A NOVEL FEATURE SELECTION WITH FUZZY DEEP NEURAL NETWORK FOR ATTACK DETECTION IN BIG DATA ENVIRONMENT

B. Vijaya Kumar
[1]Research Scholar, Department of Computer Science and Engineering,
Annamalai University, Tamilnadu, India.
vijaymtech11@gmail.com

Dr. S. Mohan
[2]Assistant Professor, Department of Computer Science and Engineering,
Annamalai University, Tamilnadu, India.
mohancseau@gmail.com

Abstract - In recent times, the massive quantity of data and its continual expansion have transformed the significance of information security and data analysis systems for Big Data. An intrusion detection system (IDS) is commonly employed to monitor and analyze data for the detection of intrusions in the network. The conventional IDS models are not adequate to handle the high volume, variety, and speed of big data. This paper presents a new quantum brain storm optimization (QBSO) based feature selection with fuzzy deep neural network (FDNN), called QBSO-FDNN model for IDS in big data environment. The proposed model enables to detection of intrusions in the big data environment. The presented model initially performs preprocessing to enhance the quality of the big data. Also, to reduce the computational complexity, QBSO algorithm is applied to elect an optimal set of features. The choice of optimal features by the QBSO algorithm helps to boost the detection performance. Besides, FDNN model is applied as a classification model for identifying the occurrence of intrusions in the network. An extensive set of simulations was carried out to highlight the results on benchmark dataset. The resultant experimental values showcased the superior performance of the QBSO-FNN model with the detection accuracy of 98.90%.

Keywords: Intrusion Detection System, Big Data, Hadoop.

1. The Main Text

The word Big Data defines enormous gathering of distinct data structures attained from several heterogeneous sources packed on memories where information is deliberate in Petabytes and zeta bytes. The 5 different dimensions are related by Big Data, classified as five Vs representing Variety, Volume, Veracity, Value, and Velocity [1]. All these dimensions have a vital part to act as Big Data Management Systems (BDMS) that is a modern term for handling Big Data Systems (BDS) [2]. Currently, information is investigated to resolve the secret pattern, unknown correlation, market trend, and several effective data to support industry and administrations in developing additional cognizant commercial decisions [3]. The BDS faces several problems in computation and investigating huge information to recover important data which can benefit in medicinal are, sports industries, business, etc. Implementing privacy in data is certainly a difficult process, in previous times, the size of information gets increased. Similarly, the huge quantity of information is created from heterogeneous sources and enters distinct methods, like structured, unstructured, and semi-structured which result in the crash of commodity hardware. In Cryptography, tokenization, deduplication of information is the certain manner which supports protecting private information from hackers. Though this method and novel cutting-edge tool have been designed to confirm the security of information, in the same way, attackers are arriving with novel methods to utilize the susceptibility existing in BDS involving its framework. Privacy in Big data strategy involves the safety of information created and security framework, availability of data, client authorization, and private message [4]. A major problem is solved by Intrusion Detection System (IDS).

The massive increase in size of Big Data over differences in structured data needs enormous computation strength and time which generate very hard to acquire the perceptions of client's information in a moment. Furthermore, storing for data stream are distinguished via the large difference in data rate can over its capabilities. However, advanced technology such as Hadoop Distributed File System (HDFS), Hadoop MapReduce [5, 6], and modern processors such as i7, mainframe, Macintosh, and supercomputer, it has the probable of processing Big Data in moments. Several information methods are besides difficult in framework however, it is related with a heterogeneous source which generates the process and investigation of information becomes more complex. Several scientists presented machine learning (ML) for IDS to decrease the false positive rate (FPR) and generate precisely. But, in order to handle Big Data, the ML conventional methods take time in learning and categorizing information [7]. Big Data methods and ML for IDS resolve several problems like speed and computation time and establish precise IDS. This model is to establish Spark Big Data methods which handle Big Data in IDS to decrease computational time and attain efficient classifier.

This paper presents a new quantum brain storm optimization (QBSO) based feature selection with a fuzzy deep neural network (FDNN), called QBSO-FDNN model for IDS in a big data environment. The proposed model enables to the detection of intrusions in the big data environment. The presented model initially performs preprocessing to enhance the quality of the big data. Also, to reduce the computational complexity, the QBSO algorithm is applied to elect a better set of features. The choice of optimal features by the QBSO algorithm helps to boost the detection performance. Besides, the FDNN model is applied as a classification model for identifying the occurrence of intrusions in the network. A widespread range of simulations was carried out to point out the supremacy of the QBSO-FDNN model on the benchmark dataset.

2. Related works

Several studies have proposed for IDS. With the appearance of Big Data, the classical methods becoming very difficult to handle Data. Thus, numerous studies proposed to utilize Big Data methods to generate maximum speed and precise IDS. Few scientists utilized ML Big Data methods for IDS to handle Big Data. [8] Utilizes clustering ML method. Many scientists utilized k-means techniques in the ML library over Spark to define either the network traffic is attacked by hackers or not. In the presented technique, the KDD Cup 1999 was utilized for testing and training. In the projected technique the scientist never utilized the FS method to choose the relevant feature. [9] developed a clustering technique for IDS-dependent Mini Batch K-means integrated by principal component analysis (PCA). The PCA investigation was utilized to decrease the dimension of the processing dataset and next mini-batch K-means++ technique was utilized for clustered data.

[10] utilizes the classifier ML method. The scientist presented an IDS method depending upon DT with Big Data in Fog platform. In these developed techniques, the scientist proposed a pre-processing technique for string in the provided dataset and next standardize the data to guarantee the value of input data so as to enhance the performance of identification. It is utilize the DT technique for IDS and linked these techniques with Naïve Bayesian (NB) and K-nearest neighbors (KNN) approach. [11] presented the efficiency of NB, support vector machine (SVM), Decision tree (DT), and Random forest (RF) classifier technique of IDS by Apache Spark. The entire efficiency comparability is calculated over UNSW-NB15 dataset with respect to the accuracy, training, and predictive time.

