International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 9, No.6 (Nov-2020)

http://dx.doi.org/10.12785/ijcds/0906015

An Efficient Spark-Based Network Anomaly Detection

Djediden Mohamed Seghire Othman[1], Reguieg Hicham[1]and Mekkakia Maaza Zoulikha[1]

[1] *Laboratoire SIMPA, Université des Sciences et de la Technologie d'Oran Mohamed Boudiaf, USTO-MB, Oran, Algeria*

Received 26 Mar. 2020, Revised 14 May 2020, Accepted 21 May 2020, Published 1 Nov. 2020

Abstract: Nowadays, with the high volume of captured data in computer networks the anomaly detection has become one of the main challenges. To deal with this some works have used machine learning algorithms and feature selection methods with traditional tools that are not dedicated to big data analysis, other works have used machine learning algorithms on big data frameworks without the feature selection methods application. In this paper, we propose an approach that aims to detect network intrusion with higher accuracy, using the minimum of features and supporting massive data. This approach combines the machine learning algorithms, the feature selection methods, and the Spark framework. For experimentation, we use the UNSW-BN15 dataset. The obtained results and the carried comparisons show that the proposed approach provides better accuracy using a small subset of features.

Keywords: Dataset, Intrusion Detection, Machine Learning, Feature Selection, Apache Spark.

1. INTRODUCTION

Cyber security is a combination of several processes and technologies designed to keep computers, networks, and data secure against all kinds of attacks and unauthorized access. This contains antivirus, firewalls, and intrusion detection systems (IDS) [1]. IDS protects and defends information systems against attack, misuse, duplication, alteration, and destruction. There are two main types of detection in support of IDS, (i) Misuse-based: By using the signatures this type is the most suitable for the detection of the known attacks, thus it avoids the generation of a high number of false alarm, but it requires regular updates of its database of signatures also it does not detect new attacks. (ii) Anomaly-based: This mode creates normal network behaviour and identifies anomalies as deviations from normal behaviour. They can detect zero-day attacks, but their disadvantage is the possibility of high rates of false alarms (FAR) because the behaviours of invisible (but legitimate) systems will be classified as anomalies [2].

With the explosion in the number of computer network users (Internet in particular), the captured data became more varied and voluminous (Big Data), making the intrusion detection process using traditional methods more difficult and complicated. That is why Big Data techniques are used in IDS to develop more accurate and efficient intrusion detection frameworks

The US National Institute of Standards and Technology (NIST) defines Big Data as "data whose data volume, acquisition speed, or data representation limits the ability to use traditional relational methods to perform an effective analysis ", this indicates that effective methods or technologies need to be developed and used to analyse and process Big Data.

Irregularities, complexity, size, and volume of data are often considered the main challenges of data processing. The data captured in the computer networks are generally of a high dimensionality, which makes the analysis and detection process of attack very long and complicated. In machine learning, Feature selection (FS) is a crucial method in the data pre-processing stage. Feature selection reduces the dimensionality of data and enhances the performance of the classification process.

The most notable examples of Big Data Processing Framework are Spark and Hadoop Map Reduce, Spark is a distributed processing system known for its speed. This versatile framework covers a wide range of big data workloads like iterative algorithms, batch applications, streaming and interactive queries. Spark offers a library dedicated to machine learning named MLlib [3-5].

Many existing works have used machine-learning classifier for intrusion detection in the undistributed environment among them works that included FS methods in order to deal with the massive volume of data [6-12]. Other works have used Apache Spark and its ML library to create distributed IDSs, the Spark ML library users do not have enough choice to apply FS method because this library only contains the Chi-Squared selector [13-21].

Datasets are needed to train and evaluate anomaly-based network intrusion detection systems. Some works

E-mail: mohamed.djediden@univ-usto.dz, hicham.reguieg@univ-usto.dz, zoulikha.mekkakia@univ-usto.dz*

have used real data captured from switches and tools that generate attacks to test their system but the inconvenient is the impossibility of comparing the results obtained with other works because when the data are different the results will also be different [20,21]. That is why many public network datasets are available. The best known are DARPA98, DARPA99, KddCup99, CTU-13, and they are used in a lot of work [15-19]. This set contains a considerable number of redundant or missing records, so they are quite old, computer networks and attacks have changed a lot since then. To address these challenges, [22] has created a new dataset called UNSW-NB15 and since then many projects have shown that UNSW-NB15 is more complex than other datasets and that this dataset can be used to evaluate reliably existing and new methods of IDS [6-11,13,14,22].

In this paper, we propose an efficient approach that aims to combine machine-learning algorithms, feature selection methods based on feature importance and the Spark framework to develop a distributed attack detection system that selects the best subset of features ensuring higher accuracy. The Proposed approach will be tested and evaluated on the UNSW-NB15 dataset.

The main contributions of our work can be summarized as follows:

1) We reduce the dimension of the network dataset to the minimum of features while ensuring better accuracy of intrusion detection.

2) Our approach overcomes the disadvantages of existing solutions such as the non-support of massive data, the limit (few choices) of Apache Spark in terms of feature selection and also ensured the high availability which is absent in most of the existing ids because it is undistributed.

3) We prove that our proposed approach is more effective and suitable compared to existing works.

For this purpose, this paper is organized as follows. In section 2, we introduce some related works on the application of ML, FS, and Apache Spark for IDS. In section 3, after the description of the chosen dataset (UNSW-NB15), we introduced the proposed approach. As well as, each step in this method is described. Section 4 presents the proposed approach results. Finally, we conclude our work and describe future work in section 5.

2. RELATED WORKS

In this section, we present works that apply ML, FS, and Apache Spark for IDS. Different works apply automatic learning algorithms and selection methods to create undistributed IDS. Next, other works combines the machine learning classifiers with the Spark framework for distributed IDS.

