluster mode, on Hadoop YARN, or on Apache Mesos or on EC2. In our model we use Spark standalone cluster mode. The main components of Apache Spark are Spark core, SQL, Streaming, MLlib, and GraphX. Figure 2 illustrates Spark on Hadoop ecosystem and it's main components.

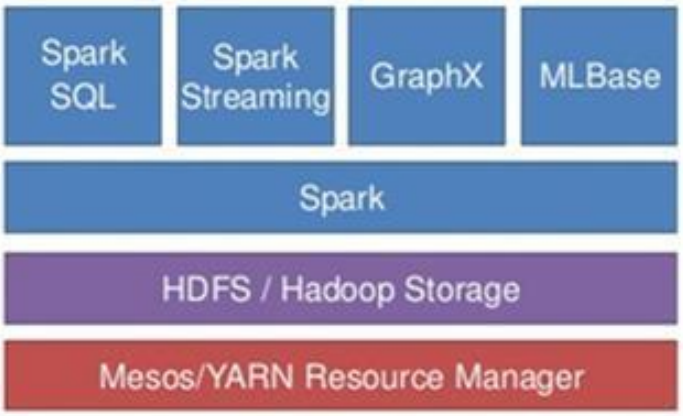

Fig. 2 Spark ecosystem and components. Spark ecosystem on Hadoop and Spark main components

Spark uses a master/slave architecture illustrated in Fig. 3. There is a driver that talks to a single coordinator called master that manages workers in which executors run. A Spark cluster has a single master and any number of slaves/workers.

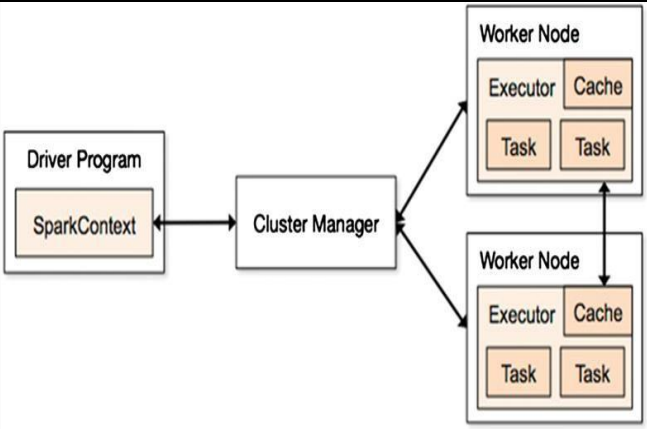

Fig. 3 Spark-architecture-official. Spark master/slave architecture

Classification

The term SVM is typically used to describe classification with support vector methods and support vector regression is used to describe regression with support vector methods. SVM (Support Vector Machine) is a useful technique for data classification. The classification problem can be restricted to consideration of the two-class problem without loss of generality. In this problem the goal is to separate the two classes by a function which is induced from available examples. The goal is to produce a classifier that will work well on unseen examples, i.e. it generalizes well. Consider the example in figure 4. Here there are many possible linear classifiers that can separate the data, but there is only one that maximizes the margin (maximizes the distance between it and the nearest data point of each class). This linear classifier is termed the optimal separating hyper plane. Intuitively, we would expect this boundary to generalize well as opposed to the other possible boundaries.

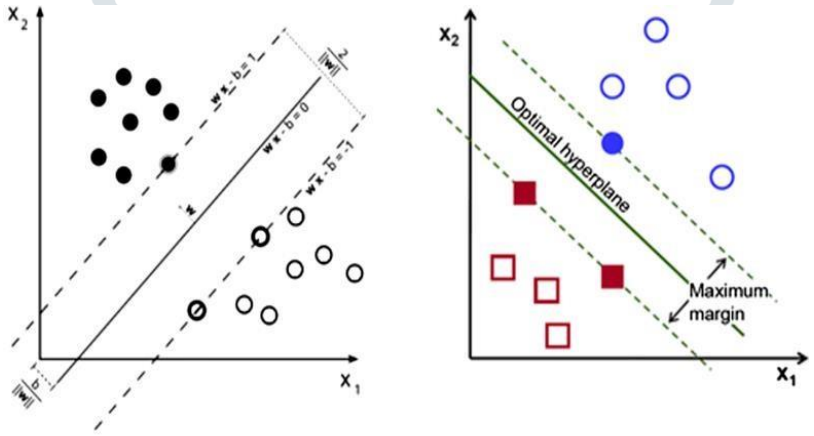

Fig. 4 SVM hyper plane. SVM hyper plane is separate the data into two classes

IV. RESULTS

After the creation of the training models, the next step is the testing phase process implementation. In order to implement a fair testing phase fully randomized 60000 have been extracted. The extracted testing data includes all 21 types of attacks within KDD dataset. There are several evaluations metrics can be used in a classification algorithm. In this paper, the confusion matrixes were generated for each machine learning classifiers. It includes significant information about existing and predicted output classes. Furthermore, the following performance metrics are computed [12].