Similarly, [12] presented real-world IDS dependent SVM and utilize Apache Storm architecture. This presented method is trained and estimated on KDD 99 datasets. Additionally, feature selection (FS) method is utilized in several studies. The PCA FS method designed in few presented IDS such as [13] introduced Big Data architecture for IDS in smart grid utilizing several techniques such as DT, SVM, NB, neural network (NN), and RF. In these techniques, a correlation dependent technique is utilized for PCA and FS to reduce dimension. The presented method is intended to minimalize the prediction time of the attackers and raise the accuracy of the classifier process. [14] presented an architecture for quick and precise identification of intrusion by Spark. The projected architecture utilizes Linear Discriminant Analysis (LDA) and Canonical Correlation Analysis (CCA) models to reduce feature, and 7 classifier techniques.

[15] Presented a similar PCA integrated by equivalent SVM technique depending upon Spark platform (SP-PCA-SVM). The PCA utilizes data investigation and extraction features to reduce dimension depending upon Bagging. The presented technique utilizes KDD99 to train and calculate. [16] presented optimize technique for FS. Scientists have developed Hadoop dependent equivalent Binary Bat Algorithm (BBA) for IDS. In these techniques, the scientist utilizes similar BBA for effective FS and optimization rate of detection. The MapReduce of Hadoop utilized to enhance computation complexities and equivalent NB gives a cost-efficient classifier. This projected technique is evaluated and trained over KDD99 datasets.

3. The Proposed QBSO-FDNN model

The working principle of the presented QBSO-FDNN model is illustrated in Fig. 1. The figure portrayed that the networking big data is initially preprocessed in different ways to enhance the data quality. Followed by, QBSO algorithm is utilized to elect an optimal set of features from the preprocessed data. Consecutively, the FDNN model is employed to perform the classification process and allocate proper class labels of the big data.

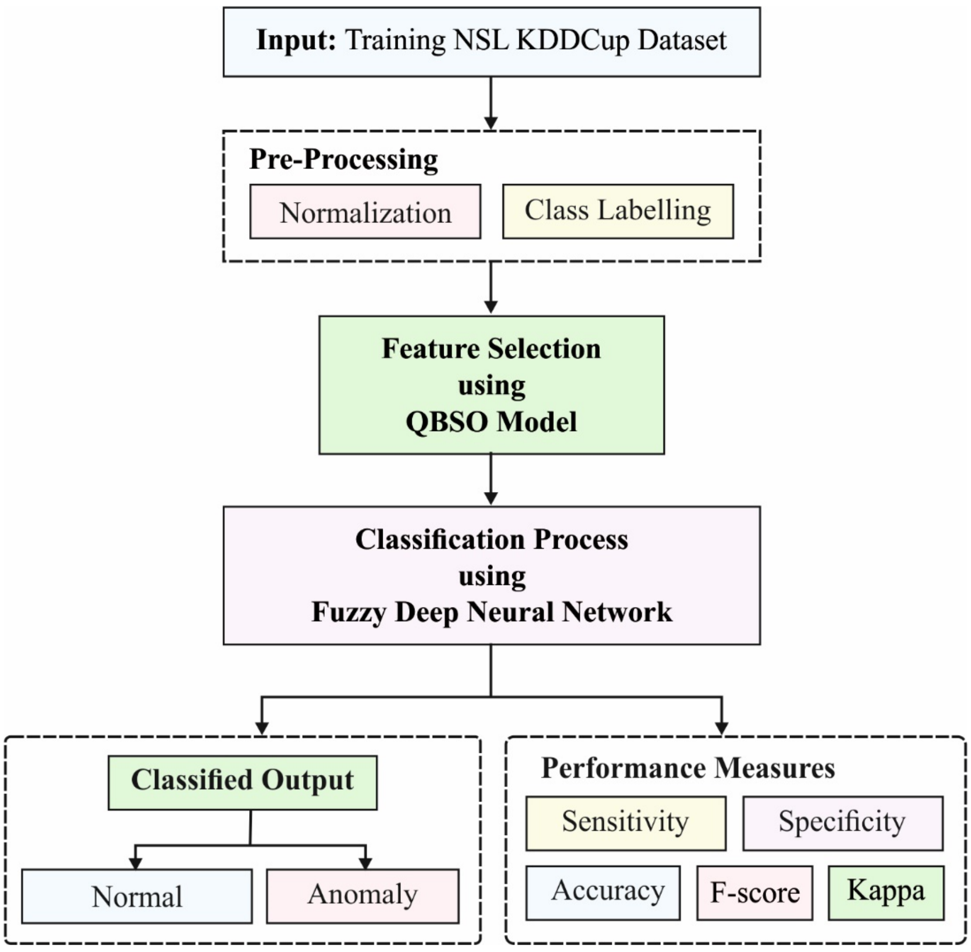

Fig. 1. Steps in Proposed Model.

3.1. *Initialization Process*

In the beginning, system's framework was authenticated, while the prolonged systems can support simultaneous processing of big datasets. Initially, NetBeans is introduced over JDK8.0, and mandatory library functions are used in NetBeans. Next, Hadoop is employed which inclines to apply Yarn, MapReduce, and HDFS. Subsequently, it undergoes initialization by Apache Spark which is in an identical cluster by Hadoop MapReduce. They are designed for initiating simultaneous processing. At last, the beginning phase gets accomplished by NSL KDD Datasets imported to NetBeans.

Hadoop MapReduce: Hadoop is called as "Big Data Handler" where the method is mostly utilized for monitoring the hugely created data. It is employed with Hadoop 2.6.0 for designing the method. This technique is used in several methods like HBase, HDFS, Pig, MapReduce, Hive, and so on. In Hadoop, YARN controls the relevant assets and task scheduled functions. In particular circumstances, the information is needed and stored in HDFS. Hadoop has the ability to store information in peta bytes and zeta bytes with no other restricted storage, and whereas it is employed with MapReduce it provides quick computation. MapReduce is determined as a batch centered program technique in Hadoop which carries out managing of data and schedule tasks. It separates the information as to independent chunks which is calculated completely from map function tasks parallel.

Apache Spark: Spark's so-called BDS placed over Hadoop and designed for Hadoop Yarn. Now, it can be applied to Apache Spark version 2.1.0. It is independent of MapReduce and HDFS. The units involved in Spark are Streaming, SQL, and MLlib (i.e., ML library). Various classifier methods in Apache Spark are yet employed. Hadoop clusters executed on streaming data, Apache Spark, and interactive queries.