A. ML and FS for undistributed IDS

Several works describe the use of the ML classifiers and FS methods for intrusion detection. Buczak and Guven [1] present a detailed survey of the use of ML algorithms for ids but not cite any work that uses UNSW-NB15 as evaluation data.

Anwer et al. [6] use different FS strategies and two ML classifiers (Decision Tree (DT) “J48 version” and Naïve Bayes (NB)) on UNSW-NB15 dataset to select the minimum number of features that achieve the highest accuracy. The approach is developed on Weka [23] and the experimental results show that the best strategy is by using the Gain Ratio (GR) selector method and DT J48 as a classifier. Divekar et al. [7] propose an approach that focuses on data pre-processing using the SMOTE oversampling and the random under-sampling technique [24] to make the data more balanced before starting the detection process with ML classifiers (Neural Network (NN), Support Vector Machine (SVM), DT, Random Forest (RF), NB and K-Means) and selection methods from the Scikit-learn library [25]. The authors tested their approach with the KddCup99, NSL-KDD, and UNSW-NB15 datasets and prove that UNSW-NB15 can substitute the archaic KDD CUP 99 dataset and even NSL-KDD when used to train ML anomaly-based IDSs.

Janarthanan and Zargari [8] analyse the UNSW-NB15 data using RF as an ML classifier and exploring significant features to improve intrusion detection. A new subset of features is proposed and compared with the previous work in the KDD'99 dataset. The results are obtained under the Weka framework, the new subset show better intrusion detection rates. In [9], the authors combine the FS algorithm (genetic algorithm-logistic regression (GALR)) with ML classifier (DT, RF, and NB) to produce the optimal subset of features that can be used to classify the instances of KDD99 and UNSW-NB15 datasets. The best subset selection is based on maximizing the accuracy of the classification and minimizing the features number. The approach is developed on Weka and the experimental results show that it makes more sense to use UNSW-NB15 over KddCup99 for the new IDS evaluation because it better represents the current networks.

In [10], authors propose a novel deep learning model comprise of two decision stages: an initial stage responsible for classifying network traffic as normal or abnormal (binary classification), using a probability score value. This stage is then used in the decision stage as an additional feature, for attack categories detection (multi-classification). The MATLAB tool [26] is used for model development and the experiments are done on the KDD99 and UNSW-NB15 datasets. Results show that the proposed approach achieving high recognition rates.

Moustafa and Slay [11] propose a new hybrid features selection method, based on the central points (CP) of attribute values and Association Rule Mining (ARM). The approach aims to reduce the processing time overall by selecting the most frequent values and to choose the highest-ranked features by removing irrelevant or noisy features. For intrusion detection, the expectation-

maximization (EM) clustering, Logistic Regression (LR) and NB classifiers are developed using Visual Studio C# 2008. This approach is applied to UNSW NB15 and NSL-KDD datasets. Experimental results show that the proposed model can improve accuracy and its processing time is extremely short. Moustafa and Slay [12] study the UNSW-NB15 complexity, the data are analysed according to three aspects of the statistical analysis phase, the feature correlation phase, and the complexity evaluation phase. In the third phase, the ML algorithms (DT, LR, NB, Artificial neural networks (ANN) and EM clustering) are used to measure the complexity in terms of accuracy and (FAR) of UNSW-NB15, then the results are compared using the KDD99 dataset, the programming environment is Visual Studio Business Intelligence 2008. This study shows that UNSW-NB15 can be used to evaluate new methods of IDS reliably.

The works cited above provide intrusion detection approaches based on frameworks (environment) that are not dedicated to Big Data. They use a centralized architecture that does not support sharing across machines and does not support functionalities such as scalability, high availability, and real-time analysis.

B. ML and Spark for IDS

Many recent works combine Big Data and ML algorithms to produce a faster and more accurate intrusion detection system. Most of them use Spark as a processing and data analysis framework.

The authors of [27] presented a complete study of the latest research carried out in the use of machine learning for big data processing. After describing existing and recent machine learning methods, the authors highlighted the different challenges of learning from big data and then presented a set of solutions to optimize this process. Then they investigate the connections of machine learning with signal processing techniques for big data processing and cites several open issues. In [28], the authors studied the Apache Spark MLLiB platform. After the detailed presentation of this platform integrated with Apache Spark and dedicated to machine learning, they carried out several real experiments to evaluate the performance of Spark MLLiB compared to the most used traditional machine learning tool (WEKA). The results of the comparisons have demonstrated that the Spark MLLIB platform offers speed, flexibility, fault tolerance, distribution and above all it is very suitable for learning from massive data.

Belouch et al. [13] study the four machine learning algorithms (SVM, NB, DT, and RF) performance using Spark. For the experiment, the authors use the dataset UNSW-NB15 and three metrics of comparison (detection accuracy, building time, and prediction time). The results show that RF is the best algorithm compared to the others. However, this approach uses the complete dataset without any FS method. Dahiya and Srivastava [14] combine FS algorithms (Linear Discriminant Analysis LDA, Canonical Correlation Analysis CCA) and seven well-known classification algorithms in Spark to create fast, efficient and accurate IDS. The authors use a small and large dataset of UNSW NB-15 dataset for performance evaluation of the proposed framework. The results show that random tree (RT) algorithm is better than other algorithms and that feature reduction methods improve the accuracy. However, this work only uses parts of the UNSW-NB15 dataset so it is impossible to compare its results with our proposed approach or with other existing works.

