Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)

Build Network Intrusion Detection System based on combination of Fractal Density Peak Clustering and Artificial Neural Network

Salam Saad Alkafagi [a]

*[1]Babylon Education Directorate, Ministry of Education, Babil, Iraq, Email: salam.s.alkafagi@gmail.com

ARTICLE INFO

Article history:
Received: 30 /11/2022
Rrevised form: 01 /01/2023
Accepted : 04 /01/2023
Available online: 15 /02/2023

Keywords:

Data Mining,

Anomaly Intrusion Detection System,

Density Peak Cluster algorithm,

Neural Network algorithm.

ABSTRACT

Imbalanced data poses a serious problem in intrusion detection systems. In this article, we propose a network intrusion detection system based on fractal density peak clustering and an artificial neural network (FD-ANN). The proposed detection system consists of three parts: data clustering based on the density-peak clustering (DPC) method, using the fractal concept as a membership weight of all data to the cluster, and a neural network to classify the data. The DPC method uses categorization of the tare data into subgroups with strongly correlated attributes to reduce the size of the training data and the imbalance of the sample. Each subgroup has its neural network to train the data. Based on fractal membership weights, the output of all classifiers of the sub-neural networks is combined using the aggregation function. The benchmarks of this model are based on the data sets NSL-KDD and UNSW-NB15. The proposed solution outperforms other known classification approaches in terms of overall accuracy, recall, precision, and F1 score.

https://doi.org/10.29304/jqcm.2023.15.1.1151

1. Introduction

The popularity of the Internet led to the need to share and access information, and this popularity led to an increase in the amount of information and resource consumption. As data is stored in the cloud or various databases, it is crucial to protect it and ensure its security[1]. Personal information is an essential component that needs to be protected from attacks. Nowadays, many organizations use various strategies to authenticate and authorize access to data that should be kept secure and confidential[2]. Network attacks are becoming more numerous and diverse: ransomware is increasing like never before, and zero-day exploits are becoming so critical that they are receiving media attention[3]–[5].

Antivirus programs and firewalls are no longer sufficient to protect an enterprise network, which must be covered with multiple layers of security. On the other hand, the massive increase in computer network usage and the development of applications running on different platforms has drawn attention to network security[6]. An intrusion detection system (IDS), which protects the network environment, is an important aspect of network security. When an IDS is used to detect threats over a network, it is called a NIDS[1], [7]. In general, NIDS is classified into two types: The first type is a signature-based or misuse-based intrusion detection system (SIDS),

∗Corresponding author **Salam Saad Alkafagia**

Email addresses: salam.s.alkafagi@gmail.com

Communicated by 'sub etitor'

which detects attacks in network traffic based on recognized patterns. These patterns reveal a series of activities that have been collected and stored in a database. The pattern database is responsible for identifying network behavior by detecting only attacks whose patterns have already been stored in this database[1].

Anomaly-based intrusion detection systems (AIDS) are the second type[8]. This system is based on network behavior as the primary parameter for analyzing network transactions. The learned system accepts network transactions with predefined behavior; otherwise, it issues a warning. The ultimate goal of a network intrusion detection scheme for a flooding attack is to have an efficient technique to track and mitigate such an attack on the victim side[9]. In our work, many crucial questions are raised to be answered by analyzing, modeling, and implementing a hybrid scheme to achieve an excellent result for multi-classification and detect the low rate attack using data mining clusters and neural network analysis.

The main motivation of this study is to develop an attack mechanism for information transmission over networks and to reduce the impact of zero-day attacks. Technically, the main component of IDS is a machine learning based on the amount of data it has been trained on, i.e., it depends on the amount of knowledge extracted from the training data. Machine learning is not able to predict a small number of categories, so it is necessary to weigh the data before making a decision. The problems based on the NIDS we can summarize in the following:

A firewall cannot detect most attacks because some attacks involve a large number of ICMP messages that block the bandwidth of the victim network or route unauthorized traffic through ports, such as ICMP flooding and TCP flooding attacks. This leads to early damage to an organization's resources, making early detection by the firewall difficult[10].