Spark RDD: It assists a methodical technique which is similar to MapReduce; however, it extends through “Resilient Distributed Datasets” (RDD) namely information sharing concept. By the applications of abstraction, Spark is capable to acquire a broader processing cost which needs an exclusive engine, like ML, streaming, SQL, and graph theory.

3.2. *Data Preprocessing*

NSL KDD cup dataset is processed before it can be employed for IDS and upgraded via removal of repetitive measures in training data and with no other overlapping in testing data. Thus, it is computed to obtain a dataset with powerful computation, deficiency of repeated values, and non-existence of values in column. It can be pre-processed over by employing Parse labeled point. In MLlib, the labeled point is applied in supervised learning methods. Similarly, it is applied twice to save a label, therefore, the labeled point can be used for classification and regression. Spark.ml package provides ML API established the data frame that develops the major part of Spark SQL library.

3.3. *Quantum BSO based Feature Selection*

The BSO model is extremely utilized for issues which could not be resolved by one individual. In order to control these issues, individuals from several backgrounds are collected for the brainstorm. An important purpose of brainstorming is to create several concepts (solutions) as feasible, and an optimal solution is achieved for solving the particular issue. The BSO is a novel population-based ST technique simulated to the human brainstorming model. In BSO creates n arbitrary feasible results, and estimates to depends on the fitness function (FF). Based on BSO has of 3 steps that are clustering individuals, disrupting the cluster centers, and generating results. The BSO clusters n separates as to m clusters utilizing the k-means clustering method. In same results are clustered composed in all generations. The novel solution is created with probabilities of P, and it exchanges the cluster center (chosen arbitrarily), with the disrupting cluster center process [17]. At last, BSO creates the novel individual utilizing one cluster or by combined 2 clusters. In order to create a novel individual, the BSO arbitrarily chooses 1 or 2 cluster(s), through a probability of P-one. Afterward, BSO arbitrarily chooses one individual depends on 1 or 2 cluster(s) center(s) as:

$$X_{selected} = \{ \begin{array}{cc} X_i & one\ \ cluster \\ rand \times X_{1i} + (1 - rand) \times X_{2i}, & two\ clusters \end{array} \tag{1}$$

where X_{1i} and $X_{2\text{i}}$ are ith dimension of chosen clusters, and $rand$ represents the random values among 0 and 1. The BSO upgrades the chosen individual as:

$$X_{new} = X_{se1ecred} + \xi * random(0,1) \tag{2}$$

where $random$ signifies the Gaussian random value by 0 mean and unit variance, correspondingly; ξ refers the adjusting factor:

$$\xi = logsin\left(\frac{0.5 * m_i - c_\text{i}}{k}\right) \times\ rand \tag{3}$$

where m_i and c_i indicates the maximal count of iterations and current iteration, correspondingly; $logsin()$ implies the logarithmic sigmoid function, $rand()$ signifies the random values among 0 and 1, and k refers the rate of changing for the slope of $logsin()$ function.

To enhance the performance of the BSO technique, quantum computing concept is introduced to it. Fig. 2 showcased the flowchart of BSO technique [18].

Quantum computing is the novel kind of computing method that implements the models connected to quantum theory namely state superposition, quantum measurement, and quantum entanglement. The fundamental unit of quantum computing is qubit. In 2 fundamental states $|0>$ and $|1>$ procedure in qubit that is represented as linear combination of these 2 fundamental states as:

$$|Q >= \alpha|0> + \beta|1>. \tag{4}$$

$|\alpha|^2$ refers the probability of observed state $|0>$, $|\beta|^2$ represents the probability of observed state $|1>$, where $|\alpha|^2 + |\beta|^2 = 1$. In quantum is develop of n qubits. Because of the nature of quantum superposition, all the quantum’s has of 2^n feasible values. The n-qubits quantum is referred to as:

$$\Psi = \sum_{x=0}^{2^n-1} C_x\,|x>, \sum_{x=0}^{2^n-1} |\,C_x|^2 = 1. \tag{5}$$

The Quantum gates are modifying the state of qubits namely NOT gate, Hadamard gate, rotation gate, etc. the rotation gate is explained as mutation function for making quanta technique optimal results and at last to determine the global optimal solution.

The rotation gate is determined as:

$$\begin{bmatrix}\alpha^d(t+1)\\ \beta^d(t+1)\end{bmatrix} = \begin{bmatrix}\cos(\triangle\,\theta^d) & -sin(\triangle\,\theta^d)\\ sin(\triangle\,\theta^d) & cos(\triangle\,\theta^d)\end{bmatrix}\begin{bmatrix}\alpha^d(t)\\ \beta^d(t)\end{bmatrix} for\ d = 1,2,\dots,n. \quad (6)$$

$\triangle\,\theta^d = \triangle \times S\,(\alpha^d, \beta^d)$, $\triangle\,\theta^d$ implies the rotation angle of qubit, where $\triangle$ and $S(\alpha^d, \beta^d)$ are size and way of rotation correspondingly.

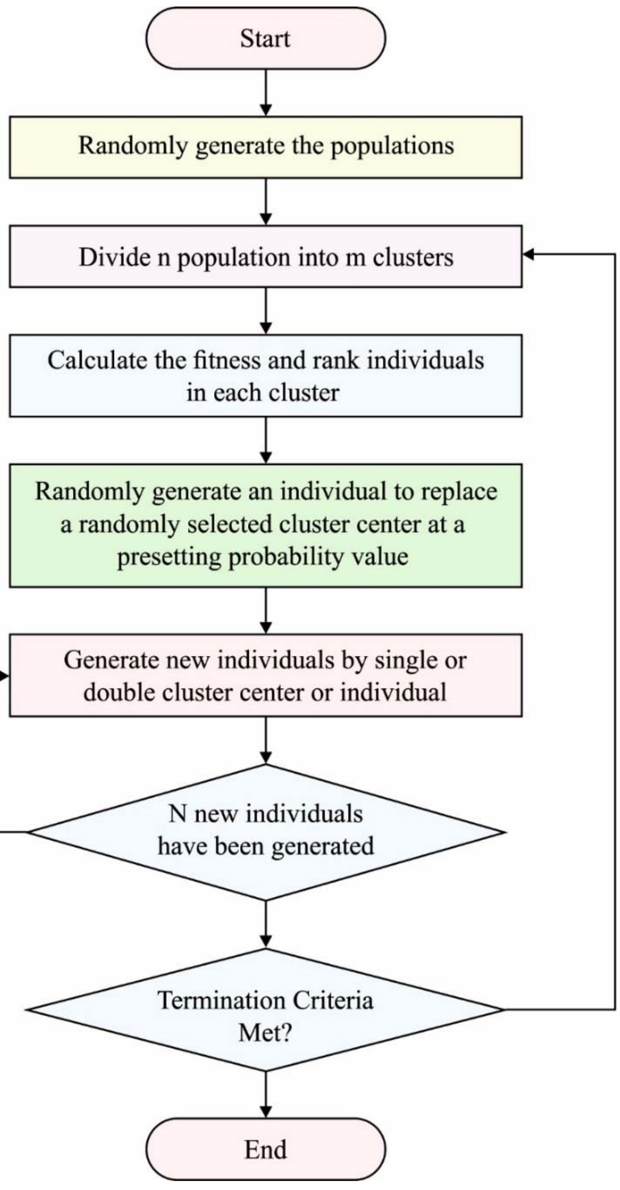
Start
Randomly generate the populations
Divide n population into m clusters
Calculate the fitness and rank individuals in each cluster
Randomly generate an individual to replace a randomly selected cluster center at a presetting probability value
Generate new individuals by single or double cluster center or individual
N new individuals have been generated
Termination Criteria Met?
End