Kamal and Sathyadevan [15] develop a new IDS model using Apache Spark and C.45 DT algorithm to detect Intrusions such as DDoS attack and Port scanning attack. The model is built over Amrita Big Data Apache-Spark framework using DARPA 1998 dataset. The experiment shows that this model is effective for the detection of DDoS and port scanning attacks, however, FS is not used when pre-processing data and authors have tested a single ML classifier (DT), also other network attacks have not been addressed in this work. Lighari and Hussain [16] use different ML algorithms (LR, SVM, NB, DT, RF, and K-Means) in Apache Spark to find the most efficient algorithm in the context of anomaly detection. Using the KddCup99 dataset, the authors compare the training time, prediction time, and the rate of accuracy of each algorithm. The results show that NB is the fastest and K-Means is the most accurate. However, the limitations of this work are the use of a very old dataset and the lack of FS methods.

Othman et al. [17] propose a new attack detection model named Spark-Chi-SVM, this model uses ChiSqSelector as an FS method, SVM as an ML classification algorithm and Spark for data processing. In the experiment, the authors compare between Chi-SVM classifier and Chi-Logistic Regression classifier, the results show that Spark- Chi-SVM has high performance and is efficient for Big Data. Moreover, the authors have not tested other ML classification algorithms known for their performances like RF and DT and the dataset used during the experiments (KddCup99) is very old. In [18], a public data (CTU-13) is analysed with a new unsupervised anomaly detection approach on Apache Spark. The proposed method is the clustering-based from the ML perspective. The results obtained show a high detection rate but this work did not introduce FS methods and did not compare the performance of the used algorithm with any other supervised classification algorithm. Wang et al. [19] propose a new framework that combines a parallel principal component analysis (PCA) and (SVM) algorithm in spark platform. The new framework called (SP-PCA-SVM) is evaluated with KddCup99 data and the experimentation shows that (SP-PCA-SVM) reduces the training time and improves model learning efficiency. However, the limitations of this work are the use of a very old dataset.

Zhou et al. [20] propose an ML-based online DDoS attack detection system using Spark Streaming. The authors use a correlation-based FS method and then compare the detection accuracy of 3 automatic learning methods (NB, LR, and DT). To evaluate their system the authors opt for data captured from a real network and use the BONESI tools to generate DDoS attacks. Experiment results show that the proposed system works well even for large Internet traffic. In [21], the same authors compare the performance of the system proposed previously by using two different frameworks Apache Spark and Apache Flink [29]. Experimental results show that Flink is faster than Spark Streaming when the input stream rate is low. As well when the batch size of Spark Streaming increases the maximum throughput also increases. These two jobs deal only with DDoS attacks, and it is impossible to compare their results with other works because they use data that is real and not public.

TABLE 1. RELATED WORKS COMPARISON USING THE UNSW-NB15 DATASET

Reference	FS	ML Algorithm	ML (Best result)	Tools
Anwer et al. [6]	Different filter and wrapper	DT and NB	DT (J48)	WEKA
Divekar et al. [7]	Gini Impurity Index	NN, SVM, DT, RF, NB and K-Means.	RF	Scikit-learn
Moustafa and Slay [11]	CP and ARM	EM clustering , LR and NB	LR	Visual studio C# 2008
Khan et al. [10]	-They do not use ML -They use Deep learning method			MATLAB
Moustafa and Slay [12]	No	NB, DT, ANN, LR, and EM clustering	DT	Visual Studio Business Intelligence 2008
Khammassi and Krichen [9]	GALR	DT,RF and NB	DT	WEKA
Janarthanan and Zargari [8]	most frequently appeared features in attack	RF	RF	WEKA
Belouch et al. [13]	No	SVM, NB, DT and RF	RF	Spark Mllib
Dahiya and Srivastava [14]	LDA,CCA	NB,RT, RF …	RT	Spark Mllib

Table 1 is a summary comparison between some references in IDS using the UNSW-NB15 dataset as discussed in this section. All the works cited above have used UNSW-NB15 to evaluate the performance of their proposed approaches, but most of them have proposed non-distributed approaches using tools and frameworks that are not dedicated to big data processing. Other works have faced these limitations by integrating the Spark framework and its ML library, but they have not used any method FS, or they have used only a part of the data UNSW-NB15 hence the impossibility to compare their results with related work and with our proposed approach. To deal with all these limitations, we propose an approach that aims to detect network intrusion with a higher accuracy using the minimum of features and supporting massive data, this approach combines the ML algorithms, the FS methods, and the Spark framework and it will be detailed in the next section.

3. PROPOSED APPROACH

This section describes the proposed approach as well as the techniques and dataset used. Fig.1 illustrates the steps in our approach that are: load dataset and export it into Data Frame in Apache Spark, data pre-processing, feature selection, train the model with the training dataset, parameter tuning, and test and evaluate the model with the testing dataset. In what follows, we detail the approach steps. The following workflow describes the steps of our approach.

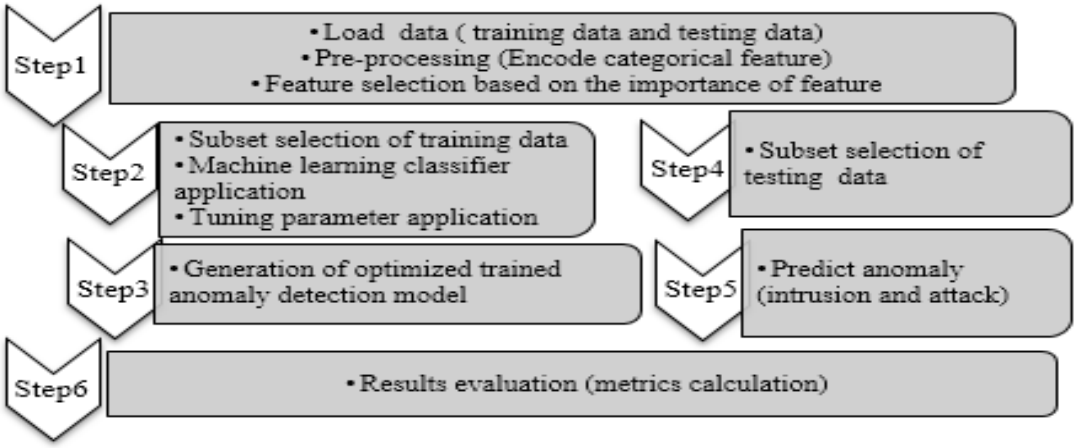

Figure 1. Proposed Approach Workflow

A. Description of UNSW-NB 15 dataset

UNSW-NB 15 datasets were created in 2015 by the IXIA Perfect Storm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS). It contains a hybrid of the realistic modern normal activities and the synthetic contemporary attack behaviours from network traffic. The UNSW-NB15 dataset was decomposed into two partitions Training datasets (#175, 341 records) and a Testing dataset (#82, 332 records) including all different 9 types attack (Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, ShellCode, and Worms) and normal records [22].