Selecting a high-quality network dataset to train the NIDS system is one of the most difficult challenges in analyzing the credibility of NIDS theories[11].

The presence of missing values in the data under study is a common problem in the intrusion detection dataset, which makes it impossible to analyze the data correctly. Therefore, it is necessary to identify and correct missing values in data mining before performing any analysis[12], [13].

Multi-classification and low rate attack detection rate is a big challenge because the actual network environment is unbalanced, which means that the threat records in the network traffic are lower than the regular records; at the same time, the classifier is biased against more frequent data, which lowers the detection rate of records of low rate attack[14].

The proposed model (FD-ANN) uses robust data mining cluster analysis (DPC) algorithm to solve the multiple class imbalance problem and increase the detection rate of low-rate attacks. A complex dataset from a real network environment is analyzed by improving the (DPC) algorithm in terms of its distance metric. The time and computational complexity are reduced by proposing a new membership matrix FFM based on the fractal factor. Definition of the performance evaluation of the proposed scheme in terms of the different types of classification attacks and the detection rate of low-rate attacks. The obtained results are validated with other works to show the effectiveness of the proposed scheme.

2. Related Works

In this section, we discuss several papers by researchers who have worked on intrusion detection techniques. Many researchers have used data mining clustering techniques such as point assignment and neural or deep network classifiers as hybrid techniques to build IDS. The main reason for the decrease in true-positive rate in several recent models is due to two main reasons, either the non-use of data balancing techniques in the training phase or the use of the inefficient built-in membership function method.

Yanqing et al [15], combined the modified density peak clustering method with Deep Belief Networks to develop a hybrid intrusion detection technique. MDPCA is a technique for detecting similarities in complex and large network data. To reduce the size of the training set and eliminate the imbalance of the training samples, MDPCA is used to divide the set into numerous subgroups with similar attribute sets. For training, each subgroup is given its sub-DBNs classifier. The membership weights are used to aggregate the output of all sub-DBNs classifiers. The MDPCA-

DBNs performance is evaluated using the NSL_KDDTest+, NSL -KDDTest-21, and UNSW_NB15 datasets. Using kernel methods with deep learning, the model produces a fairly good accuracy, but with high complexity

Chaofei et al. [16], have proposed an intrusion detection system based on the Stacked Auto-Encoder (SAE), an attention mechanism, and a deep neural network DNN (SAE-DNA). The attention mechanism helps the network to have a strong intrusion detection framework because the SAE represents data with a latent layer. The SAE encoder can automatically extract features, but also improve the detection accuracy of the DNN by initializing the weights of the DNN potential layers. The NSL KDD dataset was used to evaluate the performance of SAE-DNN in binary and multi-classification. In multi-classification, the model SAE-DNN outperforms machine learning approaches such as random forest and decision tree by identifying regularly and attacking symmetrically. The limitation of the proposed SAE-DNN is that the method does not use the construction mechanism between attack classes, which weakens the algorithm's ability to deal with zero-day attacks.

Sydney et al [17], analyzed the UNSW-NB15 intrusion detection dataset used to train and test the models. They use the K-mean clustering with the XGBoost algorithm and apply a filter-based approach to feature reduction. They examined both binary and multiclass steps in the experiment. The authors found that using the XGBoost-based feature selection strategy allows the classifiers to improve their test accuracy. Using static K-mean methods is not suitable for work with intrusion detection environments.

in [18] , the authors develop two models for network intrusion detection based on Deep Learning and two approaches to data preprocessing, a simplified preprocessing approach and a hybrid two-stage preprocessing strategy, to produce meaningful features. These models use the CNN paradigm. They create binary and multiclass classification models. The proposed method combines dimensionality reduction with feature engineering through deep feature synthesis. Two benchmark datasets, the NSL_KDD, and the UNSW_NB15 dataset are used to evaluate the performance of the models. The limitation of using CNN with intrusion detection, it fails to encode object orientation and position. data in various positions are challenging for them to categorize.

