Intrusion Detection Using Big Data and Deep Learning Techniques

Osama Faker
Cankaya University
Ankara, Turkey
osamafakre@gmail.com

Erdogan Dogdu
Cankaya University
Georgia State University (adjunct)
Ankara, Turkey
edogdu@cankaya.edu.tr

ABSTRACT

In this paper, Big Data and Deep Learning Techniques are integrated to improve the performance of intrusion detection systems. Three classifiers are used to classify network traffic datasets, and these are Deep Feed-Forward Neural Network (DNN) and two ensemble techniques, Random Forest and Gradient Boosting Tree (GBT). To select the most relevant attributes from the datasets, we use a homogeneity metric to evaluate features. Two recently published datasets UNSW NB15 and CICIDS2017 are used to evaluate the proposed method. 5-fold cross validation is used in this work to evaluate the machine learning models. We implemented the method using the distributed computing environment Apache Spark, integrated with Keras Deep Learning Library to implement the deep learning technique while the ensemble techniques are implemented using Apache Spark Machine Learning Library. The results show a high accuracy with DNN for binary and multiclass classification on UNSW NB15 dataset with accuracies at 99.16% for binary classification and 97.01% for multiclass classification. While GBT classifier achieved the best accuracy for binary classification with the CICIDS2017 dataset at 99.99%, for multiclass classification DNN has the highest accuracy with 99.56%.

CCS CONCEPTS

• **Security and privacy → Intrusion/anomaly detection and malware mitigation**; • **Computing methodologies → Supervised learning by classification**.

KEYWORDS

Intrusion detection system, big data, machine learning, artificial neural networks, deep learning, ensemble techniques, feature selection.

ACM Reference Format:
Osama Faker and Erdogan Dogdu. 2019. Intrusion Detection Using Big Data and Deep Learning Techniques. In *2019 ACM Southeast Conference (ACMSE 2019), April 18–20, 2019, Kennesaw, GA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3299815.3314439

ACMSE 2019, April 18–20, 2019, Kennesaw, GA, USA
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6251-1/19/04…$15.00
https://doi.org/10.1145/3299815.3314439

1 INTRODUCTION

Providing protection and privacy of big data is one of the most important challenges facing developers of security management systems, especially with the large expansion of the use of Internet networks and the rapid growth of the volume of data generated from several sources. This expansion and growth gave more space to hackers to launch their malicious attacks and use development techniques and tools for intrusion. On the other hand, researchers and developers of intrusion detection systems seek to increase the efficiency of malicious attack detection and the prediction of early attacks. Intrusion detection systems are one of the most important systems used in cyber security. Intrusion refers to attempts to compromise the confidentiality, integrity, availability of security mechanisms of computer or network resources or to bypass them. Intrusion detection systems (IDSs) are the hardware or software that monitors and analyzes data flowing through computers and networks to detect security breaches that threaten confidentiality, integrity or availability of a system's resources [9]. Intrusion detection systems use two basic methods to analyze events and detect attacks: misuse detection and anomaly detection. Misuse detection (or signature-based detection) is an analysis of system activities to search and detect patterns of attacks identical to or similar to previously known attack patterns and stored in a database intrusion detection system. An anomaly detection is the detection of unusual patterns of behavior in network traffic and it relies on building models that represent the normal behavior of users, hosts or the network where patterns of behavior that deviate from these models are detected and often represent abnormal behavior. The anomaly detection approach is based on machine learning, artificial neural networks, and deep learning techniques that have been widely used recently in the development of intrusion detection systems for mining and extracting knowledge through the training and testing of datasets [5]. Recently big data is being used in intrusion detection. Big data is data that is difficult to store, manage or manipulate using traditional techniques. The characteristics of big data include volume, variety and velocity [35] and they represent a major challenge for intrusion detection systems [28]. Volume refers to the quantity of data where data are generated from several different sources having exploded very dramatically over recent years. This requires monitoring and analyzing network traffic to integrate with the management and processing of big data. The large volume of data is often associated with another challenge, which is variety, meaning different data sources and therefore various data types including structured, semi-structured and unstructured data. Moreover, variety refers to heterogeneous data. Large IT infrastructures can generate huge quantities of data from many resources, such as application servers, networks, and workstations. Analyzing and monitoring heterogeneous data is a complex challenge and exacerbates the problems facing intrusion detection systems [36]. The huge change in the size and variety of data has also led to a change in the speed of data generation and streaming, referred to as velocity. Big data management and computing technologies have been created and developed over the last few years, including Hadoop [26], Apache Spark [33], Hive [30] and NoSQL [14]. Big data techniques have many advantages, such as speed in receiving, storing and processing data of various types. This work proposes an assessment of integration between the management and computation of big data and deep learning techniques using Apache Spark and the Keras deep learning library. A deep neural network, random forest and gradient boosted tree were used for classificaton, and k-means clustering technique is used feature selection by calculating the degree of homogeneity. These suggested approaches are applied on two recent datasets, namely CICIDS2018 and UNSW-NB15, both of which contain a set of common and updated attacks.

This paper organized as follows: Section 2 presents the related work. Section 3 presents a brief description of the datasets and classification techniques used. Section 4 gives details of the proposed approach. Section 5 presents the results and evaluation, followed by Section 6 with a summary and discussion. Finally, the conclusion and future work are presented in Section 7.