The two partitions are available online in Unsw.adfa.edu.au [30]. Both the Training and Testing datasets have 45 features. More precisely 43 features and 2 attributes for labelling this datasets: attack_cat represents the nine categories of the attack and the normal, and label is zero for normal and otherwise one. Table 2 and Fig.2 shows the distribution of normal and attacking instances in training and testing datasets.

In [7] and [12], the authors have proven in their research that UNSW-NB15 data are the most suitable for evaluating the performance of an IDS. They have demonstrated that UNSW-NB15 is uniform, that its skewness is noticeably lower compared to the old dataset and especially that the stationarity of the data is maintained between the training and testing sets (a similar distribution).

In summary, the UNSW-NB15 dataset contains an id feature, 39 numeric features (including the normal or attack label) and four categorical features (including the attack categories label).

TABLE 2. UNSW-NB15 DATASET DISTRIBUTION

Category	Training set size	Training set distribution %	Testing set size	Testing set distribution %
Normal	56000	31,94	37000	44,94
Generic	40000	22,81	18871	22,92
Exploits	33393	19,04	11132	13,52
Fuzzers	18184	10,37	6062	7,36
DoS	12264	6,99	4089	4,97
Reconnaissance	10491	5,98	3496	4,25
Analysis	2000	1,14	677	0,82
Backdoor	1746	1,00	583	0,71
ShellCode	1133	0,65	378	0,46
Worms	130	0,07	44	0,05
Total	175341	100%	82332	100%

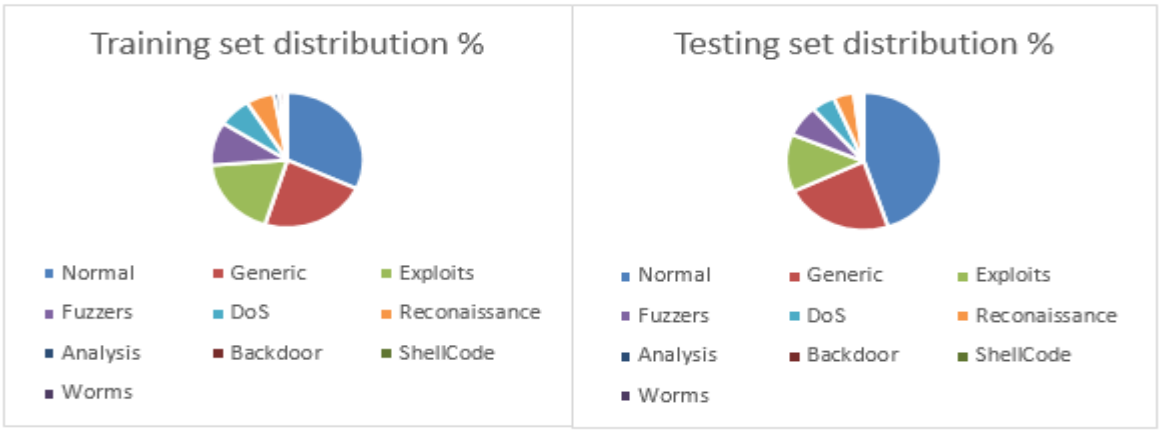

Figure 2. Training and Testing set distribution

B. *Preprocessing*

The categorical features of the UNSW-NB15 dataset pose a problem for many machine-learning algorithms. To deal with this problem the Spark ML library contains the String Indexer encoder. "String Indexer encodes a string column of labels to a column of label indices. The indices are in [0, numLabels], ordered by label frequencies, so the most frequent label gets index 0" [31]. For example, if we have a column that contains the following values (aaa, ccc, aaa, bbb, aaa, ccc), applying String Indexer on this column will get a new column with the following values (0,1,0,2, 0,1) where (aaa = 0, bbb = 2, ccc = 1).

Since our goal is the binary classification, we started by removing the attack_cat column used for the attack type detection (multi-classification) then we applied the String Indexer encoder on the three categorical features ('service', 'proto' and 'state') to convert them to numeric features.

C. *Feature Selection*

Compared to the Scikit-learn library, Spark ML does not offer much choice for the FS stage. In addition, the only feature selection method existing in the Spark ML library (ChiSquareSelector) is not compatible with the USNW-NB15 dataset because the implementation of ChiSqaure in spark only supports categorical features with a maximum limit of 10000 distinct values for each feature [32]. However, UNSW-NB15 contains 39 numerical features and several features with distinct values exceeding 10000 values for example: 'sjit', 'djit', 'sload', 'dload' and so on.

To solve this problem we integrated the FS method based on features importance (FI) in the Spark Framework. This method consists of using ML classification algorithms on training data to obtain the importance of each feature in the classification process. After we sort the features according to their importance (descending order) and then we select the most important features (Top N) that provide the best accuracy. In this step, the ML algorithm used to obtain the importance of each feature is (DT, RF, and gradient boosted trees (GBT)). We have tested all the possible combinations between these algorithms and the classification algorithm used for the creation of the learning model (next step) to find the best subset of features that gives us the best accuracy for each model.