3. Density Peak Clustering (DPC) Algorithm

The Density Peak Clustering (DPC) algorithm is one of the Density-based methods. This algorithm discovers clusters based on a high local density of data points[19]. The DPC deal with continuous regions. The original Density Peak Clustering (DPC) algorithm starts by defining the distance between two data objects to calculate the density and separation distances from other data objects with higher density[20]. Let X = {x1, x2 . . . xn} be a dataset with n data points. Each xi has M attributes. Therefore, xij is the jth attribute of data point xi.instead of Euclidean distance used to calculate the distance between the data point's xi and xj can be expressed as follows[21]:

$$dis\ (xi, xj) = \sqrt{\sum_{i=1}^{n}(xi - xj)^2} \qquad (1)$$

Equations (2) express the calculation of local density for each point in the given dataset.

$$Pi = \sum_{i,j \in S} exp\left(-\frac{di,j^2}{Cd^2}\right) \qquad (2)$$

Where di, j is the distance between points i and j and Cd is the initial specified parameter as the cutoff distance to determine the neighborhood radius. The second step is to calculate separation distance δi, which works to find the minimum distance between the point xi and any other point of higher density. Equation (3) shows the calculation of the separation distance δi [21]:

$$\delta i = \begin{cases} min(\text{ɗ}ij), & if\ i > j \\ \quad or & \\ max(\text{ɗ}ij), & \end{cases} \qquad (3)$$

This method identifies the cluster centers by anomalously high $\boldsymbol{Pi}$ and $\boldsymbol{\delta i}$ after the local density and separation distance for each point have been computed. The Peak points (center points) are selected using a two-dimensional decision graph with a local density as the horizontal axis and separation distance as the vertical axis[19]. After determining the center points, the remaining points in the dataset will be assigned to the nearest center.

4. Artificial Neural Networks (ANN)

The models inspired by human brain processes are called Artificial Neural Networks (ANNs) [22]. They are members of the parametric classifier family, with parameters such as several neurons/layers, weights, and biases, and a collection of hyper-parameters such as several epochs, learning rate, and batch size. ANN is typically made up

of an input layer, some hidden layers, and an output layer. Lastly, learning models must be tested once the training is completed to determine their effectiveness. A Multi-Layer Perceptron is a simple feed-forward neural network that differs from a linear perceptron in terms of activation function and layer count. It has three layers: an input layer, a group of hidden layers, as well as an output layer[23], [24]. Its neurons use nonlinear activation functions and are trained using backpropagation. The Multi-Layer Perceptron algorithm is technically based on two operations: First, the Forward Propagation method is used to make a prediction, which is calculated using Equations (4) and (5) as shown below [25]:

$$Net_i = \sum_{j=1}^{n} w_{ij}\, x_i + \mu_i \quad , \mathrm{j} = 1, 2, 3, \ldots \mathrm{n} \qquad (4)$$

$$Y_i' = \varphi\,(Net_i) \qquad (5)$$

Where Net_i is show the result of net inputs xi multiplied by interconnection weight w_{ij}, φ is the activation function, n number of outputs, and Y_i' Means the actual output of the net.

The next operation is to train data based on the Back Propagation (BP) method, which is responsible to updates the weights of MLP. After each training epoch, the BP method will update the weights based on product error. The error is calculated as Equation (6):

$$E = \frac{1}{n} \sum_{i=1}^{n} (y_{Actual\ i} - y_{desired\ i})^2 \qquad (6)$$

5. Proposed Fractal Density Peak Clustering and Artificial Neural Network (FD-ANN)

The proposed FD-ANN system consists of three phases: prepeoceing, clustering, predaiction by using artificial neural network. Figure 1 shows the main steps of FD-ANN.