2 RELATED WORKS

Sharafaldin et al. [24] produced a reliable CICIDS2017 dataset that contained benign and seven common attack network flows. They examined the performance and accuracy of the selected features with seven common machine learning algorithms, namely K-Nearest Neighbor, Random Forest, ID3, Adaboost, Multilayer Perceptron, Naive-Bayes and Quadratic Discriminant Analysis. In addition, the new dataset is compared with publicly available datasets from 1998 to 2016 based on 11 criteria representing common errors and criticisms of previous datasets. The comparison results show that the new dataset addresses all errors and criticisms.

Nour and Jill [19] proposed a statistical analysis to evaluate UNSW-NB15 and KDD99 datasets wherein UNSW-NB15 was divided into a test set and a training set and assessed from three aspects of statistical analysis, feature correlation, and the complexity evaluation. The results show that UNSW-NB15 is considered a reliable dataset to evaluate existing and novel methods of IDS. Coelho et al. [6] suggested the use of a homogeneity metric between labels and data clusters for semi-supervised feature selection. The results show that information retrieved from clusters can improve the estimation of feature relevance and of feature selection tasks, especially when the number of labeled data is too small and the unlabeled data is numerous. Vijayanand et al. [32] proposed a novel intrusion detection system with a genetic algorithm based feature selection and multiple support vector machine classifiers. The proposed approach relies on selecting the informational features for each category of attack instead of the common features of every attack and applied to CICIDS2017 dataset. The experiments demonstrates the effectiveness of the novel approach and achieved a high rate of accuracy in intrusion detection. Gupta and Kulariya [13] proposed a framework that combines feature selection algorithms and classification algorithms, and their implementation into the big data computing environment of Apache Spark. Correlation based feature selection and Chi squared feature selection were used for feature selection and logistic regression, support vector machines, random forest, Gradient Boosted Decision trees, and Naive Bayes for classification. Dahiya et al. [7] suggested an intrusion detection system using Apache Spark to implement the proposed approach relying on two feature reduction algorithms, Canonical Correlation Analysis (CCA) and Linear Discriminant Analysis (LDA), and using seven well-known classification algorithms, namely Naive-Bayes, REP Tree, Random Tree, Random Forest, Random Committee, Bagging and Randomizable. Beluch et al. [3] evaluated the performance of a set of classification algorithms (SVM, Decision Tree, Naive Bayes, Random Forest) using Apache Spark on the complete UNSW-NB15 dataset with all 42 features and concluded with the best performance for Random Forest (97% accuracy). Primartha and Tama [20] compared the performance of IDSs by applying a random forest classifier with respect to two performance measures, namely accuracy and false alarm rate and used a 10-fold cross validation technique. Three datasets of IDSs (NSL-KDD, UNSW-NB15 and GPRS) were used in the experiment and the results were compared for the Multilayer Percepteron (MLP), Decision Tree, and NB-Tree classifiers. The results of the study demonstrated the effectiveness of the proposed model based on Random Forest classifier with parameter settings and using a cross validation technique. Belouch et al. [2] presented a two-stage classifier based on Reduced Error Pruning Tree (RepTree) algorithm. In the first stage the method decides if it is an attack and in the second stage the type of attack is determined. This method gave better performance in speed and accuracy using datasets UNSW-NB15 and NSL-KDD in evaluation results. Al-Zewairi et al. [1] proposed a deep learning (DL) model based on Artificial Neural Networks (ANN) using the back-propagation and stochastic gradient descent methods, wherein the model was evaluated as a binomial classifier for NIDSs on the UNSW-NB15 dataset. DL model contained five hidden layers in each layer of ten neurons and the 10-fold cross validation technique is used. The results show that the model obtained high accuracy and a low alarm rate compared to earlier models.

3 DESCRIPTION OF DATASETS AND CLASSIFICATION TECHNIQUES

There are several datasets available to evaluate the proposed techniques in developing and improving the performance of intrusion detection systems. According to most of the studies on intrusion detection systems, KDD Cup 99 dataset[1](launched in 1999) [29] and a refined version named NLS-KDD datasets[2] are used mostly. These datasets include four types of attacks: DOS/DDOS, Probing, U2R, and R2L. They are relatively old and cannot be relied upon to evaluate intrusion detection systems because they do not contain new types of attacks and modern normal behaviors, especially with the great development in attack methods and the emergence of new types [8]. Therefore, we used two new intrusion detection datasets used in recent studies, namely UNSW-NB15 and CICIDS2017. In the remaining part of this section, we first present the datasets we

[1] KDD Cup 1999, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[2] NLSKDD, https://www.unb.ca/cic/datasets/nsl.html

used, then a Homogeneity Metric used for feature selection, and the classification techniques used in this work (Deep Neural Network, Random Forest, Gradient Boosted Tree).

3.1 UNSW-NB15 Dataset

UNSW-NB15[3] is one of the latest datasets created by the cyber-security research group at Australian Centre of Cyber Security (ACCS) to evaluate IDSs. It has become available to researchers since late 2015. The IXIA PerfectStorm[4] testing platform is used to generate approximately 100GB of normal and abnormal network traffic. 49 features were extracted from the raw pcap file by using the Argus[5] and Bro-IDS[6] split into five sets of basic features: flow features, content features, time features, and the additional generated features. The dataset has a total number of 2,540,047 records with 9 different types of recent and common attacks, namely, Fuzzers, Analysis, Backdoors, DOS, Exploits, Generic, Reconnaissance, Shellcode and worms. In the dataset 321,283 records are attack records, and the total number of normal records is 2,218,764. The size of the normal information packets represents 88% of the dataset size, while the attack information packets represent 12%.

