Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IMPACT FACTOR: 7.056

IJCSMC, Vol. 10, Issue. 5, May 2021, pg.113 – 124

Intrusion Detection System using Deep Neural Networks and Principal Component Analysis

Abdoulwase M. Obaid Al-Azzani[1]; Ali Mohammed Afif[2]

[1]Associate Prof. Dept. of Computer Science and Information Technology, Sana' University – Yemen

[2]Student, Department of Computer Science and Information Technology, Sana' University – Yemen

DOI: 10.47760/ijcsmc.2021.v10i05.012

Abstract— The amount of data on the internet is growing daily, because of using the smart phones, social network, and IoT. This growing had impact on data security. Therefore security become the biggest challenge for researchers and developers, moreover, most of security tools (firewall, IDS, IPS, etc.) have limitations to detect all threats. Deep Learning is one of Machine Learning approach, it is an efficient artifice that can be applied to intrusion detection, to ascertain a new outline from the massive network data, as well as it used to reduce the strain of the manual compilations of the normal and abnormal behaviour patterns. In this paper we build a model for detect threats based on Principal Component Analysis (PCA) for reduction dimensions of dataset, and deep neural network for the classification. We used KDD99 dataset and 10_ percentage of KDD99 dataset to train and test the model in Spark environment. In experiment DNN of layers ranging from 1 to 3 with 300 number of epotch. The results were compared and concluded that a DNN of 1 layer has superior performance with 0.929 as accuracy.

Keywords— Deep Neural Networks (DNN), Deep Learning (DL), Principal Component Analysis (PCA), Big Data, Spark

I. Introduction

The field of computing is rapidly developed in the last decade; the development came as result of the huge expansion of data which came from many sources such as Internet networks and the applications of the social media. The traditional technics of computing were not wildly helpful for collecting the data and its security, which consider to be the most important challenges for industries and academic institution, [1].

This expansion and growth of the data increases the opportunities for hackers to use developed techniques and tools for hacking and attacking the security of data maliciously. The researchers and developers of the security systems seek to increase the effectiveness of defensing, detecting and the predicting the malicious attacks early and before they happen. Intrusion Detection Systems (IDS) are the first internal defense systems that are used in cyber security. The term Intrusion refers to attempts and compromises the confidentiality, integrity, availability of security mechanisms of computer or network resources or bypass them [2],[3].

Therefore, (IDS) technology was rapidly developed after falling into a relatively slow period when the professor Hinton [4] proposed the theory of the deep learning in 2006. The deep learning theory and technology underwent a meteoric rise in the field of machine learning. In the scenario related to theoretical papers and practical research findings emerged endlessly and provided remarkable achievements, especially in the fields of recognition speech, image and action [5], [6].

As a matter of fact, in recent years, the theory of the deep learning and technology has been rapidly developed and leads to a new era of artificial intelligence that opened a new way to develop intelligent intrusion detection technology[6]. The most popular deep learning techniques used is DNN; it is contain an attractive essential intrusion detection function called learning by training, to deduce new information to provide a decision, this makes DNNs distinguishes from all the traditional programming techniques and make it an expert system

Contents of the paper is organized as follows: Section 2 presented the Big data environment architectures, Deep learning classifications, principal component analysis as selection features, KDD99 as Dataset and the literature review of IDS for Big Data presents related work and background: section 3 introduced the Spark-DNN-IDS model for intrusion detection that can deal with big data. The proposed model used Spark's big data platform which can process and analyse data. The Section also displayed steps of the Spark-DNN-IDS model that are load data pre-processing steps, features selection, and finally model classifier; section 4 displayed experiment steps and its results , section 5 presented the conclusions , future work, references.

II. RELATED WORK AND BACKGROUND

A. AN BIG DATA COMPUTING ENVIRONMENT

The good knowledge about several fundamental technologies are closely associated with big data, these technologies play an important role in analyzing and storing these huge datasets. Secondly, the Big data technology must support search, development, and analysis services for all data types from transaction and application data to machine and sensor data to image, social data, geospatial data, and others. Emerging technologies such as Map-reduce, Apache Spark model, and Spark's MLlib are designed to address the characteristics of the big data[7]

- *MAP-REDUCE MODEL*

Hadoop is a processing framework used to support the processing of large datasets in distributed computing. Map-Reduce acts as a strong pillar in the Hadoop ecosystem. It is a distributed and parallel programming model which enables the processing of the Big Data in a cluster computing environment. Map-Reduce programming mainly consists of two functions: the map and reduce. Mapper provides the mapping of input data according to the number of input partitions provided to the worker node and generates a Key-Value pair as an output. The sort and shuffle phase provides the sorting of the data according to the given key input and generates the format readable for the reducer. The reducer phase takes the input of these intermediate data and makes the transformations on the values for a given key value and generates the required output. [8]. [9],[10];

- *APACHE SPARK MODEL*

Apache Spark is a new framework that provides improved alternatives to the Map-Reduce model. Unlike the Map-Reduce model, the Spark model does not flush out the data to the disk in each step, but the data is processed in the memory until the memory becomes full. Once the memory is filled, it spills over the data to the hard disk. Hence, the spark model can also be considered as memory processing. This advantage of the spark model makes its processing very quickly compared to the map-reduce models.