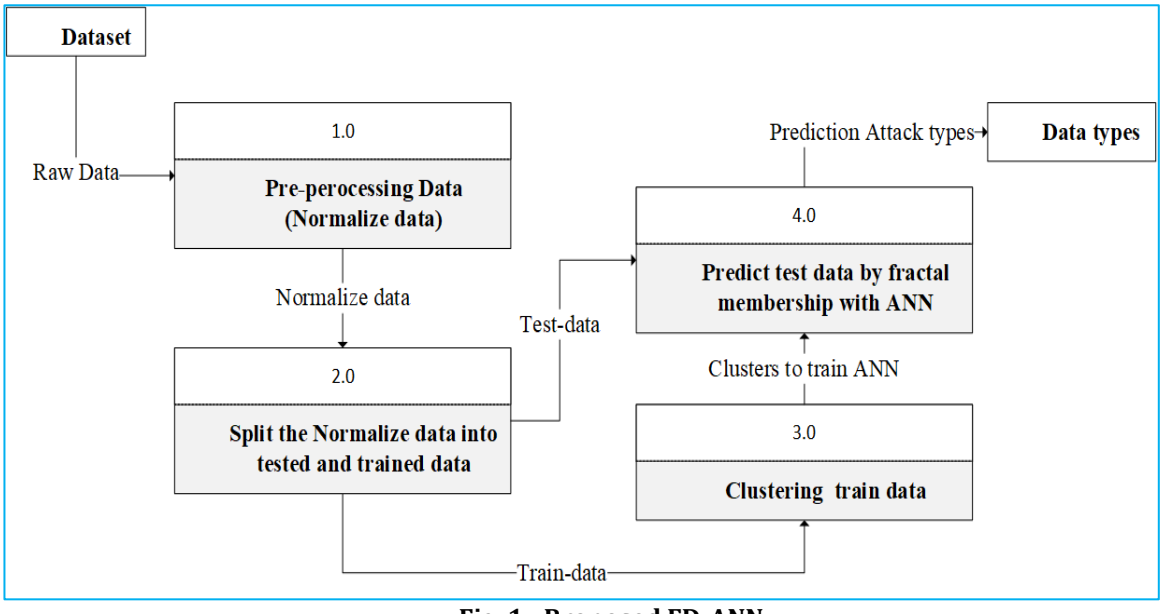


Fig. 1 - Proposed FD-ANN

5.1 Preprocessing

In this step, hot coding is proposed to convert the nominal attributes into numeric attributes for two datasets and analyze the network characteristics for normal and attacking traffic. The non-numeric attributes were converted to numeric attributes using a one-hot coding method. Normalizing the dataset for specific scaling values of each attribute in a certain range is a necessary step of the preprocessing phase. This strategy has the advantage of

removing biases from the datasets without affecting their statistical properties. To train the proposed model, the data are divided into two groups: Train and Test.

5.2 Clustering

The core idea of DPC for computing cluster centers is based on measuring local density and separation distance. These two metrics are based on the distance between any two points in the dataset. The original DPC calculates the distance between two data points using the Euclidean distance metric. However, the Euclidean distance leads to misclassification when the dataset is complex and has high-dimensional features. Therefore, the Gaussian kernel distance is used to calculate the distance between two points in high-dimensional data. This is the most important step in handling geometry, increasing accuracy and decreasing false alarm rate.

5.3 Predication test data by Fractal Membership function (FFM) and ANN

After assigning each point in the train data to the most appropriate cluster, we calculate the Membership function between the test sample and data point in each cluster. To find the FM needs to visit all points in each cluster. This process potentially requires some processing time and computational complexity, therefore, a novel FM based on fractal factor is proposed to overcome the time consuming and computational complexity. This process will extract subcultures ($FDC'_1, FDC'_2, \dots . FDC'_n$) from the main clusters ($DC_1, DC_2, \dots . DC_N$). The new subcultures have high similarity in behavior to the test item (test sample). The FFM has inversely proportional with the distance (d) between the nearest point in sub cluster (FDC'_n) and the test point. i.e. the point in a cluster with a small distance value to test item (TEST SAMPLE), the FFM degree will be greater than the point in a cluster with a high distance value. . Algorithm 1 illustrates the Fractal Membership function.