Fig. 2. Flowchart of BSO.

In BSO, all the solutions, S, has $p \times d$ features as:

$$S = \{D_1^1, D_2^1, , D_d^1, D_1^2, D_2^2, , D_d^2, , D_1^p, , D_d^p\} \quad (7)$$

where p denotes the count of prototype nodes from f_2^a layer and d imply the dimension of all the prototype nodes. After that, D_d^p is regarded as:

$$D_d^P = \begin{cases} don't\ care\ feature, if\ D_d^P < \theta \\ other\ feature, if\ D_d^P \geq \theta \end{cases} \quad (8)$$

where $0 < \theta < 1$.

In BSO FF is expressed for reducing the classifier error as:

$$Fitness = 1 - \frac{NCP}{TNS} \quad (9)$$

where NCP represents the number of correctly classified samples and TNS signifies the total number of samples. The purpose is for minimizing the rate of error.

3.4. *FDNN based Classification*

FDNN model is employed for the simultaneous extraction of the fuzzy and neural representation data. The knowledge learned from the 2 aspects is integrated at the fusion layer to attain final representation for classifying data. The fuzzy based depiction minimizes the uncertainty and neural depiction discards the noise that exists in the input data. The FDNN model makes use of two effective representations producing the fused depiction for end classification [19]. The FDNN model comprises l as layer number, $a_i^{(l)}$ indicates the input of the i^{th} node and $o_i^{(l)}$ is the equivalent outcome. At the same time, the different parts involved in FDNN are discussed below. Fig. 3 demonstrates the structure of DNN model [20].

Fuzzy logic (FL) representation

The nodes in the input layer are linked to many membership functions which allocates linguistic labels to every input parameter. The fuzzy membership function determines the degree to which the input node comes under a particular fuzzy set. At this layer, the i^{th} fuzzy neuron $u_i(\cdot): R \rightarrow [0,1]$ undergo mapping of the k^{th} input as the fuzzy degree

$$o_i^{(l)} = u_i\left(a^{(l)}\right) = e^{-\left(a_k^{(l)} - \mu_i\right)^2 / \sigma_i^2}, \forall i \tag{10}$$

The Gaussian membership function with mean μ and variance σ^2 is applied. The fuzzy rule layer carries out the 'AND' FL function, i.e., $o_i^{(l)} = \Pi_j o_j^{(l-1)}$, $\forall j \in \Omega_i$, where Ω_i represents the collection of the nodes on the $(l-1)^{th}$ layer which is linked to i. Here, the $(l-1)^{th}$ layer indicates the input layer. The output denotes the fuzzy degree.

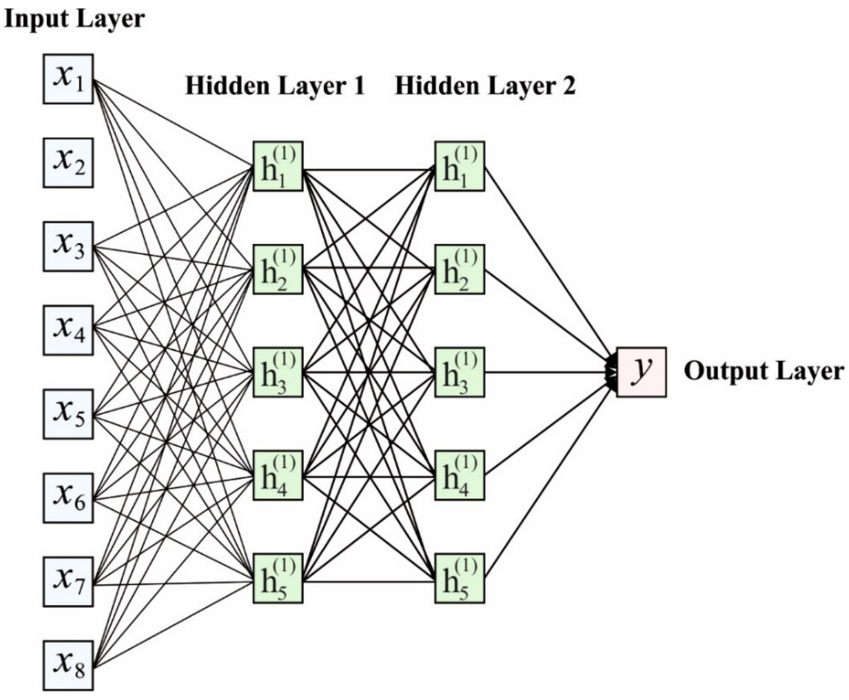

Fig. 3. Structure of DNN.

Neural representation

It makes use of neural learning concept for transforming the input as some high-level representation. The layer is fully connected denoting that every node on the $(l)^{th}$ layer is linked to every other node on the $(l-1)^{th}$ layer with variables $\theta^{(l)} = \{\mathrm{w}^{(l)}, \mathrm{b}^{(l)}\}$, i.e.

$$o_i^{(l)} = \frac{1}{1 + e^{-a_i^{(l)}}}, a_i^{(l)} = \mathrm{w}_i^{(l)} \mathrm{o}^{(l-1)} + b_i^{(l)}, \tag{11}$$

where $\mathrm{w}_i^{(l)}$ and $b_i^{(l)}$ denotes the weights and bias linking to node i on the lth layer.