The spark model is generally referred to be 10x times faster than the map reduce models. The spark framework can be implemented in various forms such as Standalone, on top of Hadoop Yarn (cluster manager), Cassandra or HBase. This formulae is another advantage of the spark model which unlike the map-reduce doesn't require HDFS (Hadoop Distributed File System) to run [11].

B. DEEP LEARNING

The term deep learning comes from the advancements of neural networks. In deep learning, various methods have been applied to overcome the limitations of the hidden layer. Those methods employ successive hidden layers which are hierarchically structured. Since a lot of methods belong to the deep learning method, the classification of each deep learning method is essential. The deep learning is classified into three sub-groups according to Deng: generative, discriminative, and hybrid. The classification is based on the intention of architectures and techniques, like synthesis/generation or recognition/classification [12]. The Classification of the deep learning methods it will be covered briefly in the following paragraph:

- *CLASSIFICATION OF THE DEEP LEARNING METHODS*

There are three types of Deep learning methods as following:

a) GENERATIVE (UNSUPERVISED) LEARNING

Generative Learning is also known as "Unsupervised learning" in which the data are unlabeled. The main concept of applying generative architectures to pattern recognition is unsupervised learning or pre-training since learning the lower levels of subsequent networks is difficult that is why the deep generative architectures are required. Therefore, with limited training data, learning each lower layer in a layer-by-layer approach without relying on all above layers is important. There are several methods that classified as unsupervised learning as they are presented in the follows

- Auto Encoder (AE)
- Boltzmann Machine (BM)

b) SUPERVISED LEARNING

The aim of supervised learning or discriminative deep architectures is to distinguish some parts of data for pattern classification. An example of the discriminative architecture is (CNN) which employs a special suitable architecture particularly for recognizing images. One advantage of (CNN) is that it is fast to train because of its structure. CNN can train multilayer networks with gradient descent to learn complex, high-dimensional, nonlinear mappings from large collections of data. CNN uses three basic concepts: local receptive fields, shared weights, and pooling. One extensive research that successfully deployed using CNN is Alpha Go by Google[13].

c) HYBRID

The deep architectures of hybrid combine both generative and discriminative architectures. The hybrid architecture aims to distinguish data as well as discriminative approach. However, in the early step, it has assisted in a significant way with the generative architectures results. An example of the hybrid architecture is Deep Neural Network (DNN) although the terms DNN and DBN are confused. In the open literatures, (DBN) also uses back propagation discriminative training as a "fine-tuning". This concept of DBN is really similar to Deep Neural Network (DNN) [14]. DNN is defined according to Deng[14] as a multilayer network with cascaded fully connected hidden layers, and is often use stacked RBM as a pre-training phase.

- *DEEP NEURAL NETWORK (DNN)*

While the traditional machine learning algorithms are linear, the deep neural networks are stacked in an increasing hierarchy of complexity as well as abstraction. Each layer applies a nonlinear transformation onto its input and creates a statistical model as output from what it learns. In simple terms, the input layer is received by the input layer and passed onto the first hidden layer. These hidden layers perform mathematical computations on the inputs. One challenge facing creating neural networks is deciding the hidden count of the layers and the count of the neurons for each layer. Each neuron has an activation function that is used to standardize the output from the neuron. The term Deep learning refers to having more than one hidden layer. The output layer returns the output data until the output has reached an acceptable level of accuracy, epochs are continued. Figure1 explains how a Deep Neural Network works.

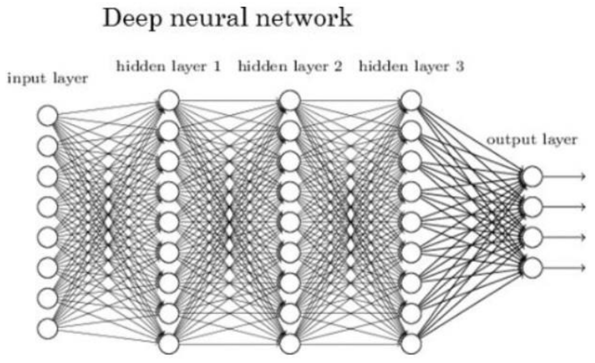

FIGURE 1: Deep Neural Network

C) Related Work

Gupta and Kulariya [11] proposed a framework that combines feature selection algorithms and classification algorithms, and their implementation into the big data processing environment of Apache Spark. The Correlation-based on 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 the result showed the highest accuracy approximately for Logistic regression based scheme.