Algorithm 1 : Fractal Membership function

Input: TD←test dataset , $\boldsymbol{\beta} \leftarrow \boldsymbol{threshold}$

$\mathbf{DC_1}, \mathbf{DC_2}, \dots . \mathbf{DC_N}$ ←train clusters,

Output: Fractal Membership (FFM)

Begin {

For each i ∈ DC do {

$\mathbf{FDC_i}$=[]

$\boldsymbol{R'} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{DC_i}$ //$\boldsymbol{R'}$ is the mean of $\boldsymbol{DC}$,

$\boldsymbol{n}$ is dimensional of point $\boldsymbol{DC_i}$

$\mathbf{FDC_i} = \frac{(\sum_{i=1}^{n} (DC - R')^2)^2}{\sum_{i=1}^{n} (DC - R')^4}$ // fractal of train clusters

}// EndFor

FFM← []

For each t ∈ TD do {

$\mathbf{FFM_i} \leftarrow []$

$\mathbf{FDT}_t = \frac{(\sum_{t=1}^{n} (TD - R')^2)^2}{\sum_{t=1}^{n} (TD - R')^4}$ // fractal of test dataset

$\mathbf{DC'_1}, \mathbf{DC'_2}, \dots . \mathbf{DC'_n} \leftarrow \{\mathbf{subcluster_t} \mid \frac{|\mathbf{FDT}_t - \mathbf{FDC_N}|}{\mathbf{FDT}_t} \leq \boldsymbol{\beta}\}$

$\boldsymbol{d_t} \leftarrow []$

For each $\mathbf{DC'}$ in $\mathbf{subcluster_t}$ do {

$\boldsymbol{d_t}$[j] ← the nearest distance (**DC**′, t)

}// EndFor

For each d ∈ $\boldsymbol{d_t}$ and k ∈ cluster no.

$\mathbf{FFM_i[k]} \leftarrow \frac{1}{\sum_{k=1}^{n}(\frac{d}{d_t[k]})^2}$ // $\boldsymbol{n}$ is the number of clusters

}// EndFor

Append the $\boldsymbol{FFM_i}$ to FFM

}// EndFor

Return FFM

}}//End Algorithm

The result of the previous step is two sub-clusters DC1, DC2 from the training dataset. Each sub-cluster is used to train the corresponding ANN classifier. Since they were trained on different subsets, these ANNs are different from each other. An input layer, hidden layers and an output layer are different in each ANN. Aggregation is the process of joining modules (by linking the output of one module to the output of another). In our proposed system, the production of each ANN is multiplied by the corresponding membership and then another production of ANN. The result of predicting the test sample xj in each sub classifier ANN i is defined as ANNi (xj). The proposed FFM reduced the search space for the point the used to represent cluster. Figure 2 shows the scenario of pick up the point in cluster i the used in constructing the Fractal Membership function (FFM).