3.2 CICIDS2017 Dataset

CICIDS2017[7] [17] dataset is released in late 2017 from Canadian Institute for Cybersecurity (CIC). The dataset contains benign and the most current common attacks. B-Profile system [18] is used for the abstract behavior of human interactions and to generate benign natural background traffic and CICFlowMeter [24] is used to extract the features from the dataset. CICIDS2017 Dataset contains the most common attacks based on the 2016 McAfee report (DOS, DDOS, Web-based, Brute force, Infiltration, Heart-bleed, Bot and Scan) with more than 80 features extracted from generated network traffic. To create the dataset a complete network topology, including different operating systems (Windows, Ubuntu, MAC OS X) and network devices (modems, switches, firewall, routers) are used. CICIDS2017 dataset Contains 14 types of attacks with 2,273,097 records of normal packet information (BENIGN) (80% of the dataset) and 557,646 records of attack packets information (20% of the dataset).

3.3 K-means Clustering and Homogeneity Metric

K-means clustering algorithm is one of the most common unsupervised machine learning algorithms and is defined as a method in which data are divided into K groups in a manner such that objects in each group share more similarity than with other objects in other groups [31]. K is the number of clusters determined by the user. After determining the number of clusters,

- the centroids are chosen randomly for each cluster.
- the distance between the centroid and the data points is measured by the Euclidean equation.
- the data points closest to the centroid are grouped.
- the mean distance between these data points is calculated and the mean is defined as a new centroid.
- the process is repeated until no data points move between the clusters.

Homogeneity on the other hand is a clustering metric that is used to determine the homogeneity of data points in a single cluster. The cluster must assign only those data points that are members of a single class to a single cluster. This means that the cluster entropy should be zero. Entropy refers to randomness or unpredictability. Homogeneity is defined as [22]:

$$h = \frac{H(C,K)}{H(C)}$$

where $H(C,K)$ is the conditional entropy of the classes given the cluster assignments. $H(C,K)$ is calculated as:

$$H(C,K) = -\sum_{k=1}^{|K|}\sum_{c=1}^{|C|} \frac{n_{c,k}}{n} \log \frac{n_{c,k}}{n_k}$$

and $H(C)$ is the entropy of the classes and is given by:

$$H(C) = -\sum_{c=1}^{|C|} \frac{n_c}{n} \log \frac{n_c}{n}$$

where n is the number of datapoints, n_c is the number of datapoints belonging to class c, n_k the number of datapoints respectively belonging to cluster k, and $n_{c,k}$ the number of datapoints from class c assigned to cluster k.

3.4 Deep Neural Networks (DNN)

Neural Networks [23] is a technique created to simulate the human brain for patterns recognition, and used in many learning tasks [27]. In general, it consists of three layers, one layer for input, one for output and at least one layer hidden between them. Input passes from the input layer through the hidden layers, up to the output layer through a set of neuron nodes in each layer, whether it be a linear relationship or a non-linear relationship. DNN indicate that there is more than one hidden layer in the neural network. It is widely used in supervised and unsupervised learning, and for classification and clustering.

Input data is passed to the hidden layers by a group of neurons in each layer such that these neurons are connected to each other by weight, which represents the importance of the input value, the more valuable neurons from the others will have a greater impact on the next layer of neurons. Various types of Artificial Neural Networks (ANN) have been developed, the first and simplest Neural Networks that are widely used are Feed-Forward Neural Networks, which is proposed in this work as one of the machine learning techniques to be used. In this type of networks, information is transmitted in parallel from the input layer directly through the hidden layers and then into the output layer without cycles/loops. The proposed Deep Neural Network contains three hidden layers. Each layer is fully connected to the next layer in the network, where ReLU function is used in hidden layers and sigmoid function is used in the output layer for binary classification while the SoftMax function is used in the output layer for multiclass classification.

[3] https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/
[4] https://www.ixiacom.com/products/perfectstorm
[5] https://qosient.com/argus/index.shtml
[6] https://qosient.com/argus/index.shtml
[7] http://www.unb.ca/cic/datasets/ids-2017.html

3.5 Random Forest (RF) and Gradient Boosted Tree (GBT)

Ensemble methods are used in machine learning to minimize noise, bias and variance factors [34]. They aim to improve the stability and accuracy of machine learning algorithms. An ensemble method is a set of predictions that are integrated to obtain a final prediction where several different predictions are combined to reach the target prediction. Ensemble techniques are classified as Boosting and Bagging.

Random Forest (Bagging) [4] is a supervised learning algorithm that is one of the most used algorithms as it is flexible and easy to use for classification and regression tasks. It depends on the ensemble method where it creates a set of Decision Trees and combines them to obtain a higher accuracy and stability prediction. RF rely on the Decision Tree algorithm to build trees. The more trees that are built, the greater the ability of RF to resist noise and increase efficiency of classification. Furthermore, the ability of a RF algorithm to operate within distributed and parallel computing and efficiently process data by simple classifiers built by the Decision Tree gives it the advantage of scalability and adaptation to large changes in data volume and variety [16]. The growth of a single Decision Tree can be described within RF in the following steps:

- Several random sub-sampling sets are created from the basic data set with a replacement.
- The features are selected with the same sampling approach where a subset of the features is randomly selected from the sum of the overall features.
- Training the Decision Tree on each sample.
- Decision Trees grow without pruning.
- Integrating all the Decision Trees prediction results by simple majority voting.