Suad Mohammed, et al, [15] proposed a model for intrusion detection by using a support vector machine (SVM) classifier on Apache Spark Big Data platform. Using Chi Sq Selector for feature selection, trained and tested the model within the KDD99 dataset, the proposed model introduced a comparison between Chi-SVM classifier and Chi-Logistic Regression classifier. The results of the experiment showed that the Spark-Chi-SVM model has high performance, reduces the training time, and is effective for Big Data.

Vimal Kumar K & Radhika N [16] proposed a compared model for machine learning techniques such as deep neural networks, support vector machines, random forest, decision trees, and Naïve Bayes classifications had done for the Synchrophasing or dataset, the results are compared using metrics of accuracy, recall, false rate, specificity, and prediction time. Feature selection and dimensionality reduction algorithms are used to reduce the prediction time taken by the proposed approach. The results showed that the deep neural network provides the highest accuracy for the raw dataset. Baluch, et al [17] evaluated the performance for a set of classification algorithms SVM, Decision Tree, Naive Bayes, and Random Forest using Apache Spark for processing and UNSW-NB15 dataset for trained with all 42 features and concluded with the best performance for Random Forest (97% accuracy).

The researchers still seek to find an effective way to detect the intrusions with high performance, high speed, and a low false-positive alarm rate. The main objective of the current study is to improve the performance and speed of intrusion detection within a Big Data environment. In this method, the researchers used Apache Spark Big Data tools because it is 100 times faster than Hadoop[36], the feature selection that takes the amount of computation time which can be reduced when using DNN on KDD 99 datasets. Therefore, the DNN algorithm was used with PCA for feature selection to build and test the model in this paper

III. THE KDD99 DATA SET

The most known and widely used dataset for experiments on anomaly detection in computer networks is the KDD Cup '99 datasets. The KDD Cup '99 datasets are a collection of data transfer from a virtual environment to be used for the Competition of the Third Knowledge Discovery and Data Mining Tools (KDD CUP '99 datasets, 1999). It is the subset of the 1998 DARPA dataset that was collected by simulation of the operation of a typical US Air Force LAN with multiple attacks and acquired nine weeks of TCP dump data. The dataset was collected and distributed at the Massachusetts Institute of Technology (MIT) Lincoln Laboratory. The KDD Cup '99 intrusion detection benchmark consists of three components[18].

The KDD99 data set is used to evaluate the proposed model. The numbers of instances that are used are equal to 494,021. The KDD99 dataset has 41 attributes and the "class" attributes which indicate whether the given instance is a normal instance of an attack. Table 1 describes KDD99 dataset attributes. The majority of published results were tested and trained with only a 10% training set observing the feature reduction on the KDDCup-'99' datasets.

TABLE 1 KDD99 DATASET ATRIBUTES

No.	Attributes	No	Attributes	No	Attributes
1	Duration	15	su_attempted	29	same_srv_rate
2	protocol type	16	num_root	30	diff_srv_rate
3	Service	17	num_file_creations	31	srv_diff_host_rate
4	Flag	18	num_shells	32	dst_host_count
5	src_bytes	19	num_access_files	33	dst_host_srv_count
6	dst_bytes	20	num_outbound_cmds	34	dst_host_same_srv_rate
7	Land	21	is_host_login	35	dst_host_diff_srv_rate
8	wrong_fragment	22	is_guest_login	36	dst_host_same_src_port_rate
9	Urgent	23	count	37	dst_host_srv_diff_host_rate
10	Hot	24	srv_count	38	dst_host_serror_rate
11	num_failed_logins	25	serror_rate	39	dst_host_srv_serror_rate

12	logged_in	26	srv_serror_rate	40	dst_host_rerror_rate
13	num_compromised	27	rerror_rate	41	dst_host_srv_rerror_rate
14	root_shell	28	srv_rerror_rate		

IV. Feature Selection

Feature selection is a very efficient way to reduce the dimensionality of the problem. Redundant and irrelevant variables are removed from the data before being fed to the machine learning algorithm which is used as a classifier. Feature selection is a preprocessing step that can be an independent choice from the learning algorithm or a part of it. It can be used to improve the computational speed with a minimum reduction of accuracy. Other advantages of this way that it reduces noise and increasing robustness against overfitting as well as reduces drastically the variance. Generally, automatic selection of features works much better than manual selection because the algorithm can find correlations between the features that are not always obvious even for a human expert. For example, as Guyton pointed out that the variable that is completely useless by itself can provide a significant performance improvement when taken with others[19].