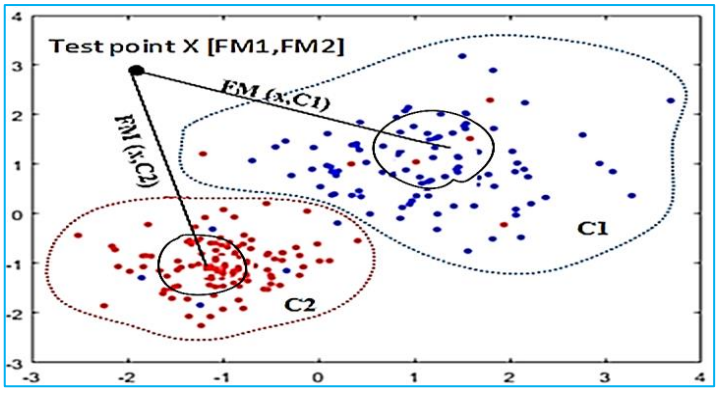


Fig. 2 - Example of points are selected for Fractal Membership function

5.4 Aggregation Model

Aggregation is the process of connecting modules (by linking outputs of one module to output of another). In our proposed system, the production of each ANN is multiplied with corresponding fuzzy membership, then another production of ANN. The predictions result of the test sample *xj* in each sub- ANN i classifier is defined as ANNi (xj). Figure (3) declare the aggregation method predict the final output.

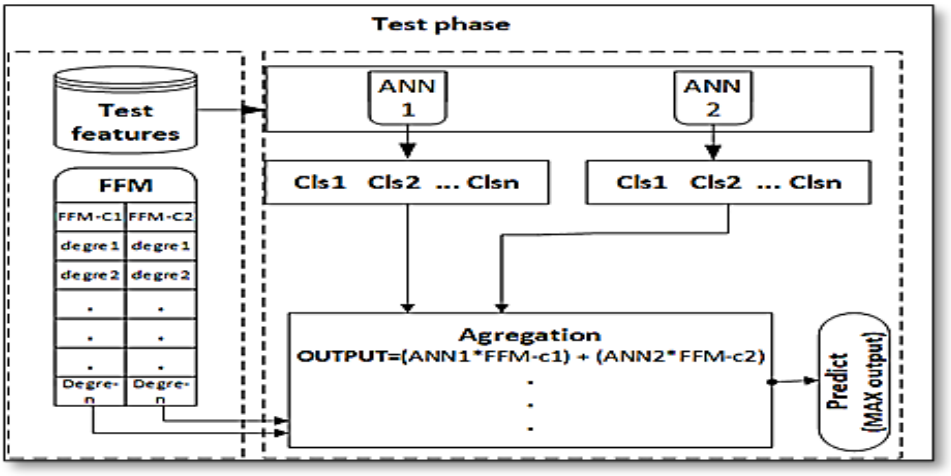


Fig. 3 - Illustrates The Aggregation Method Predict Final Output.

6. Experimental and Result Discussion

6.1 Data Sets Description

NSL_KDD[26], UNSW_NB15 [27] datasets consist of a large number of packets; the first one contains four major attacks, while the second has two million packets and contains nine major attacks. The features of these datasets include TCP/IP header information of TCP/IP suite. The following subsections present the details of these datasets.

6.1.1 NSL_KDD Data Set

The NSL_KDD data set is not the most recent, but it is a new version that addresses the shortcomings of the KDDCup 1999 data set. The NSL_KDD data set consists of four files are: the first file is called NSL_KDDTrain+ for training, which is the complete training data; the second file is called 20 Percent Training Set, which is also for training data. Moreover, two files for testing are called NSL_KDDTest+ and NSL_KDDTest-21. The description of NSL_KDD training and testing of our system is shown in table (1).

Table 1 - Description Of 20% NSL_KDD Training And Full Testing Data Set Files.

	Training Dataset	Testing Dataset	
Category	**20 % from full NSL_KDDTrain+**	**NSL_KDDTest-21**	**NSL_KDDTest+**
Dos	9234	4342	7458
Normal	13,449	2152	9711
Probe	2289	2402	2421
U2R	12	200	200
R2L	197	2754	2754
Total	25,192	11,850	22544

The (training) NSL_KDDTrain+ dataset has 22 different sub-types of attacks aggregated to four major types. In comparison, the (testing) NSL_KDDTest-21 and NSL_KDDTest+ datasets have 37 other sub-types of attacks, which means the testing dataset has an additional 15 novels, unknown attacks not available in the training datasets