Gradient Boosting Tree (Boosting) [11] is a supervised machine learning method that is widely used for classification and regression tasks. GBT is an ensemble method similar to RF. However, the difference lies in the creation of the predictors (Decision Trees). GBT is based on weak learners (high bias, low variance). A weak learner in a Decision Tree means a shallow tree such that the GBT starts with the shallow tree to build a predictor, followed by calculating the error of expectations and passing the errors to the second tree as a target. The second tree adopts the new prediction model according to the data of the first tree model. The error is calculated for the new predictor model and passed to the third tree and so forth.

4 PROPOSED APPROACH

The proposed method includes a set of steps that begins with the preprocessing of the datasets and then the selection of features, where the K-means clustering (with homogeneity metric) is used as an unsupervised feature selection technique for the selection of relevant features from the datasets to improve the performance of classifiers. Five-fold cross validation is used to estimate and improve the performance of machine learning models. Deep neural network and two ensemble techniques (RF and GBT) are used to extract the models using the relevant subsets of features.

Our proposed approach consists of three major phases. These are data preprocessing, feature ranking and selection, and creating and evaluating the learning models, as explained below. The evaluation code can be accessed from the project repository[8].

4.1 Dataset Preprocessing

To provide more suitable data for the neural network classifier, the dataset is passed through a group of preprocessing operations. These operations are summarized below:

- Removing socket information: As the original dataset includes the IP address and port numbers of the source and destination hosts in the network, it is important to remove such information to provide unbiased detection. Using such information may result in over-fitted training towards the socket information. However, it is more important to allow the classifier to learn from the characteristics of the packets itself, so that any host with similar packet information is filtered out regardless of its socket information.
- Removing white spaces: Some of the multi-class labels in the dataset include white spaces. Such white spaces result in different classes as the actual value differs from the labels of other tuples in the same class.
- Label encoding: The multi-class labels in the dataset are provided with the attack names, which are string values. Thus, it is important to encode these values into numerical values, so that the classifier can learn the class number to which each tuple belongs. This operation is executed using the multi-class labels only, as the binary labels are already in zero-one formation.
- Data normalization: The numerical data in the dataset is of different ranges, which poses a number of challenges to the classifier during training to compensate these differences. Thus, it is important to normalize the values in each attribute so that the minimum value in each attribute is zero and the maximum being one. This provides more homogeneous values to the classifier while maintaining relativity among the values of each attribute.
- Removing/replacing missing and infinity values: CICIDS2017 dataset contains 2,867 tuples as missing or infinity values. This is addressed in two ways and that produced two datasets. The first is a dataset without the missing or infinite values (Rem-CICIDS2017), where all missing and infinity values are removed. The second is a dataset with the infinite values replaced with the maximum value and the missing values are replaced with the average values (Rep-CICIDS2017). Both datasets are used to evaluate the proposed method.
- Removing normal traffic: For multiclass classification, information packets that represent normal network traffic from both datasets are ignored since they consist of a large portion of the traffic and only the attack information packets are used to evaluate the proposed method.

4.2 Features Ranking and Selection using Homogeneity Metric

After the preprocessing phase, K-means clustering algorithm is applied to the dataset for feature ranking. The technique used for

[8] https://github.com/OsamaFaker/INTRUSION-DETECTION-BIG-DATA

feature ranking and selection is that taking each attribute separately, then use it to cluster the dataset. In binary classification (K=2) the data points are clustered into two groups, normal and anomaly. For multi-class classification, K equals to the number of attack types in the dataset. Thereafter, the homogeneity score is calculated for the resulting clusters, which is then used as a ranking score for the feature used for clustering. High ranking scores indicate that better classification can be conducted relying on those features, while lower scores indicate that those features do not have a significant role in the classification. When the rank of homogeneity is determined for each feature, they are arranged in descending order from the highest rank to the lowest rank. The homogeneity rank value is between zero and one, zero referring to the lack of homogeneity among the data points of the feature in the clusters, and one referring to high homogeneity for the feature. Finding a unique feature that belongs to a single class is probably more effective than relying on common features in the classification process, especially with the increasing volume of heterogeneous data generated from several sources.

4.3 Applying Deep Neural Networks and Ensemble Techniques

Deep Neural Networks, Random Forest, and Gradient Boosting Tree classification algorithms are applied first on the full CICIDS2018 and UNSW-NB15 datasets, and then on the subsets of selected features from both datasets. Deep Neural Networks is considered one of the Feed-Forward Artificial Neural Networks. DNN consists of multiple layers of nodes. Each layer is fully connected to the next layer in the network. 43 and 78 nodes are used in input layer to represent the number of input features in UNSW-NB15 and CICIDS2017 datasets respectively. Three hidden layers with 128, 64, and 32 nodes are set per layer respectively. *ReLU* activation function is used in the hidden layers. $SoftMax$ function used in the output layer for multiclass classification and *sigmoid* function is used for binary classification. With the backpropagation for learning the model, the training epoch is set to 1000 (epoch means one pass over the full training set) and batch size is set to 1 million, the total number of training examples present in a single batch. We tested the datasets in two scenarios, one for binary classification in which the data points are considered for normal and attack types, and in the second scenario all attack types are considered for a multi-class classification task.