- ***Principal Component Analysis (PCA)***

Principal Component Analysis is one of the most widely used dimensionality reduction techniques for analyzing and compressing the data. (PCA) is a useful statistical technique that has found application in fields, such as face recognition and image compression. It is a common technique for finding patterns in data of high dimension and it is depending on transforming a relatively large number of variables into a smaller number of uncorrelated variables by finding a few orthogonal linear combinations of the original variables with the largest variance.

The first principal component of the transformation is the linear combination of the original variables with the largest variance and the second principal component is the linear combination of the original variables with the second-largest variance and orthogonal to the first principal component and so on. In many data sets, the first several principal components contribute most of the variance in the original data set, so that the rest can be disregarded with minimal loss of the variance for dimension reduction of the data[20]. (PCA) reduces the number of dimensions required to classify new data and produces a set of principal components which are orthonormal eigenvalue/eigenvector pairs. The steps for doing principal component analysis are given below:

Algorithm:

Suppose $x_1, x_2, \dots x_M$ are Nx1 vectors

1. Get input data

$$\overline{x} = \frac{1}{M}\sum_{i=1}^{M} x_i \qquad (1)$$

2. Subtract the mean

$$\Phi_i = x_i - \overline{x} \qquad (2)$$

3. From the matrix $A = \Phi_1\ \Phi_2 \dots \Phi_m$ Calculate the covariance

$$C = \frac{1}{M}\sum_{N=1}^{M} \Phi_n\, \Phi_n = \mathrm{AA}^{\mathrm{T}} \qquad (3)$$

4. Calculate the eigenvectors and eigenvalues of the covariance matrix.

$$C: \lambda_1 > \lambda_2 > \cdots > \lambda_N \qquad (4)$$

5. Sort the Eigenvalues in descending order.

$$(x - \overline{x}) = b_1 u_1 + b_2 u_2 + \cdots + b_N u_N = \sum_{i=1}^{N} b_i\, u_i \qquad (5)$$

6. Calculate the feature vectors

$$x - \overline{x} = \sum_{i=1}^{K} b_i\, u_i \qquad (6)$$

To choose K, use the following criterion :

$$\frac{\sum_{i=1}^{K} \lambda_i}{\sum_{i=1}^{N} \lambda_i} > \text{Threshold (e.g., 0.9 or 0.95)} \qquad (7)$$

V. SPARK-DNN-IDS MODEL

In this section, The proposed intrusion detection system discover the attacks by using Principal Component Analysis and a deep neural network algorithm. We describe the tools and techniques used in the proposed method. Figure 2 shows Spark-DNN-IDS model. The steps of the proposed model can be summarized as follows:

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

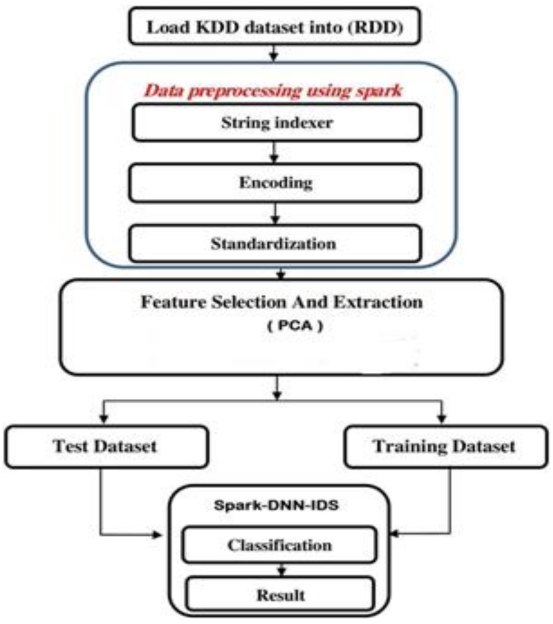

Figure 2 Spark -DNN -IDS model The sequence of steps that in Spark-DNN-IDS model

- ***Deep Neural Network Algorithm***

Deep Neural Network (also called Artificial Neural Network) is inspired by the human neural system and it is used in different areas like pattern recognition, optimization, control, etc. The neural network is composed of several processing units (nodes) and directed links between them. These connections are weighted representing a relation between input and output neurons[21].

- ***Application of Rectified Linear Units (ReLU)***

ReLu has turned out to be more effective and has the capacity to accelerate the entire training process altogether. Usually, the Neural networks use a sigmoidal activation function or the (hyperbolic tangent) activation functions. But these functions are prone to vanishing gradient problems. Vanishing gradient occurs when lower layers of the DNN have gradients of nearly null because the units of higher layers are nearly saturated at the asymptotes of the function. ReLU offers an alternative to sigmoidal non-linearity which addresses the issues mentioned so far [22].