- **True Positive (TP):** this value represents the correct classification attack packets as attacks.
- **True Negative (TN):** this value represents the correct classification normal packets as normal.
- **False Negative (FN):** this value illustrates that an incorrectly classification process occurs. Where the attack packet classified as normal packet, a large value of FN presents a serious problem for confidentiality and availability of network resources becau se the attackers succeed to pass through intrusion detection system.
- **False Positive (FP):** this value represents incorrect classification decision where the normal packet classified as attack, the increasing of FP value increases the computation time but; on the other hand, it is considered as less than harmful of FN value increasing. • Precision: is one of the primary performance indicators. It presents the total number of records that are correctly classified as attack divided by a total number of records classified as attack. The precision can be calculated according to the following equation:

$$P = \frac{TP}{(TP + FP)} \quad (1)$$

In addition, the number of both the correctly and the incorrectly classified instances are recorded with respect to the time taken for proposed training model. During the testing phase, the following parameters were applied for the machine learning classifiers. Random forest classifier also tested with number of trees =100 and seed =1. Random tree classifier was tested with min variance = 0.001 and seed = 1.

Table IV presents the TP rate and the Precision values of the selected classifiers in the experiments. It can be concluded thatthe random forest classifier achieved highest TP rate of 93.1% while the random tree classifier achieved the lowest TP rate of 90.6%. In other words, random tree classifier reached the lowest value of attacks classification process. Form another perspective, the decision table classifier reached the lowest precision value of 94.4% and that indicates the decision table classifier suffers of an increasing false positive value. Therefore, there are a large number of normal packets classified as attack packets.

TABLE 4 True Positive Rate and Precision Ratios

Machine Learning Classifiers	TP Rate	Precision
Decision Tree	92.4	0944
Random forest	90.6	0.992
Support Vector Machine	99.3	0.998

Regarding Table VI that Support Vector Machine classifier recorded the highest value 0.999 based on ROC value while Random Forest classifier presented as lowest value 0.953. Furthermore, Decision Tree classifier had the lowest value 0.0682 based on RMSE indicator while the decision table presented as highest value 0.0903. Through the testing and classification of 60000 instances of records from the KDD dataset, the total number of incorrectly classified records for each selected classifiers are presented in the Table VI. The average accuracy rate is calculated by the following formula:

$$AverageAccuracyRate = \frac{TP+TN}{TP+FN+FP+TN} \quad (2)$$

Machine Learning Classifiers	Correctly classified Instances	Incorrectly classified Instances	Accuracy Rate
Random Forest	9649	325	97.92%
Decision tree	9711	2885	90.57%
SVM	69175	426	98.60%

V.CONCLUSION

In this paper, we introduced Spark- SVM model for intrusion detection that can deal with Big Data. The proposed model used Spark Big Data platform which can process and analyze data with high speed. Big data have a high dimensionality that makes the classification process more complex and takes a long time. Therefore, in the proposed model, the researchers used Random Forest to select related features and DT,RF and SVM to classify data into normal or attack. The results of the experiment showed that the model has high performance and speed. In future work, the researchers can extend the model to a multi-classes model that could detect types of attack.

REFERENCES

[1]. Tchakoucht TA, Ezziyyani M. Building a fast intrusion detection system for high-speed-networks: probe and DoS attacks detection. Procedia Comput Sci. 2018;127:521–30

[2]. Zuech R, Khoshgoftaar TM, Wald R. Intrusion detection and big heterogeneous data: a survey. J Big Data. 2015;2:3.

[3]. Sahasrabuddhe A, et al. Survey on intrusion detection system using data mining techniques. Int Res J Eng Technol. 2017;4(5):1780–4.

[4]. Dali L, et al. A survey of intrusion detection system. In: 2nd world symposium on web applications and networking (WSWAN). Piscataway: IEEE; 2015. p. 1–6.

[5]. Scarfone K, Mell P. Guide to intrusion detection and prevention systems (idps). NIST Spec Publ. 2007;2007(800):94.