Fusion part

The fusion concept is based on the principle of multi-modal learning [21], which imposes that the feature extraction from an individual view is insufficient for capturing the complicated structure of high content data. It creates a set of features from several views and synthesizes it to a high-level representation to perform classification. In FDNN, fuzzy and neural portions are utilized for seeking effective representation by the reduction of uncertainties and noises of the input data. In addition, the neural and fuzzy learning portions are configured in the neural network. So, it is instinctive to design the feature fusion step. Here, the commonly used multi-model neural network is employed for the combination of neural and fuzzy representations with densely connected fusion layers.

$$o_i^{(l)} = \frac{1}{1+e^{-a_i^{(l)}}} \tag{12}$$

$$a_i^{(l)} = (\mathrm{w}_d)_i^{(l)}(\mathrm{o}_d)^{(l-1)} + (\mathrm{w}_f)_i^{(l)}(\mathrm{o}_f)^{(l-1)} + b_i^{(l)},$$

In (11), the output from the deeper representation term (o_d) and FL representation term (o_f) undergo fusion with weights w_d and w_f, correspondingly. Afterward, the fused data undergo deep transformation by keeping many FC layers next to the fusion layer. The output combining the fuzzy degree and neural representation together is no longer fuzzy degree. The fuzzy based learning scheme is utilized owing to the three merits. Firstly, it offers a comfortable method of reducing the uncertainty that exists in the input data.

This essential ambiguity reduction pursuit is the necessary feature of fuzzy system and it is essential. Secondly, the fuzzy learning results in soft logistic value (fuzzy representation) in the interval of $(0,1)$. In addition, every individual dimension in the neural output also lie in the interval of $(0,1)$ (see Eq. (11)). The quantity of fusion and neural output exists in identical ranges so that it is easier to fuse the two outputs. Thirdly, the fuzzy learning term enables task-driven parameter learning. At this point, the draining handcrafted parameter selection process undergoes replacement with the smart data-driven learning by backpropagation.

Task-driven part

The last section is the classifier layer which allocates the fused representation to its respective class label. Here, softmax function is employed for the classification of data points into proper class labels. It is denoted that the (f_i, y_i) as the i^{th} input and its equivalent label, $\pi_\Theta(f_i)$ denotes the feed-forward conversion of the FDNN from the input layer to the final task driven layer. Next, the soft-max function is utilized as the output layer with the c^{th} entry as determined as,

$$\hat{y}_{ic} = p(y_i|\mathrm{f}_i) = \frac{e^{w_c}\pi_\Theta(f_i) + b_c}{\sum_c e^{w_c}\pi_\Theta(f_i) + b_c}, \tag{13}$$

where w_c and b_c signify the regression coefficient and bias of the c^{th} classes and $\hat{y}_i = [\hat{y}_{i1}, \dots, \hat{y}_{ik}]$. Afterward, the mean square error is represented over m training instances,

$$C = \frac{1}{m}\sum_{i}^{m} \|\hat{y}_i - y_i\|_2^2 \tag{14}$$

4. Performance Validation

The performance of the proposed QBSO-FDNN model has been validated against NSL-KDDCup dataset. The details related to the dataset are given in Table 1. The dataset contains samples under normal and abnormal classes including several subclasses under anomaly type.

Table 1. Types of Attacks in NSL-KDD Dataset.

Attack Type	Description	No. of Samples
Anomaly Instances		
Dos	Denial of service attack	45,927
R2l	Unauthorized access from a remote host	995
Probe	Port monitoring or scanning	11,656
U2r	Unauthorized local super user privileged access	52
Normal Instances		
Normal	Not a Attack	67,343

Table 2. Results of Existing with Proposed QBSO-FS Method on Applied IDS Dataset.

Methods	Best Cost	Selected Features
QBSO-FS	0.00076323	2,3,5,6,7,8,9,11,14,16,18,32,36,39
BSO-FS	0.00079351	2,4,5,6,7,9,11,13,15,16,17,19,20,21,23,38,38,40
WOA-FS	0.00093985	3,5,8,13,18,20,21,22,23,25,26,28,30,32,33,34,36,38,40
GA-FS	0.00115060	21,7,27,32,25,34,1,2,35,3,24,40,28,26,10,5,33,14,16,12,36,23,30, 38,22,15,37,9

Table 2 and Fig. 4 illustrate the results obtained by the QBSO-FS model during the selection of features.

A comparative result takes place with the BSO-FS, WOA-FS, and GA-FS models. The table values denoted that the GA-FS model has reached insufficient results with the least best cost of 0.00115060. Simultaneously, the BSO-FS and WOA-FS models have accomplished slightly increased best cost of 0.00079351 and 0.00093985 respectively. At last, the QBSO-FS model has exhibited superior results with the best cost of 0.00076323.