- ***Feed forward Neural Networks***

The Multilayer Perceptron (MLP) architecture is the most popular paradigm of artificial neural networks that is used today. The neural network architecture shares a common feature that all neurons in a layer are connected to all neurons in adjacent layers through unidirectional branches. The branches and links can only broadcast information in one direction, that is, the "forward direction". The branches have associated weights that can be adjusted according to a defined learning rule Fig *3* shows a standard multilayer feed-forward network with five layers (three hidden).

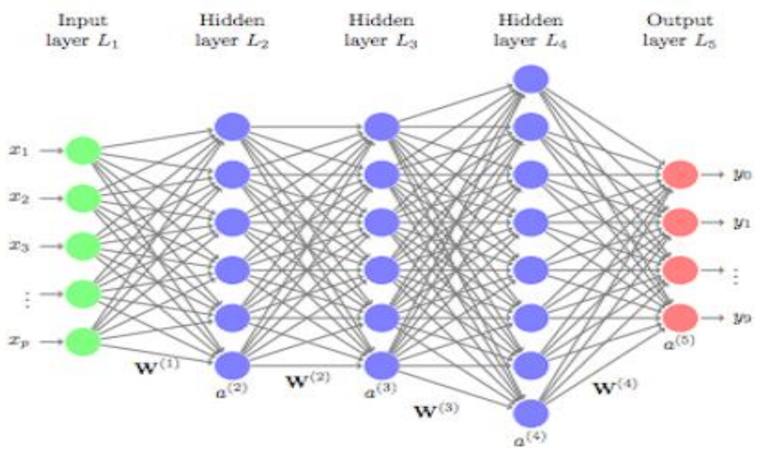

Figure 3: Architecture of feed-forward neural network

The Feed forward neural network training is usually carried out using the called back propagation algorithm. Training the network with back propagation algorithm results in a non-linear mapping between the input and output variables. Thus, given the input/output pairs, the network can have its weights adjusted by the back propagation algorithm to capture the non-linear relationship. After training, the networks with fixed weights can provide the output for the given input. In our model Spark-DNN-IDS, the one hidden layers were used. The standard back propagation algorithm for training the network is based on the minimization of an energy function representing the instantaneous error. In other words, the researcher hopes to minimize a function defined as the following formula.

$$E(m)=\frac{1}{2}\sum_{q=1}^{q}\left[d_q-y_q\right]^2 \quad (11)$$

Where d_q represents the desired network output for the qth input pattern and yq is the actual output of the neural network. Each weight is changed according to the rule:

$$\Delta w_{ij}=-k\frac{dE}{dw_{ij}} \quad (12)$$

where k is a constant of proportionality, E is the error function and w_{ij} represents the weights of the connection between neuron j and neuron i. The weight adjustment process is repeated until the difference between the node output and actual output is within some acceptable tolerance.

In the Spark-DNN-IDS model, the input-layer consists of 19 neurons. The neurons in the input-layer to hidden-layer and hidden to output-layer are connected completely. The back-propagation mechanism is used to train the DNN networks. The proposed network is composed of fully connected layers, bias layers, and dropout layers to make the network more robust. Input and hidden layers: This layer consists of 19 neurons. These are then fed into the hidden layers. Hidden layers use ReLU as the non-linear activation function. Then weights are added to feed them forward to the next hidden layer. The neuron count in each hidden layer is decreased steadily from the first to the output to make the outputs more accurate and at the same time reducing the computational cost. Regularization: To make the whole process effective and time-saving, Dropout (0.01). The function of the dropout is to unplug the neurons randomly, making the model more robust and hence preventing it from overfitting the training set. Output layer and classification: The out layer consists only of two neurons Attack and Benign. Since the 1024 neurons from the previous layer must be converted into just 2 neurons, a sigmoid activation function is used. Due to the nature of the sigmoid function, it returns only two outputs, hence favoring the binary classification that was intended in this paper.

- *Evaluation Matrices*

For evaluation purposes, Precision (P), Recall (R), F-measure (F), and Accuracy (ACC) metrics are used. These metrics are calculated by using four different measures, true positive (TP), true negative (TN), false positive (FP), and false-negative (FN):

- TP: the number of anomaly records correctly classified.
- TN: the number of normal records correctly classified.
- FP: the number of normal records incorrectly classified.
- FN: the number of anomaly records incorrectly classified.