6.1.2 UNSW_NB15 Data Set

The Australian Centre for Cyber Security (ACCS) developed this dataset to test network intrusion detection. It solved the problems of older benchmarks datasets like KDD_CUP99. The UNSW_NB2015 dataset contains a combination of real-world and synthetic attack activities of the network traffic. In comparison to the NSL_KDD dataset, the UNSW_NB15 has more forms of attacks. The UNSW_NB15 consists of two million packets; each packet has 49 attributes with the class label in the whole dataset, while the partitions UNSW_NB15_training-set.csv and UNSW_NB15_testing-set.csv is configured as partitions of the UNSW_NB15 dataset with only 42 attributes pulse label of class. The partition dataset has six attributes (srcip,, sport,, dstip,, dsport,, Stime, and Ltime,) removed from the total data set. In our proposed system, utilize the random state method to choose 20% of train data from a partition UNSW_NB15_training set (35069) as training data and 20% of test data from a partition UNSW_NB15_testing set (16466) as testing data in our proposed system such as table (2).

Table 2 - The Category Descriptions of UNSW_NB15' Training and Testing Set Files.

Category	20% UNSW NB15-train set	20% UNSW NB15-test set
Normal	11190	7349
Fuzzers	3599	1205
Analysis	406	155
Backdoors	336	114
DoS	2491	833
Exploits	6675	2223
Generic	8044	3780
Reconnaissance	2097	730
Shellcode	206	69
Worms	25	8
Total	35069	16466

6.2 Evaluation Metrics

A confusion matrix can calculate the performance measures for intrusion detection. The Confusion Matrix is a tool for summarizing a classifier's predicted performance on test data. The confusion matrix is used to determine all standard measures, such as True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). True Negative (TN) is the number of cases successfully forecasted as Normal class in cyber security, while True Positive (TP) is the number of instances correctly predicted as attacks, The number of normal occurrences identified as an attack is known as False Positive (FP), while the number of attack instances classified as normal is known as False Negative (FN). To evaluate the proposed system, standard metrics are used, such as measures like accuracy, precision, recall, and F1 score, while the Silhouette Index measure is presented to evaluate the quality of clusters[27].

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (7)$$

$$Recall = \frac{TP}{TP+FN} \quad (8)$$

$$Precision = \frac{TP}{TP+FP} \quad (9)$$

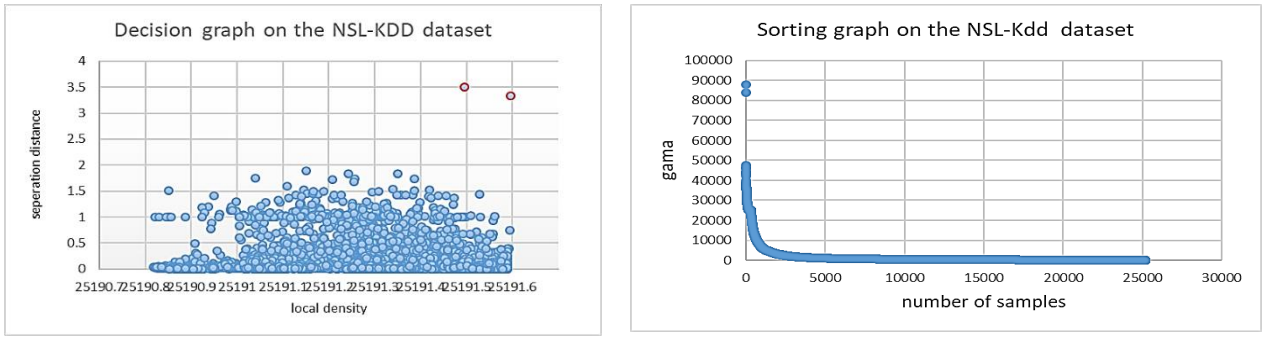
$$F1\ score = \frac{2\times Precision\times Recall}{Precision+Recall} \quad (10)$$

$$\text{Silhoute index } = \frac{x(i)-y(i)}{\max\{(x(i),y(i)\}} \quad (11)$$

Where x(i) is a lower distance among cluster and object and y(i) is the higher distance between object and clusters.