D. *ML Classifier for intrusion detection (training the model)*

In this section, we describe the ML algorithms used in intrusion detection and the reason for choosing these algorithms.

Decision tree classifier: A decision tree is a structure based on a sequential decision process. The process starts with the root and evaluates a feature and then one of the two branches is selected. This procedure is repeated until a final leaf is reached, this leaf represents the desired classification target. DT are preferred to other algorithms because they are easy to interpret, handle categorical features, support multi-label classification, and work efficiently with un-normalized datasets [33, 34].

Random forest classifier: A random forest is a set of decision trees built on random samples using a different policy for splitting a node. This strategy uses a random subset of features (for each tree) to find the threshold that best separates the data. That is why many trees will be formed more weakly and each will produce a different prediction to reduce the risk of over fitting [33, 34].

The concept of FI that we previously introduced can also be applied using DT and RF. In our work we have opted for these two algorithms either for the stage of FS or for the creation of the intrusion detection model because they offer a very high accuracy, they are easy to

implement and also the other existing works have proved their performance for intrusion detection [6-9, 12].

E. Parameter Tuning

The presented ML algorithms have many parameters, these parameters have default values and can be manually defined to optimize the model and improve its accuracy. Our challenge is to find the best combination of these parameters adapted to the UNSW-NB15 and which ensures a more efficient detection model. The CrossValidator method [35] of the Spark ML library divides the dataset into a set of folds that are used as separate training and test datasets. E.g., with k = 5 folds, CrossValidator will generate 5 (training, test) dataset peers, each of which uses 4/5 of the data for training and 1/5 for testing. To evaluate a particular Parameter combination, CrossValidator computes the average evaluation metric for the 5 Models produced by fitting the Estimator on the 5 different (training, test) dataset peers.

For each parameters combination, we used the training dataset for the creation of the model and the testing dataset for the prediction then evaluates the model accuracy, in the end, the combination that offers us the best accuracy will be chosen.

F. Evaluation Metrics

In order to analyse and evaluate the performance of the classifiers (DT and RF), three evaluation metrics are used: accuracy, f1-weighted, and Area under ROC. Values for each metric are calculated from the confusion matrix of predictions.

Confusion matrix: The Confusion matrix is the easiest metrics used for evaluating the model performances. This matrix is not a performance measure, but all the performance metrics are based on the Confusion Matrix. The confusion matrix is a table with two dimensions and sets of "classes" in both dimensions. Target classifications are columns and Predicted ones are Rows. Four terms are still associated with this matrix: (i) True Positives (TP): are the cases when the target class was true and the predicted is also true. (ii) True Negatives (TN): are the cases when the target class was false and the predicted is also false. (iii) False Positives (FP): are the cases when the target class was false and the predicted is true. (iv) False Negatives (FN): are the cases when the target class was true and the predicted is false [36].

Accuracy: the proportion of correct predictions made by the model, it is defined as well:

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

F1-weighted: the F1-Scores average over all dataset classes, each weighted by its Support. The standard F-measure is F1, which gives equal importance to recall and precision. F1-Score is the harmonic mean of recall and precision [36].

$$\text{F1-Score} = 2 * \left(\frac{\text{recall} \times \text{precision}}{\text{recall} + \text{precision}}\right) \quad (2)$$

$$\text{F1-Weighted} = \frac{\sum_{i=1}^{K} \text{Support}_i \,.\, \text{F1}_i}{\text{Total}} \quad (3)$$

Where F1i is the F1-Score predicted for the ith target class.

Area under Roc (AUROC): is the integral of the area under the ROC Curve. ROC curve is a representation of sensitivity vs 1-specificity values obtained over a thresholds range [36, 37].

$$\text{AUROC} = \int_0^1 \frac{TP}{P}\, d\left(\frac{FP}{N}\right) \quad (4)$$

The metrics described above will be used in the next section to evaluate the proposed approach performance and to compare the results obtained with the related work.

4. RESULTS AND DISCUSSION

This section presents the proposed approach results. In this experiment, we tested all possible combinations between the FS based on FI and the ML algorithms used for training the intrusion detection model, three ML algorithms (DT, RF, and GBT) are used for FS step and two ML algorithms (DT and RF) for model formation step. The calculation result of parameters values by CrossValidator is summarized in table3 and the results of the experiments are listed in Table 4 and Fig.3.

Based on these results we can notice that the application of the FS methods for the reduction of the dataset to be analysed improves the performance of the ML model. As an example: (i) The accuracy of DT without FS in the first combination (89.6043) and the second combination where DT was used for both FS step and model formation (89.7415) with an improvement of 0.1372. (ii) The accuracy of RF without FS in the 7th combination (88.2220) and the 8th combination where RF was used for both FS step and model formation (88.7055) with an improvement of 0.4835.

TABLE 3. PARAMETER TUNING

Parameter of ML algorithm after the tuning	ML classifier	
	DT	RF
maxDepth	8	30
impurity	entropy	Entropy
maxBins	137	141
labelCol	"label"	"label"
featuresCol	"features"	"features"
seed		42
numTrees		2

TABLE 4. THE PROPOSED APPROACH RESULTS OF ALL THE COMBINATIONS (FS / ML)

ML classifier	Combination Number	FS method	Number of feature	Accuracy	F1-Weighted	AUROC
DT	1	Without FS	All (42)	89,6043	89,4628	88,731
	2	FI with DT	16	**89.7415**	**89,6068**	**88,8888**
	3	FI with DT	12	89.6808	89,5382	88,8003
	4	FI with GBT	16	89,6431	89,4981	88,7549
	5	FI with GBT	14	89,6371	89,492	88,7489
	6	FI with RF	19	89,6030	89,4571	88,713
RF	7	Without FS	42	88,2220	88,1056	87,4885
	8	FI with RF	29	**88,7055**	**88,6189**	**88,0808**
	9	FI with DT	23	88,5889	88,4828	87,883
	10	FI with RF	18	88,5439	88,4304	87,8129
	11	FI with GBT	16	88,5305	88,408	87,7652
	12	FI with RF	13	88,5014	88,3872	87,7688
	13	FI with DT	18	88,4771	88,3823	87,8225
	14	FI with DT	16	88,4188	88,3035	87,6911