[6]. Debar H. An introduction to intrusion-detection systems. In: Proceedings of Connect, 2000. 2000.

[7]. Ferhat K, Sevcan A. Big Data: controlling fraud by using machine learning libraries on Spark. Int J Appl Math Electron Comput. 2018;6(1):1–5.

[8]. Peng K, Leung VC, Huang Q. Clustering approach based on mini batch Kmeans for intrusion detection system over Big Data. IEEE Access. 2018.

[9]. Peng K. et al. Intrusion detection system based on decision tree over Big Data in fog environment. Wireless Com- mun Mob Comput. 2018. https://doi.org/10.1155/2018/4680867.

[10]. Belouch M, El Hadaj S, Idhammad M. Performance evaluation of intrusion detection based on machine learning using Apache Spark. Procedia Comput Sci. 2018;127:1–6.

[11]. Manzoor MA, Morgan Y. Real-time support vector machine based network intrusion detection system using Apache Storm. In: IEEE 7th annual information technology, electronics and mobile communication conference (IEMCON), 2016. Piscataway: IEEE. 2016; p. 1–5.

[12]. Vimalkumar K, Radhika N. A big data framework for intrusion detection in smart grids using Apache Spark. In: Inter- national conference on advances in computing, communications and informatics (ICACCI), 2017. Piscataway: IEEE; 2017. p. 198–204.

[13]. Dahiya P, Srivastava DK. Network intrusion detection in big dataset using Spark. Procedia Comput Sci. 2018;132:253–62.

[14]. Wang H, Xiao Y, Long Y. Research of intrusion detection algorithm based on parallel SVM on Spark. In: 7th IEEE Inter- national conference on electronics information and emergency communication (ICEIEC), 2017 . Piscataway: IEEE; 2017. p. 153–156.

[15]. Natesan P, et al. Hadoop based parallel binary bat algorithm for network intrusion detection. Int J Parallel Program. 2017;45(5):1194–213.

[16]. https://spark.apache.org.

[17]. Akbar S, Rao TS, Hussain MA. A hybrid scheme based on Big Data analytics using intrusion detection system. Indian J Sci Technol. 2016. https://doi.org/10.17485/ijst/2016/v9i33/97037

[18]. Zaharia M, et al. Apache spark: a unifed engine for big data processing. Commun ACM. 2016;59(11):56–65.

[19]. Chambers MZaB. Spark: The Defnitive Guide: O?Reilly Media, Inc. , 1005 Gravenstein Highway North, Sebastopol, CA 95472. 2017.

[20]. Kato K, Klyuev V Development of a network intrusion detection system using Apache Hadoop and Spark. In: IEEE conference on dependable and secure computing, 2017 .Piscataway: IEEE. 2017; p. 416–423.

[21]. Deng Z, et al. Efcient kNN classifcation algorithm for big data. Neurocomputing. 2016;195:143–8.

[22]. Sung AH, Mukkamala S. The feature selection and intrusion detection problems. In: ASIAN. Berlin: Springer; 2004. p. 468–482.

[23]. Cortes C, Vapnik V. Support-vector networks. Mach Learn. 1995;20(3):273–97.

[24]. Cherkassky V, Ma Y. Practical selection of SVM parameters and noise estimation for SVM regression. Neural Netw. 2004;17(1):113–26. https://doi.org/10.1016/S0893-6080(03)00169-2.

[25]. Karamizadeh S. et al. Advantage and drawback of support vector machine functionality. In: International conference on computer, communications, and control technology (I4CT), 2014. Piscataway: IEEE. 2014; p. 63–65.

[26]. Enache A-C, Sgârciu V. Enhanced intrusion detection system based on bat algorithm-support vector machine. In: 11th international conference on security and cryptography (SECRYPT), 2014 . Piscataway: IEEE; 2014. p. 1–6.

[27]. Bhavsar H, Ganatra A. A comparative study of training algorithms for supervised machine learning. Int J Soft Com- put Eng (IJSCE). 2012;2(4):2231–307.

[28]. Bradley AP. The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognit. 1997;30(7):1145–59.

[29]. http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html

[30]. Gupta GP, Kulariya M. A framework for fast and efcient cyber security network intrusion detection using Apache Spark. Procedia Comput Sci. 2016;93:824–31.

[31]. Kulariya M. et al. Performance analysis of network intrusion detection schemes using Apache Spark. In: International conference on communication and signal processing (ICCSP), 2016. Piscataway: IEEE; 2016. p. 1973–1977.