Accuracy (AC): the percentage of true detection over total traffic record

$$AC = \frac{TP+TN}{TP+TN+FP+FN} \qquad (13)$$

Precision (P): the percentage of predicted anomalous instances predicted are actual anomalous instances,

$$P = \frac{TP}{TP+FP} \qquad (14)$$

Recall (R): the percentage of predicted anomalous instances versus all the anomalous instances presented,

$$R = \frac{TP}{TP+FN} \qquad (15)$$

F1 score - F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar costs. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall. In our case, F1 score is:

$$f(\beta) = (1+\beta^2).\left(\frac{PPV.TPR}{\beta^2.PPV+TPR}\right) \qquad (16)$$

VI. Experiments and Result Analysis

The results of the Spark –DNN-IDS model for intrusion detection were presented in this section. The proposed model was implemented in python programming using the Sklearn deep learning library, Keras as a wrapper on top of Tensor- Flow [43] as a software framework. For increasing the validity of processing the data in deep-learning architectures in Apache Spark, tests were conducted on a personal computer with 2.0GHZ CORE™ i3 CPU and 6 GB of memory under windows10, the KDD 99 data set was used to evaluate the results. The experiment and implementation were divided into multi-steps and each step was displayed and compared as the following:

a) Load KDD into RDD

The result of this step showed that, the KDD data was load into memory and partition on cluster nodes for the analysis process.

b) Data Preprocessing

Data pre-processing was done to clean and for making the data more standardized, the following libraries were used:

1. Pandas [44] offers better data manipulation and visualization to perform some analysis.
2. NumPy [45] (numerical python), offering a set of mathematical and numerical features, which were used beside the pandas for manipulating the data.
3. Scikit-learn [46] :It is also known as Sklearn, it is used to provide the sklearn preprocessing package that offers common utility functions, it was used for feature scaling and data encoding.

The output of these steps is numerical data as shown in table 2 which is used in the next steps for reduction features.

Table 2: The Record of Dataset after Standardization:

The Record Before Standardization	res1: org.apache.spark.mllib.regression.LabeledPoint = (1.0,[0.0,181.0,5450.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,8.0,8.0,0.0, 0.0,0.0,0.0,1.0,0.0,0.0,9.0,9.0,1.0,0.0,0.11,0.0,0.0,0.0,0.0,0.0])
The Record After Standardization	res2: org.apache.spark.mllib.regression.LabeledPoint = (1.0,[0.0,1.8315794844034117E4,0.16495156759878019,0.0,0.0,0.0,0.0,0.0,2.814168444874875,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.03753270996838475,0.03247770581832668,0.0,0.0,0.0,0.0,2.576061480099788,0.0,0.0,0.13900605646702138,0.0848732827397667,2.434387313317322,0.0,0.22854329046843286,0.0,0.0,0.0,0.0,0.0

c) Feature Selection

Principal Component Analysis reduces the dimensionality of data by restricting attention to those directions in the feature space in which the variance is greatest. In PCA, the proportion of the total variance accounted by a feature is proportional to its eigenvalue. The PCA was performed on a 10% training data subset that aimed to reduce the cardinality of the dimensions (d) in the data where (d = 41). Two features were excluded: (1) the number of outbound commands and (2) is host login as their values remained constant throughout the dataset, reducing (d to 39). The correlation matrix for the training dataset was computed, then the eigenvalues and sort them in decreasing order was also computed. The first eigenvalue e1 corresponds to the first principal component and the second eigenvalue e2 corresponds to the second principal component and so on. Table 3 shows the first 20 eigenvalues that arranged in decreasing order.

Table 3 : The first 20 features and corresponding eigenvalues and standard deviations (S.D). The first 19 features are selected based on the results of the Scree test and Critical Eigenvalue test

Feature Name	S. D	Eigen Value
src bytes	51926	9.74
dst bytes	29423	4.75
Duration	773.65	3.65
is guest login	247.93	2.76
is host login	217.9	1.87
srv diff host rate	106.64	1.60
diff srv rate	69.196	1.27
Service	1.2974	1.14
Flag	1.1499	1.11
protocol type	0.9664	1.04
num root	0.7784	1.02
Hot	0.7769	1.00
num compromised	0.6720	0.96
dst host same srv rate	0.4849	0.95
dst host count	0.4128	0.88
rerror rate	0.3886	0.85
srv count	0.3816	0.82
dst host srv diff host rate	0.3815	0.76
Count	0.3813	0.74
dst host same src port rate	0.3812	0.45

In the second step, the actual features in KDD Cup 1999 datasets that correspond to the 19 significant principal components was found. Since the principal components explain almost all the variance in the data, the 39 original features in the decreasing order of their standard deviation (shown in Table 3) was sorted and the first 19 features as the features that correspond to the 19 principal components was selected.