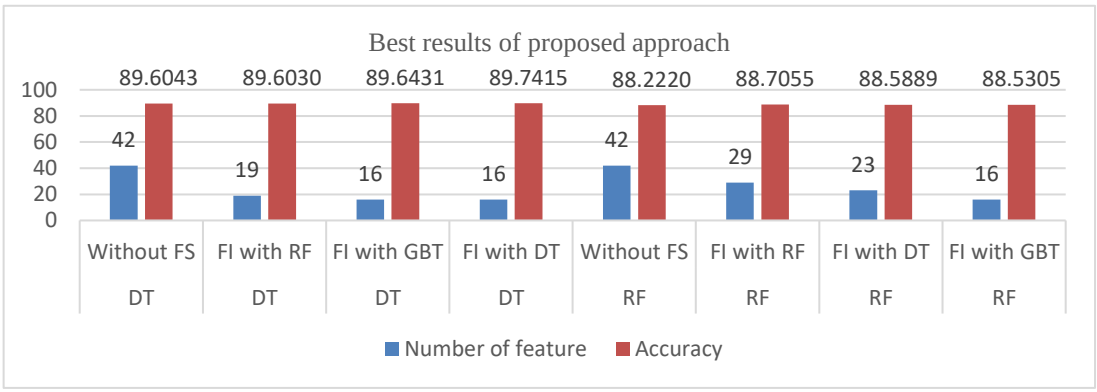

Figure 3. Best results of proposed approach

In addition, the comparison between the different combinations shows that the best result (the second combination, accuracy = 89.7415) is obtained during the use of the DT for both stages (FS and ML Model Training) with only 16 features. Comparisons between our results and the results obtained in the existing works are made to prove that our proposed approach ensures a better accuracy with a reduced number of features.

Comparing with Divekar et al. [7] in Table 5 and Fig.4 using F1-weighted metric. We prove that our proposed approach is better. When using the same ML (RF) algorithm used by the authors with our FS method (FI with RF), we found a better f1-weighted (88.6189 compared to 88.5) with a lower number of features used in the analysis (29 instead of 30). In addition, when using DT for both stages (FS and model training) we found results largely better (f1-weighted = 89.6068) with a very small number of features (only 16 features).

TABLE 5. F1-WEIGHTED OF PROPOSED APPROACH AND DIVEKAR ET AL. [7]

Article	Machine learning classifier	FS Method	Number of feature	Tools / Framework	F1-Weighted
Divekar et al. (2018)	RF	GI	30	Scikit-learn	88.5
Proposed approach	RF (combination 8)	FI with RF	29	Spark ML	88,6189
	DT (combination 2)	FI with DT	16	Spark ML	**89.6068**

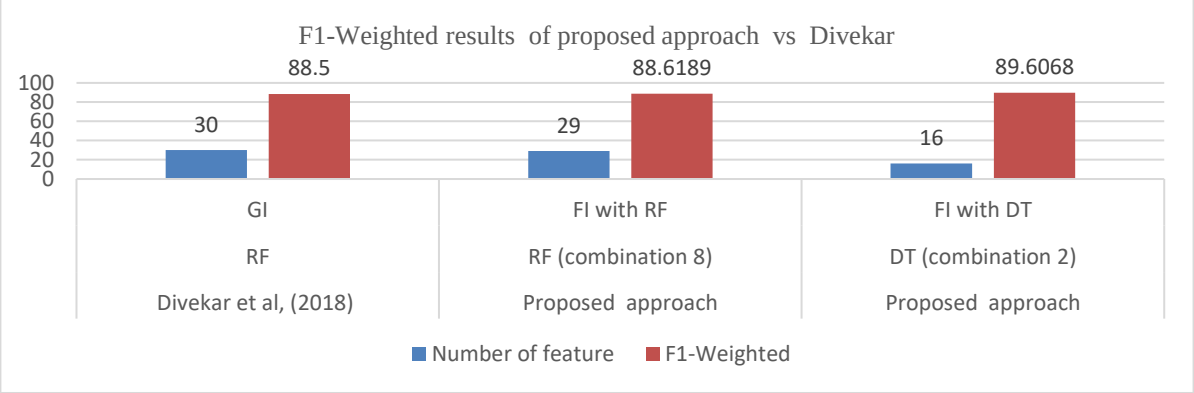

Figure 4. F1-Weighted of proposed approach vs Divekar [7].

To prove the approach effectiveness, we compared our best results with other works in Table 6 and Fig.5. First, Mustapha et al. [12] did not use any FS method and Khammassi and Krichen [9] provided results with and without FS. For this we compare our best result obtained when using the complete set of data without any FS with their results, and we find that the accuracy of our approach using DT as classifier (89.6043) is better of both (85.56, 81.49). Also, when using the FS, the best combination of our proposed approach (DT for both FS and model training) is much better than the Khammassi and Krichen approach [9] (GALR for FS and DT for model training) with an improvement of 8.3215 in the accuracy. The comparisons made and the results obtained in this section show that spark framework and its ML library provides the best results compared to tools used in existing works (WEKA, MATLAB, Scikit-learn...). Likewise, FS is a very essential step I the intrusion detection process, not only does it reduce the dataset to be analysed, but it also improves classification performance by keeping only the most useful features. The FS method used in our approach, which is based on the importance of the features, is very efficient compared to the other FS. Finally, DT offers the best performance for attack detection whether with or without FS.