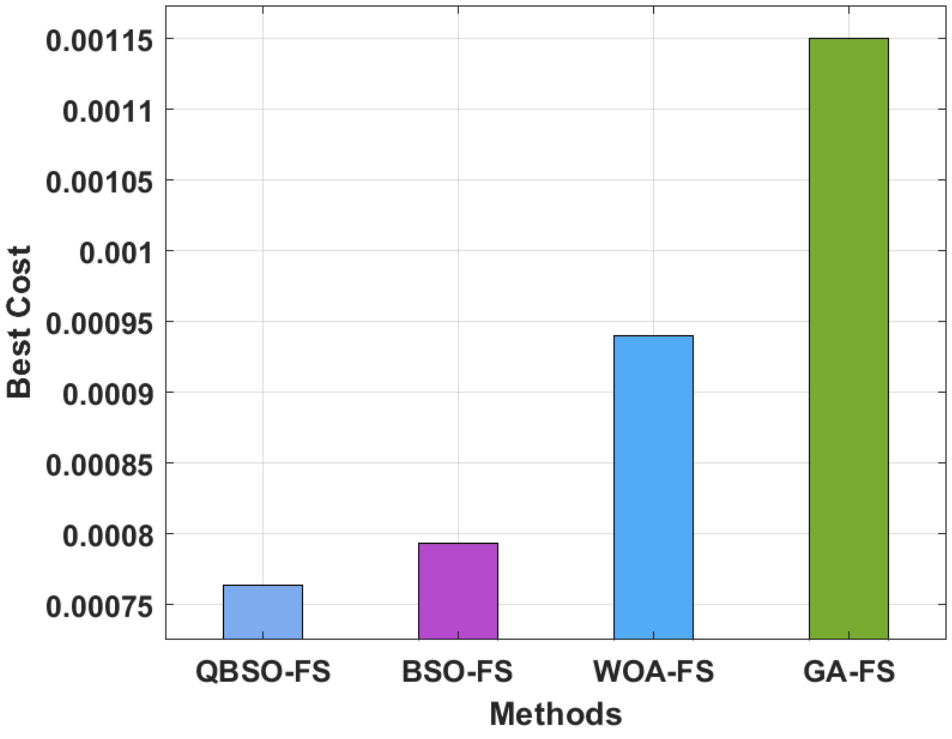

Fig. 4. Best cost analysis of QBSO-FS model.

Table 3 and Fig. 5 depict the IDS outcomes analysis of the QBSO-FDNN model on the detection of distinct types of intrusions. The results depicted that the QBSO-FDNN model has showcased effective intrusion detection performance on the applied dataset. For instance, the QBSO-FDNN model has classified the DoS attack with the sensitivity, specificity, accuracy, F-Measure, and Kappa of 98.90%, 99.18%, 98.42%, 97.36%, and 97.18% respectively. Along with that, the QBSO-FDNN model has classified the R21 attack with the sensitivity, specificity, accuracy, F-Measure, and Kappa of 98.96%, 99.64%, 98.98%, 99.75%, and 98.46% correspondingly. At the same time, the QBSO-FDNN approach has classified the Probe attack with the sensitivity, specificity, accuracy, F-Measure, and Kappa of 98.98%, 99.29%, 98.90%, 98.63%, and 98.23% respectively. Simultaneously, the QBSO-FDNN method has classified the U2r attack with the sensitivity, specificity, accuracy, F-Measure, and Kappa of 98.36%, 99.36%, 98.93%, 98.45%, and 98.8% correspondingly. Concurrently, the QBSO-FDNN technique has classified the normal attack with the sensitivity, specificity, accuracy, F-Measure, and Kappa of 99.27%, 99.63%, 99.26%, 99.32%, and 99.12% respectively. Besides, the QBSO-FDNN methodology has classified with average values of sensitivity, specificity, accuracy, F-Measure, and Kappa of 98.89%, 99.42%, 98.90%, 98.70%, and 98.36% correspondingly.

Table 3. Result Analysis of Proposed QBSO-FDNN Model.

Attacks	Sensitivity	Specificity	Accuracy	F-Measure	Kappa
Dos	98.90	99.18	98.42	97.36	97.18
R2l	98.96	99.64	98.98	99.75	98.46
Probe	98.98	99.29	98.90	98.63	98.23
U2r	98.36	99.36	98.93	98.45	98.80
Normal	99.27	99.63	99.26	99.32	99.12
Average	**98.89**	**99.42**	**98.90**	**98.70**	**98.36**

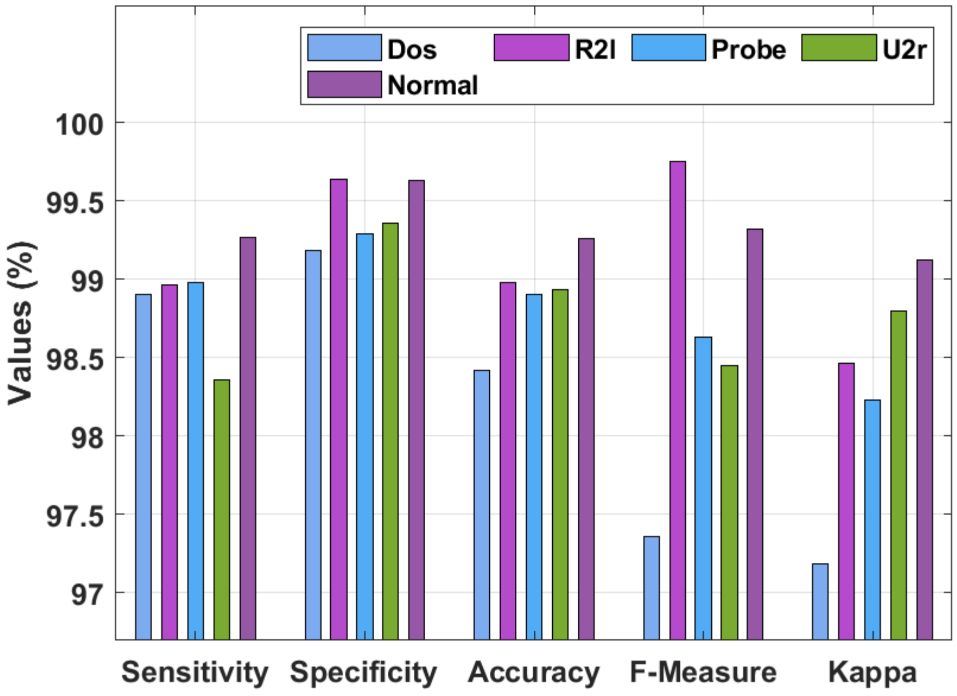

Fig. 5. Result analysis of QBSO-FDNN Model with distinct measures.

Table 4 and Fig. 6 provide a detailed comparative results analysis of the QBSO-FDNN model with other existing methods. On analyzing the detection results interms of accuracy, the RBFNetwork method has resulted in worse detection performance with the accuracy of 92.93% whereas a slightly increased accuracy of 93.04% has been obtained by the Rand Forest model. Next to that, the DT(J48) model has depicted certainly increased accuracy of 95.53%. Simultaneously, the Rand. Tree model has resulted in a manageable accuracy of 95.55%. Though the LR model has exhibited near optimal accuracy of 97.10%, the presented QBSO-FDNN model has reported a maximum accuracy of 98.90%.

Table 4. Performance Analysis of Traditional Classifiers with Proposed Model for Applied Dataset.