The first scenario uses binary classification as pointed out, in which the aim of the classification process is to identify two groups; predicting each packet's type as normal or attack traffic. Machine learning techniques are initially applied to datasets with all features (full consideration). Then, we remove the feature with the lowest rank (the lowest homogeneity score) from the dataset and repeat the classification and evaluate the accuracy metric. We keep removing the next lowest ranking feature in each repetition and evaluating the classification result until all the last feature (the highest scoring feature). We applied this scenario on both datasets (UNSW-NB15 and CICIDS2017) using the above explained algorithm for all three classification algorithms. In DNN, the output has two nodes and the *sigmoid* function is used; in RF, the number of trees is set to 100, the depth of the tree is set to 4; in GBT, *log loss* function is used and 100 iterations are applied.

The second scenario is for multiclass classification, where the attack types are the classes. There are nine different attacks in UNSW-NB15 dataset and fourteen different attacks in CICIDS2017 dataset. Therefore, for Deep Learning the number of nodes in the output layer is set to the number of attack types, and $SoftMax$ activation function is used.

To obtain high performance and reduce the bias of machine learning techniques, k-fold cross-validation (5-fold in our experiments) is used in this work to evaluate the machine learning models. The entire dataset is split into 5 bins randomly, each bin is then used once for testing, while the remaining bins are used for training in each iteration. The average accuracy of those evaluations are then used as the final accuracy value for each model.

Feature ranking and selection algorithm is summarized in Algorithm 1 below.

```
input  : Dataset $D$ with fatures $f_0$,$f_1$, ..., $f_{n-1}$ and attack label
         ($1...k$) for each record
output : Best accuracy and the set of features resulting in the
         best accuracy
for $i \leftarrow 0$ to $n - 1$ do
    // K-means clustering with $k$ = # attack types
    $C \leftarrow Kmeans(D[f_i], k)$;
    $HS\{f_i\} \leftarrow homogeneity\_score(C)$;
end
$HS' = reverse\_sort(HS)$ // in descending order;
D' = D;
for $i \leftarrow n - 1$ to 1 do
    $HS', D' \leftarrow$
      $remove\ the\ lowest\ scored\ feature\ f_i\ from\ HS'\ and\ D'$;
    $accuracy\{f_i\} \leftarrow train\_test\_n\_fold(D', 5)$;
end
$index \leftarrow max(accuracy)$;
$best\_accuracy \leftarrow accuracy(index)$;
return $best\_accuracy$, $HS'(0, index)$;
```

Algorithm 1: Feature Ranking and Selection

5 RESULTS AND EVALUATION

This section presents the results of evaluations for the two scenarios, namely the binary and multi-class classifications. The experiments are conducted on Amazon's Elastic Map Reduce (EMR) cloud services with PySpark setup. A cluster of ten nodes are used for these experiments in which each node had a 2.4 GHz Intel Xeon E5 2676 v3 processor and 64 GB of memory. The random forest and GBT classifiers are implemented using Apache Spark's built-in machine learning library MLlib. The deep learning model is implemented using the distributed Keras[9] library.

5.1 Binary Classification

In binary classification with UNSW-NB15 dataset, although the differences are marginal. Table 1 shows the accuracy results, where

[9] https://keras.io

the three classifiers achieved high accuracy rates. Full dataset and the best feature selection results are reported. #ftrs column reports the number of features used in the best accuracy case. DNN with three hidden layers is the best performing classifier, with 99.19% accuracy, and then RF classifier achieved 98.86% accuracy. GBT provided an extremely low average prediction time performance per each packet at 0.35 ns, compared to DNN and RF classifiers' prediction times with 1.35 ns and 11.36 ns respectively. This is due to the lowest number of features used in this case (26 features). Moreover, by comparing the results of the proposed method (feature selection) with the results of the classifiers on full dataset, there was a slight improvement in the accuracy, yet the prediction times improved considerably, especially with GBT from 0.7 ns to 0.35 ns since the number of features used is dropped from 49 to 26.

Table 1: Binary Classification Results with UNSW-NB15 Dataset

	Full Dataset		Selected Features		
	Accuracy	Prediction Time (ns)	Accuracy	Prediction Time (ns)	#ftrs
DNN	99.16%	1.43	**99.19%**	1.35	41
RF	98.85%	12.13	98.86%	11.36	36
GBT	97.83%	0.70	97.92%	0.35	26

In the case of CICIDS2017 dataset with binary classification we also tried two versions of the dataset, one with replacement of missing and infinite values (Rep-CICIDS2017), and another with removal of those values (Rem-CICIDS2017). The best accuracy is obtained with GBT classifier on Rep-CICIDS2017 with 99.97% but the lowest prediction time is obtained by DNN with 0.05 ns (Table 2). RF on the other has the worst performance with 92.72% accuracy, but using the lowest number of features (6 out of 49) with the worst average prediction time performance (6.81 ns). There are also minimal improvements in accuracy with feature selection in comparison to the full dataset, meaning eliminating some features helps in improving the accuracy as well the prediction times (1.23 ns vs 0.53 ns in GBT case).