TABLE 6. ACCURACY OF PROPOSED APPROACH AND SIMILAR WORKS [9, 12]

Reference	FS Method	ML classifier	Number of feature	Tools / Framework	Accuracy
Moustafa and Slay [12]	Without FS	DT	42	Visual Studio Business Intelligence 2008	85.56
Khammassi and Krichen [9]		DT	42	WEKA	81.49
Proposed approach		DT	42	Spark ML	**89.6043**
Khammassi and Krichen [9]	GA-LR	DT	20	WEKA	81.42
Proposed approach	FI with DT	DT	16	Spark ML	**89.7415**

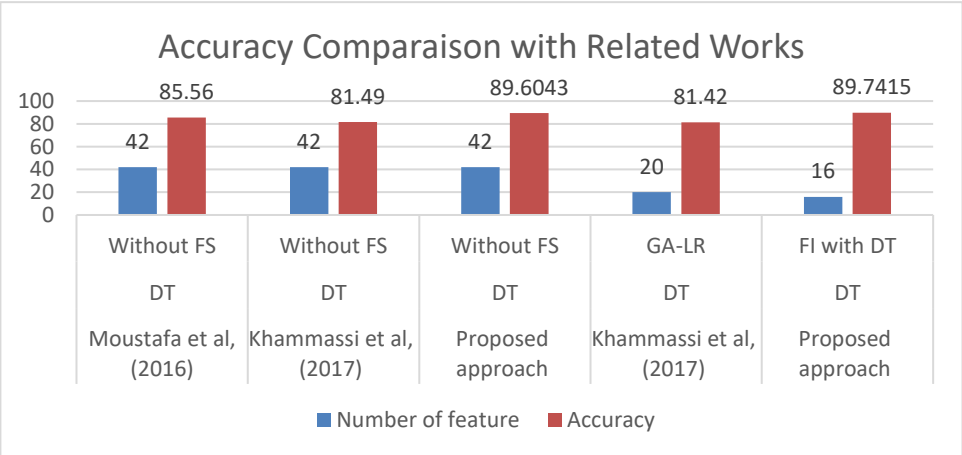

Figure 5. Accuracy of proposed approach and similar works

5. CONCLUSION

The proposed intrusion detection approach combines machine-learning algorithms, feature selection methods based on the features importance and the spark framework. The approach allows selecting the best subset of features ensuring higher accuracy. We used the UNSW-NB15 dataset, and we test the performance of the proposed approach by applying different combinations of ML (DT and RF) algorithms and FS method (based on the features importance) on the Spark framework. The evaluations result show that when we used the complete dataset without FS method, then DT is the best classifier. Also, we can notice that the application of the FS methods for the reduction of the analysed dataset improves the proposed approach performance, as proof, we obtained the best result during the use of the DT for both stages (FS and ML Model Training) with only 16 features. From the comparison with existing works, we can conclude that our approach ensures a better accuracy with a reduced number of features. As future work, we aim to improve our approach for attack type's detection (Multi classification) and test the approach with other public datasets.

ACKNOWLEDGMENTS

This research was supported by the Algerian Ministry of Higher Education and Scientific Research, General Direction of Scientific Research and Technological Development.

REFERENCES

[1] Buczak, A. L., & Guven, E. (2016). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. IEEE Communications Surveys & Tutorials, 18(2), 1153-1176. doi:10.1109/comst.2015.2494502

[2] Sung, A., Abraham, A., & Mukkamala, S. (2005). Cyber-Security Challenges. Enhancing Computer Security with Smart Technology, 125-164. doi:10.1201/9781420031225.ch6

[3] Chen, M., Mao, S., Zhang, Y., & Leung, V. C. (2014). Big Data. SpringerBriefs in Computer Science. doi: 10.1007/978-3-319-06245-7

[4] Karau, H., Konwinski, A., Wendell, P. and Zaharia, M. (2015). Learning Spark. 1st ed. Sebastopol, CA: O'Reilly Media, pp.1-8.

[5] Spark Overview. (n.d.). Retrieved June 25, 2019, from https://spark.apache.org/docs/2.2.0/

[6] Anwer, H. M., Farouk, M., & Abdel-Hamid, A. (2018). A framework for efficient network anomaly intrusion detection with features selection. 2018 9th International Conference on Information and Communication Systems (ICICS), 157-162. doi:10.1109/iacs.2018.8355459

[7] Divekar, A., Parekh, M., Savla, V., Mishra, R., & Shirole, M. (2018). Benchmarking datasets for Anomaly-based Network Intrusion Detection: KDD CUP 99 alternatives. 2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS). doi:10.1109/cccs.2018.8586840

[8] Janarthanan, T., & Zargari, S. (2017). Feature selection in UNSW-NB15 and KDDCUP99 datasets. 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE). doi:10.1109/isie.2017.8001537

[9] Khammassi, C., & Krichen, S. (2017). A GA-LR wrapper approach for feature selection in network intrusion detection. Computers & Security, 70, 255-277. doi:10.1016/j.cose.2017.06.005

[10] Khan, F. A., Gumaei, A., Derhab, A., & Hussain, A. (2019). TSDL: A Two-Stage Deep Learning Model for Efficient Network Intrusion Detection. IEEE Access, 7, 30373-30385. doi:10.1109/access.2019.2899721

[11] Moustafa, N., & Slay, J. (2015). A hybrid feature selection for network intrusion detection systems: Central points. ArXiv, abs/1707.05505.