Methods	Sensitivity	Specificity	Accuracy	F-score	Kappa
Proposed QBSO-FDNN	98.89	99.42	98.90	98.70	98.36
RBFNetwork	93.40	92.38	92.93	93.38	85.79
LR	97.26	96.92	97.10	97.29	94.19
Rand. Forest	92.39	93.83	93.04	93.58	85.99
Rand. Tree	95.68	95.39	95.55	95.84	91.06
DT (J48)	95.68	95.37	95.53	95.83	91.03

On examining the detection outcomes with respect to sensitivity and specificity, the Rand. Forest method has resulted in a least detection performance with the sensitivity and specificity of 92.39% and 93.83% whereas a somewhat improved sensitivity and specificity of 93.4% and 92.38% has been attained by the RBFNetwork model. Along with that, the DT(J48) technique has showcased certainly higher sensitivity and specificity of 95.68% and 95.37%. At the same time, the Rand. Tree approach has resulted in a manageable sensitivity and specificity of 95.68% and 95.39%. Besides, the LR approach has showcased near better sensitivity and specificity of 97.26% and 96.92%, the proposed QBSO-FDNN methodology has reported a higher sensitivity and specificity of 98.89% and 99.42%.

On investigating the detection results interms of F-score and kappa, the RBFNetwork technique has resulted in lower detection performance with the F-score and kappa of 93.38% and 85.79% whereas a slightly higher F-score and kappa of 93.58% and 85.99% has been reached by the Rand. Forest approach. Next to that, the DT(J48) method has exhibited certainly a superior F-score and kappa of 95.83% and 91.03%. Concurrently, the Rand. Tree manner has resulted in a manageable F-score and kappa of 95.84% and 91.06%. Also, the LR method has demonstrated near optimal F-score and kappa of 97.29% and 94.19%, the projected QBSO-FDNN technique has reported a superior F-score and kappa of 98.7% and 98.36%.

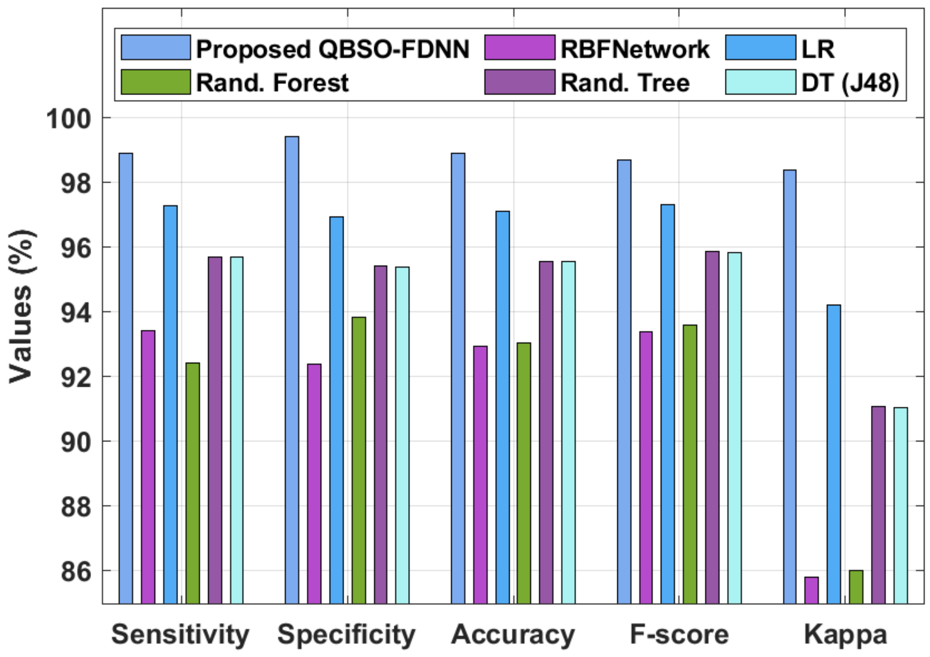

Fig. 6. Comparative analysis of QBSO-FDNN model with existing methods.

Table 5 and Fig. 7 illustrate the comparative results analysis of the QBSO-FDNN model with recently developed models interms of accuracy [22, 23]. From the attained results, it is evident that the CS-PSO and GBT models have demonstrated insignificant detection results with the accuracy of 75.51% and 84.25% respectively. At the same time, the Gaussian Process and DNN-SVM models have portrayed slightly improved and closer accuracy of 91.06% and 92.03% respectively. Similarly, the Fuzzy c-means, GA-Fuzzy, and Cuckoo optimization methodologies have demonstrated certainly enhanced outcomes with the accuracy of 95.3%, 96.53%, and 96.88% respectively. Followed by, the IntruDTree and Behaviour based IDS models have reported the competitive accuracy of 98% and 98.80% respectively. But the presented QBSO-FDNN model has resulted in a maximum accuracy of 98.9%.

Table 5. Performance Analysis of Recent Methods with Presented QBSO-FDNN Model on Applied Dataset.

Methods	**Accuracy (%)**
Proposed QBSO-FDNN	98.90
IntruDTree (2020)	98.00
CS-PSO (2019)	75.51
Behaviour Based IDS (2019)	98.80
Cuckoo Optimization (2018)	96.88
DNN+SVM (2018)	92.03
GA+ Fuzzy (2018)	96.53
Fuzzy C-Means (2018)	95.30
Gradient Boosting (2018)	84.25
Gaussian Process (2015)	91.06