Table 2: Results of Binary Classification with Rep-CICIDS2017 Dataset

	Full Dataset		Selected Features		
	Accuracy	Prediction Time (ns)	Accuracy	Prediction Time (ns)	#ftrs
DNN	97.72%	0.05	97.73%	0.05	59
RF	92.54%	7.71	92.72%	6.78	6
GBT	99.81%	1.23	**99.97%**	0.53	23

Table 3 shows the results on Rem-CICIDS2017 dataset for binary classification. There are very small differences in the results compared to Rep-CICIDS2017 dataset, however GBT achieved the best performance with (99.99%) accuracy.

Table 3: Binary Classification Results with Rem-CICIDS2017 Dataset

	Full Dataset		Selected Features		
	Accuracy	Prediction Time (ns)	Accuracy	Prediction Time (ns)	#ftrs
DNN	97.71%	0.05	97.72%	0.05	59
RF	92.54%	7.68	92.71%	6.81	8
GBT	99.81%	1.24	**99.99%**	0.49	21

5.2 Multiclass Classification

For multiclass classification, information packets that represent normal network traffic from both data sets are deleted and only the attack information packets are kept. 321,283 tuples that represent packet information of 9 different types of attacks in UNSW-NB15data set and 557,646 tuples in CICIDS2017 data set from 14 different types of attacks are used. Here, we only tested DNN and RF classifiers, since GBT does not support multiclass classification in Apache Spark.

UNSW-NB15 dataset is tested with DNN and RF, and the results show that DNN has a much higher accuracy with 97.04% in comparison to RF, and much lower prediction time (4.71 ns) with lower number (29) of features selected (Table 4). Accuracy is slightly better in feature selection method.

Table 4: Results of Multiclass Classification with UNSW-NB15 Dataset

	Full Dataset		Selected Features		
	Accuracy	Prediction Time (ns)	Accuracy	Prediction Time (ns)	#ftrs
DNN	97.01%	5.15	**97.04%**	4.71	29
RF	91.76%	24.85	91.77%	23.63	31

By using attack packets of Rep-CICIDS2017, DNN achieved a high accuracy (99.55%) in the classification of attack packets with (0.64 ns) average prediction time, which is very low using the full dataset (Table 5). DNN classifier obtained a slightly higher accuracy on the dataset with feature selection, again with a high improvement over RF classifier (Table 5).

Table 5: Results of Binary Classification with Rep-CICIDS2017 Dataset

	Full Dataset		Selected Features		
	Accuracy	Prediction Time (ns)	Accuracy	Prediction Time (ns)	#ftrs
DNN	99.55%	0.64	**99.57%**	0.70	75
RF	92.57%	7.40	92.72%	6.22	8

Same experiments are repeated on Rem-CICIDS2017 dataset (Table 6). There is no significant differences in comparison to the results of Rep-CICIDS2017 dataset. This means there is little between deleting or replacing the lost and infinite values. DNN again resulted in the highest accuracy (99.56%) in both full dataset and feature selected dataset cases). This is again much better than RF's (92%) accuracy levels.

Table 6: Results of Binary Classification with Rem-CICIDS2017 Dataset

	Full Dataset		Selected Features		
	Accuracy	Prediction Time (ns)	Accuracy	Prediction Time (ns)	#ftrs
DNN	**99.56%**	0.63	**99.56%**	0.73	67
RF	92.54%	6.98	92.71%	6.04	6

6 DISCUSSION

In binary classification with the UNSW-NB15 dataset, although the differences are marginal, the results show that the best accuracy results are obtained by DNN at (99.16%). GBT provided an extremely lower average prediction time per packet at (0.70 ns), compared to the other classifiers which consumed (1.43 ns) and (12.13 ns). When we compare the full dataset results with the results of the feature selection approach, we see a slight improvement in the accuracy and time prediction. Using the CICIDS2017 dataset with the two methods that replace and remove the missing and infinity values, the best accuracy rate was provided by GBT classifier at (99.81%). DNN classifier achieved the lowest prediction time with (0.05 ns). For multi-class attack classification, the results show the superiority of DNN using both attack datasets with (5.15 ns) and (0.64 ns) prediction times, providing (97.01%) and (99.55%) accuracy levels with UNSW-NB15 and CICIDS2017 datasets respectively.

In contrast to earlier methods, the proposed approach achieved better performances in accuracy with the use of the homogeneity metric in feature ranking and selection.

Apache Spark technique has greatly improved the training and prediction time of the three classifiers when compared to traditional techniques [10]. This improvement gives intrusion detection systems the ability to make decisions more efficiently in terms of blocking or allowing data to pass through a network. In addition, the integration between Apache Spark and the Keras Deep Learning Library has increased the capabilities of deep learning algorithms to work more efficiently and more quickly. The slight improvement in classifier performance using the proposed feature selection approach suggests that this approach can be developed to deal more efficiently with heterogeneous data, which is one of the most significant challenges of intrusion detection systems.

As shown in the following Table 7, the accuracy of classifiers used in this study is compared with the previous studies in binary classification of UNSW-NB15 dataset. Our method with feature selection has the best accuracy (99.19%) using DNN classifier on UNSW-NB15 dataset.