[12] Moustafa, N., & Slay, J. (2016). The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Information Security Journal: A Global Perspective, 25(1-3), 18-31. doi:10.1080/19393555.2015.1125974

[13] Belouch, M., Hadaj, S. E., & Idhammad, M. (2018). Performance evaluation of intrusion detection based on machine learning using Apache Spark. Procedia Computer Science, 127, 1-6. doi:10.1016/j.procs.2018.01.091

[14] Dahiya, P., & Srivastava, D. K. (2018). Network Intrusion Detection in Big Dataset Using Spark. Procedia Computer Science, 132, 253-262. doi:10.1016/j.procs.2018.05.169

[15] Kamal, A.K.U, & Sathyadevan, S. (2017). Intrusion detection system using big data framework. ARPN Journal of Engineering and Applied Sciences. 12. 3909-3913.

[16] Lighari, S. N., & Hussain, D. M. (2017). Testing of algorithms for anomaly detection in big data using apache spark. 2017 9th International Conference on Computational Intelligence and Communication Networks (CICN). doi:10.1109/cicn.2017.8319364

[17] Othman, S. M., Ba-Alwi, F. M., Alsohybe, N. T., & Al-Hashida, A. Y. (2018). Intrusion detection model using machine learning algorithm on Big Data environment. Journal of Big Data, 5(1). doi:10.1186/s40537-018-0145-4

[18] Terzi, D. S., Terzi, R., & Sagiroglu, S. (2017). Big data analytics for network anomaly detection from netflow data. 2017 International Conference on Computer Science and Engineering (UBMK). doi:10.1109/ubmk.2017.8093473

[19] Wang, H., Xiao, Y., & Long, Y. (2017). Research of intrusion detection algorithm based on parallel SVM on spark. 2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC). doi:10.1109/iceiec.2017.8076533

[20] Zhou, B., Li, J., Ji, Y., & Guizani, M. (2018). Online Internet Traffic Monitoring and DDoS Attack Detection Using Big Data Frameworks. 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC). doi:10.1109/iwcmc.2018.8450335

[21] Zhou, B., Li, J., Wu, J., Guo, S., Gu, Y., & Li, Z. (2018). Machine-Learning-Based Online Distributed Denial-of-Service Attack Detection Using Spark Streaming. 2018 IEEE International Conference on Communications (ICC). doi:10.1109/icc.2018.8422327

[22] Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). 2015 Military Communications and Information Systems Conference (MilCIS). doi:10.1109/milcis.2015.7348942

[23] Weka 3: Machine Learning Software in Java. (n.d.). Retrieved June 25, 2019, from https://www.cs.waikato.ac.nz/ml/weka/

[24] Yap, B. W., Rani, K. A., Rahman, H. A., Fong, S., Khairudin, Z., & Abdullah, N. N. (2013). An Application of Oversampling, Undersampling, Bagging and Boosting in Handling Imbalanced Datasets. Lecture Notes in Electrical Engineering Proceedings of the First International Conference on Advanced Data and Information Engineering (DaEng-2013), 13-22. Doi: 10.1007/978-981-4585-18-7_2

[25] Scikit-learn. (n.d.). Retrieved June 25, 2019, from https://scikit-learn.org/stable/

[26] MATLAB. (n.d.) Retrieved June 26, 2019, from https://www.mathworks.com/products/matlab.html

[27] Qiu, J., Wu, Q., Ding, G. et al. A survey of machine learning for big data processing. EURASIP J. Adv. Signal Process. 2016, 67 (2016). https://doi.org/10.1186/s13634-016-0355-x

[28] M. Assefi, E. Behravesh, G. Liu and A. P. Tafti, "Big data machine learning using apache spark MLlib," 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, 2017, pp. 3492-3498.

[29] Apache Flink. (n.d.). Retrieved June 30, 2019, from https://flink.apache.org/

[30] Unsw.adfa.edu.au (n.d.). Retrieved July 04, 2019, from https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/

[31] Extracting, transforming and selecting features. (n.d.). Retrieved July 07, 2019, from https://spark.apache.org/docs/2.1.0/ml-features.html#stringindexer

[32] Nassar, M., Safa, H., Mutawa, A. A., Helal, A., & Gaba, I. (2019). Chi squared feature selection over Apache Spark. Proceedings of the 23rd International Database Applications & Engineering Symposium on - IDEAS 19. doi: 10.1145/3331076.3331110

[33] Bonaccorso, G. (2017). Machine learning algorithms: Reference guide for popular algorithms for data science and machine learning. Birmingham, UK: Packt.

[34] Classification and regression. (n.d.). Retrieved July 08, 2019, from https://spark.apache.org/docs/2.2.0/ml-classification-regression.html

[35] ML Tuning. (n.d.). Retrieved July 09, 2019, from https://spark.apache.org/docs/latest/ml-tuning.html#cross-validation

[36] Tharwat, A. (2018). Classification assessment methods. Applied Computing and Informatics. doi:10.1016/j.aci.2018.08.003

[37] Evaluation Metrics. (n.d.). Retrieved July 09, 2019, from https://spark.apache.org/docs/2.2.0/mllib-evaluation-metrics.html

Mohamed Seghire Othman Djediden: obtained her master degree in the area of information and communication technologies from USTO in 2016. At present, he is PhD student in department of data processing, University of Sciences and the Technology of Oran (USTO), Algeria. His research interests include cyber security, intrusion detection system, feature selection, big data analysis, and machine learning.

Hicham Reguieg obtained his phd degree in the area of big data and process mining from university of Blaise Pascal Clermont-Ferrand, France in 2014. Now, he is a university lecturer of computer science at the University of Science and Technology in Oran (USTO), Algeria. His research interests include events correlation discovery, process mining, big data analysis and machine learning.

Zoulikha Mekkakia Maaza obtained her PhD degree in the area of engineering protocols from USTO in 2004. At present, she is working as professor in department of data processing, University of Sciences and the Technology of Oran (USTO), Algeria. She is published several research papers in national and international conferences and journals. Her teaching and research interests include Ad hoc and sensors network, Qos, Distributed system, cloud computing, machine learning and big data analysis. Is research projects Head in Usto University, Reviewer in several international conference and journal.