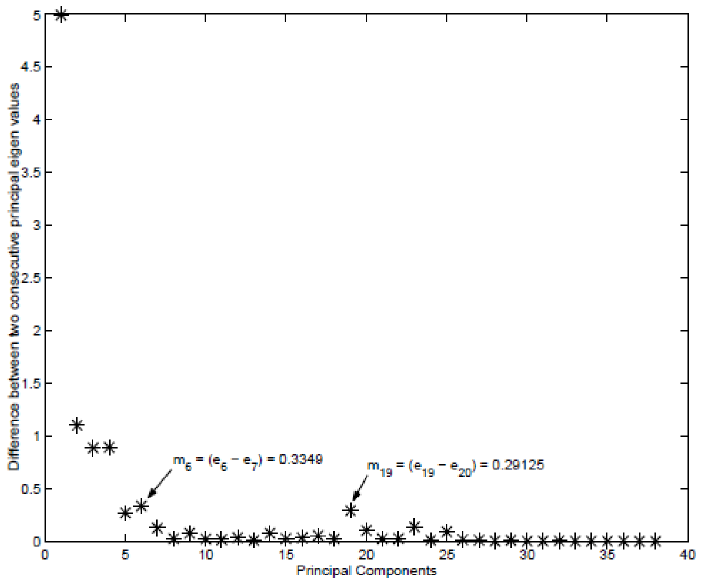

Figure 4. Scree plot illustrates the difference in eigenvalues of the sixth and seventh (m6), and nineteenth and twentieth (m19) principal components.

The principal components against the differences mi in successive sorted eigenvalues (e.g., mi = ei − ei+1) were plotted. The Scree plot in Figure 4 shows principal components plotted against the successive differences in eigenvalues shown in Table 3 In this plot, the difference between successive eigenvalues decreases regularly from (4.9918 to 0.2707) for the first six principal components. Then there is an increase in the difference between the sixth and seventh eigenvalues (m6 = e6 −e7 = 0.3349), breaking the decreasing trend. The difference again decreases regularly from the seventh to the nineteenth principal component. Then, there is a break in decreasing trend between the nineteenth and twentieth principal components (m19 = e19 − e20 = 0.2912), suggesting that either the first six components or the first nineteen components are the most significant components in the data.

d) Model Classifier

The KDDCup-’99’ dataset after reduction features was fed into DNNs of varying hidden layers. After the training is completed, all models were compared for accuracy, recall, precision and f1-score, with the test dataset. The scores for the same have been compared in detail in Table 4.

Table 4 Result of classification Experiments

Algorithm	Accuracy	Recall	Precision	F1score
PCA+ DNN 1 Hidden layer 300 time epotch	0.929	0.925	0.985	0.954
PCA+ DNN 2 Hidden layer 300 time epotch	0.795	76.6	97.4	0.857
PCA+ DNN 3 Hidden layer 300 time epotch	0.562	0.464	0.983	0.63

The DNN rate (0.1) of learning is applied and is run many times for 300 number of epochs and KDDCup-’99’ dataset has been used for training and benchmarking the network. For comparison purposes, DNN of layers ranging from 1 to 3.

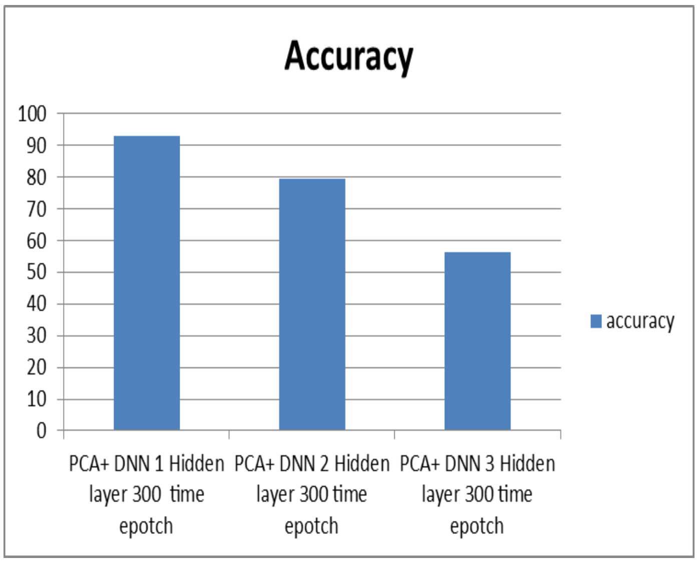

Figure 5. Result of Accuracy

The findings of the Experiments were as the following:

- For PCA + DNN 1 Hidden layers for 300-time epoch and 0.929 was gotten as accuracy, 0.925 as recall, 0.985 as precision and 0.954 as f1 score.
- For PCA + DNN 2 Hidden layers for 300-time epoch and 0.795 was gotten as accuracy, 76.6 as recall, 97.4 as precision and 0.857 as f1 score.
- For PCA + DNN 3 Hidden layers for 300-time epoch and 0.562 was gotten as accuracy, 0.464 as recall, 0.983 as precision and 0.63 as f1 score.