Table 8 shows the comparison between the accuracy of this study and the earlier studies using UNSW-NB15 dataset for multi-class classification. Again, our feature selection method with DNN classifier achieved the best accuracy (97.04%).

Table 9 compares our binary classification with feature selection accuracy results on CICIDS2017 dataset (with replacement) to the earlier studies. Again, our method performs better than the earlier studies with 99.97% accuracy using GBT classifier.

Table 7: Comparison of Accuracy of Binary Classification Prediction with Earlier Studies Used UNSW-NB15 Dataset

Study	Classifier	Acc (%)
Primartha and Tama [18]	Random Forest	95.5
	Multilayer Perceptron	83.50
Nour and Slay [12]	Naive Bayes	79.50
	Expectation-Maximization	77.20
	Linear Regression	83.00
Belouch, et al. [19]	RepTree	87.80
	Naive Bayes	80.04
	Random Tree	86.59
	Decision Tree	86.13
	Artificial Neural Network	86.31
Zewairi, et al. [20]	Deep Learning	98.99
Our Work	Deep neural network	**99.19**
	Random Forest	98.86
	Gradient Boosted Tree	97.92

Table 8: Comparison of Prediction's Accuracy Attack of Multiclass Classification with Earlier Studies Used UNSW-NB15 Dataset

Study	Classifier	Acc (%)
Belouch, et al. [19]	RepTree	79.20
	Random Tree	76.21
	Naïve Bayes	73.86
	Artificial Neural Network	78.14
Gharaee, Hamid [36]	Genetic + SVM	93.25
Our Work	Deep neural network	**97.04**
	Random forest	91.77

Table 9: Comparison of Accuracy of Binary Classification Prediction with Earlier Studies Used CICIDS2017 Dataset

Study	Classifier	Acc (%)
Iman , et al. [11]	K-Nearest Neighbors	96.00
	Random Forest	98.00
	ID	98.00
	Adaboost	77.00
	Multilayer Perceptron	77.00
	Naive Bayes	88.00
	Quadratic Discriminant Analysis	97.00
Vijayanand, et al.[14]	SVM+ Genetic	99.85
Alves and Drummond.[37]	Genetic + Profiling	92.85
Our Work	Deep neural network	97.73
	Random forest	92.72
	Gradient Boosted Tree	**99.97**

7 CONCLUSION AND FUTURE WORK

This paper presented a method to improve the performance of intrusion detection systems by integrating big data technologies and deep learning techniques. UNSW-NB15 and CICIDS2017 datasets are used to evaluate the proposed approach. In our method we used the homogeneity metric to rank and select the features. Deep Neural Networks (DNN), Random Forest (RF), and Gradient Boosted Tree (GBT) classifiers are used to classify the attacks in binary and multiclass modes. All experiments are conducted on Apache Spark with the Keras Deep Learning Library. RF and GBT classifiers are used from Apache Spark Machine Learning Library. The results show high accuracy levels with DNN for binary and multiclass classification on UNSW-NB15 dataset (99.19% and 97.04% respectively) and very low prediction times. GBT classifier achieved the best accuracy (99.99%) for binary classification using the CICIDS2017 dataset, and (99.57%) accuracy for multiclass classification on the same dataset using DNN classifier.

For future work, improving the performance of intrusion detection with feature selection using homogeneity metric will be investigated with better feature selection schemes. We did not report the performance of distributed Apache Spark processing with varying node counts in the cluster, that is also in our agenda for future work and analysis.

REFERENCES

[1] M. Al-Zewairi, S. Almajali, and A. Awajan. 2017. Experimental Evaluation of a Multi-layer Feed-Forward Artificial Neural Network Classifier for Network Intrusion Detection System. *2017 International Conference on New Trends in Computing Sciences (ICTCS), Amman, Jordan*, pp. 167-172, IEEE

[2] M. Belouch, S. El Hadaj, and M. Idhammad. 2017. Two-stage Classifier Approach Using RepTree algorithm for Network Intrusion Detection. *International Journal of Advanced Computer Science and Applications*, 8(6), pp. 389-394.

[3] M. Belouch, S. El Hadaj, and M. Idhammad. 2018. Performance Evaluation of Intrusion Detection based on Machine Learning Using Apache Spark. *Procedia Computer Science* 127, pp. 1-6.

[4] L. Breiman. 2001. Random Forests. *Machine Learning*, 45(1), pp. 5-32.

[5] V. Chandola, A. Banerjee, and V. Kumar. 2009. Anomaly Detection: A Survey. *ACM Computing Surveys*, 41(3), pp. 1-15.

[6] F. Coelho, A. Braga, and M. Verleysen. 2012. Cluster Homogeneity as a Semi-supervised Principle for Feature Selection Using Mutual Information. *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Bruges, Belgium.*

[7] P. Dahiya and D. Srivastava. 2018. Network Intrusion Detection in Big Dataset Using Spark. *Procedia Computer Science* 132, pp. 253-262.

[8] L. Dhanabal, and S. p.Shantharajah. 2015. A Study on NSL KDD Dataset for Intrusion Detection System based on Classification Algorithms. *International Journal of Advanced Research in Computer and Communication Engineering*, 4(6), pp. 446-452.

[9] R. Di Pietro and L. V. Mancini, eds. 2008. Intrusion Detection Systems. *Springer Science & Business, vol. 38.* Media.