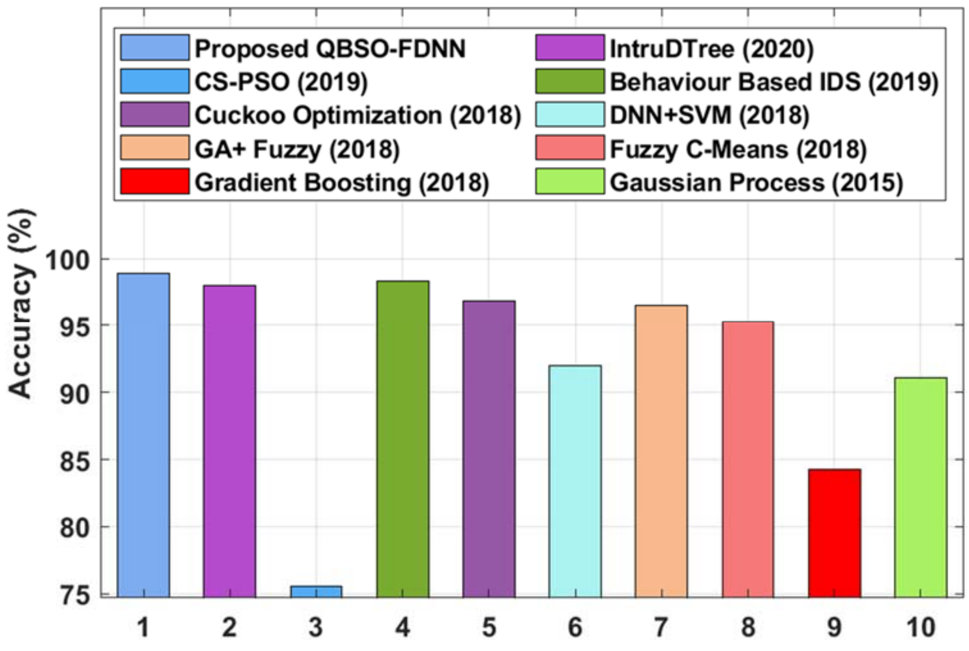

Fig. 7. Accuracy analysis of QBSO-FDNN model with recent techniques

5. Conclusion

This paper has presented an optimal QBSO-FDNN model for IDS in big data environment. The proposed model enables to detection of intrusions in the big data environment. Firstly, the networking big data is initially preprocessed in different ways to enhance the data quality. In addition, to decrease the computational complexity, QBSO technique is applied to elect a better set of features. The choice of optimal features by the QBSO algorithm helps to boost the detection performance. Consecutively, the FDNN model is employed to perform classification process and allocate proper class labels of the big data. A widespread range of simulations was performed to point out the supremacy of the QBSO-FDNN model on benchmark dataset. The resultant experimental values showcased the superior performance of the QBSO-FNN model with the maximum detection accuracy of 98.90%.

References

[1] Zuech R, Khoshgoftaar TM, Wald R. Intrusion detection and big heterogeneous data: a survey. J. Big Data 2015;2(1):3.
[2] Dietrich D, Heller B, Yang B. Data science & big data analytics: discovering. Analyzing, visualizing and presenting data; 2015.
[3] Chen M, Mao S, Liu Y. Big data: A survey. Mobile Netw. Appl. 2014;19(2):171–209.
[4] Moreno J, Serrano MA, Fernández-Medina E. Main issues in big data security. Future Internet 2016;8(3):44.
[5] Wang L, Jones R. Big data analytics for network intrusion detection: a survey. Int J Netw Commun 2017;7(1):24–31.
[6] Download link (Hadoop): https://archive.apache.org/dist/hadoop/core/. Accessed on December 2017.

[7] Donkal, G. and Verma, G.K., 2018. A multimodal fusion based framework to reinforce IDS for securing Big Data environment using Spark. Journal of information security and applications, 43, pp.1-11.
[8] Ferhat K, Sevcan A. Big Data: controlling fraud by using machine learning libraries on Spark. Int J Appl Math Electron Comput. 2018;6(1):1–5.
[9] Peng K, Leung VC, Huang Q. Clustering approach based on mini batch Kmeans for intrusion detection system over Big Data. IEEE Access. 2018.
[10] Peng K. et al. Intrusion detection system based on decision tree over Big Data in fog environment. Wireless Com mun Mob Comput. 2018. https://doi.org/10.1155/2018/4680867.
[11] Belouch M, El Hadaj S, Idhammad M. Performance evaluation of intrusion detection based on machine learning using Apache Spark. Procedia Comput Sci. 2018;127:1–6.
[12] Manzoor MA, Morgan Y. Real-time support vector machine based network intrusion detection system using Apache Storm. In: IEEE 7th annual information technology, electronics and mobile communication conference (IEMCON), 2016. Piscataway: IEEE. 2016; p. 1–5.
[13] Vimalkumar K, Radhika N. A big data framework for intrusion detection in smart grids using Apache Spark. In: Inter national conference on advances in computing, communications and informatics (ICACCI), 2017. Piscataway: IEEE; 2017. p. 198–204.
[14] Dahiya P, Srivastava DK. Network intrusion detection in big dataset using Spark. Procedia Comput Sci. 2018;132:253–62.
[15] Wang H, Xiao Y, Long Y. Research of intrusion detection algorithm based on parallel SVM on Spark. In: 7th IEEE Inter national conference on electronics information and emergency communication (ICEIEC), 2017 . Piscataway: IEEE; 2017. p. 153–156.
[16] Natesan P, et al. Hadoop based parallel binary bat algorithm for network intrusion detection. Int J Parallel Program. 2017;45(5):1194–213.
[17] Pourpanah, F., Shi, Y., Lim, C.P., Hao, Q. and Tan, C.J., 2019. Feature selection based on brain storm optimization for data classification. Applied Soft Computing, 80, pp.761-775.
[18] Yu Y., Yang L., Wang Y., Gao S. (2019) Brain Storm Algorithm Combined with Covariance Matrix Adaptation Evolution Strategy for Optimization. In: Cheng S., Shi Y. (eds) Brain Storm Optimization Algorithms. Adaptation, Learning, and Optimization, vol 23. Springer, Cham. https://doi.org/10.1007/978-3-030-15070-9_6
[19] Deng, Y., Ren, Z., Kong, Y., Bao, F. and Dai, Q., 2016. A hierarchical fused fuzzy deep neural network for data classification. IEEE Transactions on Fuzzy Systems, 25(4), pp.1006-1012.
[20] Ramesh, S. and Vydeki, D., 2020. Recognition and classification of paddy leaf diseases using Optimized Deep Neural network with Jaya algorithm. Information processing in agriculture, 7(2), pp.249-260.
[21] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, “Multimodal deep learning,” in Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 689–696.
[22] Sarker, I.H., Abushark, Y.B., Alsolami, F. and Khan, A.I., 2020. Intrudtree: a machine learning based cyber security intrusion detection model. Symmetry, 12(5), p.754.
[23] Maheswari, M. and Karthika, R.A., 2021. A Novel QoS Based Secure Unequal Clustering Protocol with Intrusion Detection System in Wireless Sensor Networks. Wireless Personal Communications, pp.1-23.