Finally, the findings of the experiment showed that, the more adding number of hidden layers was, the more difficulty of learning would be. Therefore, the best performance was shown by PCA+DNN 1 layer with 0.929 accuracy compared to the others. The detailed statistical results for different network structures are reported in table 4 and Figure 5.

VII. CONCLUSION

The study aimed to recapitulate the usefulness of DNNs in IDS comprehensively. The KDD Cup 99 dataset was used for the experiment, some redundant was found, and irrelevant data like noise data in KDD Cup 99 was used. To remove unimportant data effectively, the dimension reduction technique which is the PCA algorithm was combined. The goal of PCA is to reduce the noise data effectively to get a smaller dimensionality dataset from KDD Cup 99 dataset. Therefore, the deep neural network for classification normal or attack was used. The findings of the experiment showed that, the performance of the model was high and the model can be extended to be multi-classes model to detect types of attacks in future.

REFERENCES

[1] D. Bank, "Big Data."
[2] O. Faker, "Intrusion Detection Using Big Data and Deep Learning Techniques," pp. 86–93, 2019.
[3] B. Clicknet, C. Idc, I. S. S. Nai, and O. D. S. Tripwire, "Intrusion Detection Systems."
[4] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning", 2015.
[5] B. Y. Zhang, M. Ieee, T. Cao, M. Ieee, S. Li, and X. Tian, "Parallel Processing Systems for Big Data : A Survey," 2016.
[6] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks," vol. 5, 2017.
[7] M. Mridul, A. Khajuria, S. Dutta, and N. Kumar, "Analysis of Bidgata using Apache Hadoop and Map Reduce," vol. 4, no. 5, pp. 555–560, 2014.

[8] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, “The Hadoop Distributed File System,” pp. 1–10, 2010.
[9] M. Essid and F. Jemili, “Combining intrusion detection datasets using MapReduce,” pp. 4724–4728, 2016.
[10] L. Wang, “Big Data in Intrusion Detection Systems and Intrusion Prevention Systems,” vol. 4, no. 1, pp. 48–55, 2017.
[11] G. P. Gupta and M. Kulariya, “A Framework for Fast and Efficient Cyber Security Network Intrusion Detection using Apache Spark,” *Procedia - Procedia Comput. Sci.*, vol. 93, no. September, pp. 824–831, 2016.
[12] M. Erza and K. Kim, “Deep Learning in Intrusion Detection System : An Overview,” pp. 1–12.
[13] L. Mohammadpour, T. C. Ling, C. S. Liew, and C. Y. Chong, “A Convolutional Neural Network for Network Intrusion Detection System,” pp. 50–55, 2018.
[14] J. Ahmad, H. Farman, and Z. Jan, “Deep Learning Methods and Applications,” *SpringerBriefs Comput. Sci.*, pp. 31–42, 2019.
[15] S. M. Othman, F. Mutaher, B. Alwi, N. T. Alsohybe, and A. Y. Al Hashida, “Intrusion detection model using machine learning algorithm on Big Data environment,” *J. Big Data*, 2018.
[16] K. Vimalkumar and N. Radhika, “A Big Data Framework for Intrusion Detection,” pp. 198–204, 2017.
[17] M. Belouch, S. El Hadaj, and M. Idlianmiad, “Performance evaluation of intrusion detection based on machine learning using apache-spark,” *Procedia Comput. Sci.*, vol. 127, pp. 1–6, 2018.
[18] D. Protić, “Review of KDD Cup ’99, NSL-KDD and Kyoto 2006+ datasets,” *Vojnoteh. Glas.*, vol. 66, no. 3, pp. 580–596, 2018.
[19] K. P. Singh, N. Basant, and S. Gupta, “Support vector machines in water quality management,” *Anal. Chim. Acta*, vol. 703, no. 2, pp. 152–162, 2011.
[20] D. Subramanyam, “Classification of Intrusion Detection Dataset using machine learning Approaches,” *Proc. Int. Conf. Comput. Tech. Electron. Mech. Syst. ITEMS 2018*, pp. 280–283, 2018.
[21] S. J. R. and P. Norvig, *Artificial Intelligence A Modern Approach*, no. 1. 2002.
[22] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier nonlinearities improve neural network acoustic models,” *ICML Work. Deep Learn. Audio, Speech Lang. Process.*, vol. 28, 2013.