[10] Osama Faker. 2018. Intrusion Detection Using Big Data and Deep Learning Techniques. *MS Thesis, Cankaya University.*

[11] J.H. Friedman. 2002. Stochastic Gradient Boosting. *Computational Statistics & Data Analysis*, 38(4), pp. 367-378.

[12] H. Gharaee and H. Hosseinvand. 2016. A New Feature Selection IDS based on Genetic Algorithm and SVM. *Telecommunications (IST), 2016 8th International Symposium on. IEEE*, pp. 139-144.

[13] G.P. Gupta and M. Kulariya. 2016. A Framework for Fast and Efficient Cyber Security Network Intrusion Detection Using Apache Spark. *Procedia Computer Science* 93, Kochi, India, pp. 824-831.

[14] J. Han, E. Haihong, G. Le, and J. Du. 2011. Survey on NoSQL Databases. *In Pervasive Computing and Applications (ICPCA)*, Port Elizabeth, South Africa 2011 6th International Conference on, pp. 363-366. IEEE.

[15] A. Lashkari, G. Draper-Gil, M. Mamun, and A. Ghorbani. 2017. Characterization of Tor Traffic Using Time based Features. *The 3rd International Conference on Information Systems Security and Privacy*, pp. 253-262.

[16] Y. Liu. 2014. Random Forest Algorithm in Big Data Environment. *Computer Modelling & New Technologies*, 18(12A), pp. 147-151.

[17] N. Moustafa and J. Slay. 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(13), pp. 18-31.

[18] N. Moustafa and J. Slay. 2015. UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set). *Military Communications and Information Systems Conference (MilCIS)*, Canberra, Australia, pp. 1-6, IEEE.

[19] N. Moustafa and J. Slay. 2018. 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), pp. 18-31.

[20] R. Primartha and B. Tama. 2017. Anomaly Detection Using Random Forest: A Performance Revisited. *Data and Software Engineering (ICoDSE), International Conference on*, Palembang Sumatra Selatan, Indonesia, pp. 1-6, IEEE.

[21] P. Resende and A. Drummond. 2018. Adaptive Anomaly-based Intrusion Detection System Using Genetic Algorithm and Profiling. *Security and Privacy*, e36, pp. 1-13.

[22] A. Rosenberg and J. Hirschberg. 2007. V-measure: A Conditional Entropy-based External Cluster Evaluation Measure. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (EMNLP-CoNLL), pp. 410-420.

[23] J. Schmidhuber. 2015. Deep Learning in Neural Networks: An Overview. *Neural Networks*, vol. 61, pp. 85-117.

[24] I. Sharafaldin, A. Lashkari, and A. A. Ghorbani. 2018. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018).* Funchal, Madeira-Portugal, pp. 108-116.

[25] I. Sharafaldin, A. Gharib, A. H. Lashkari, and A. A. Ghorbani. 2018. Towards a Reliable Intrusion Detection Benchmark Dataset. *Software Networking*, 2018(1), pp. 177-200.

[26] K. Shvachko, H. Kuang, S. Radia, and R. Chansler. 2010. The Hadoop Distributed File System. *Mass Storage Systems and Technologies (MSST), IEEE* 26th symposium on, pp. 1-10.

[27] O.B. Sezer, M. Ozbayoglu, E. Dogdu. 2017. A Deep Neural-Network Based Stock Trading System Based on Evolutionary Optimized Technical Analysis Parameters. *Procedia Computer Science*, 114, pp. 473-480.

[28] S. Suthaharan. 2014. Big Data Classification: Problems and Challenges in Network Intrusion Prediction with Machine Learning. *ACM SIGMETRICS Performance Evaluation Review* 41(4), pp. 70-73.

[29] M. Tavallaee, E. Bagheri, W. Lu, and A. A.Ghorbani. 2009. A Detailed Analysis of the KDD CUP 99 Data Set. *In Computational Intelligence for Security and Defense Applications.* CISDA 2009. IEEE Symposium on, pp. 1-6, IEEE.

[30] A. Thusoo, et al.2009. Hive: A Warehousing Solution over a Map-Reduce Framework. *Proceedings of the VLDB Endowment* 2(2), pp. 1626-1629.

[31] E.D. Ubeyli and E. Dogdu. 2010. Automatic Detection of Erythemato-squamous Diseases Using K-means Clustering. *Journal of Medical Systems*, 34(2), pp. 179-184.

[32] R. Vijayanand, D. Devaraj, and B. Kannapiran. 2018. Intrusion Detection System for Wireless Mesh Network Using Multiple Support Vector Machine Classifiers with Genetic-Algorithm-based Feature Selection. *Computers & Security* 77, pp. 304-314.

[33] M. Zaharia, et al. 2016. Apache Spark: A Unified Engine for Big Data Processing. *Communications of the ACM* 59(11), pp. 56-65.

[34] C. Zhang and Y. Ma, eds. 2012. Ensemble Machine Learning: Methods and Applications. *Springer Science & Business Media*, Springer.

[35] P. Zikopoulos and C. Eaton. 2011. Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data. *McGraw-Hill Osborne Media.*

[36] R. Zuech, T. M. Khoshgoftaar, and R. Wald. 2015. Intrusion Detection and Big Heterogeneous Data: A Survey. *Journal of Big Data*, 2(3), pp. 1-41.