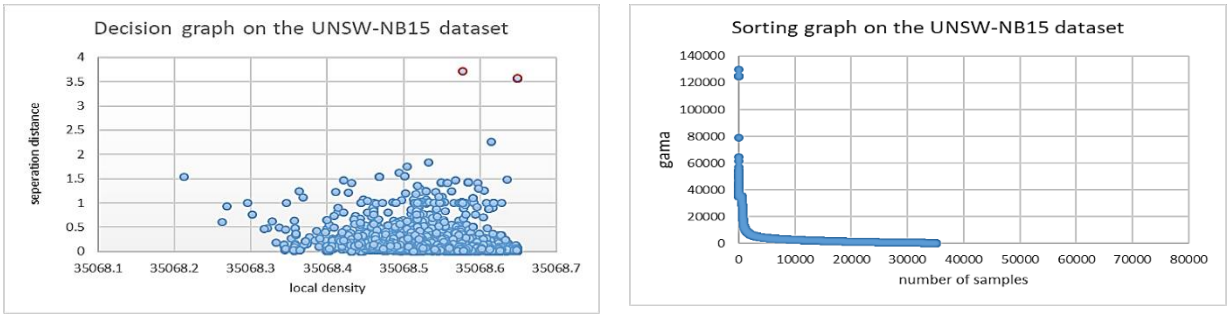
6.3 Analysis FFM on the performance of clustering model

After the preprocessing phase, we used an DPC algorithm to dynamically divide the training dataset into two clusters using a decision and sorting graphs based on feature similarity to break the training dataset's imbalance and improve the detection rate of low rate classes. The process of determining the peak points is based on plots of all data points in a decision graph and sorting graph, as shown in figures (4) for the NSL_KDD data set and (5) for the UNSW_NB15 data set.



(a) **(b)**

Fig.4 - (a)Decision Graphs ;(b) Sorting Graphs Based On NSL_KDD Train Data.



(a) **(b)**

Fig. 5 - (a) Decision Graphs; (b) Sorting Graphs Based On UNSE_NB15 Data Set with 20% Training.

The training data will distribute for each peak according to the most feature similarity, which breaks the imbalance of train data. Experiments are carried out to evaluate IDPC's performance selection for both datasets, which enhance detection performance and helps to increase the homogeneity of training data distribution of the clustering process. Figure (6) shows two Clusters of the NSL_KDD training dataset and Figure (7) Clusters Description Of UNSW_NB15 Data Set With Full Features.

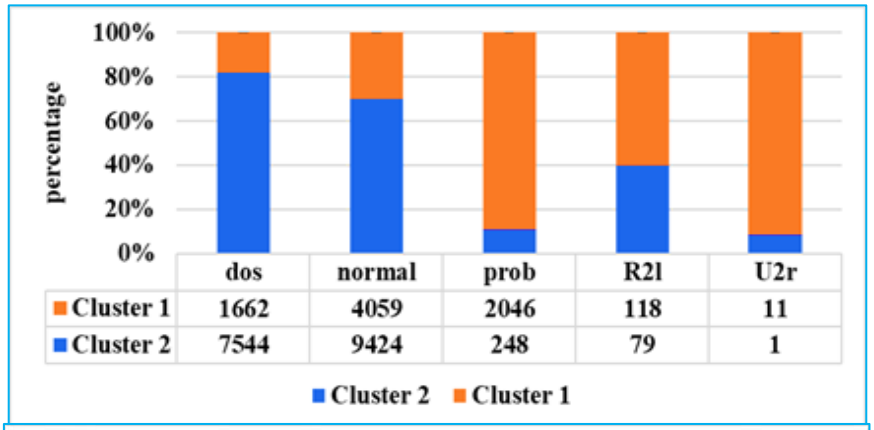


Fig. 6 - Clusters Description of NSL_KDD Data Set with Full Features.

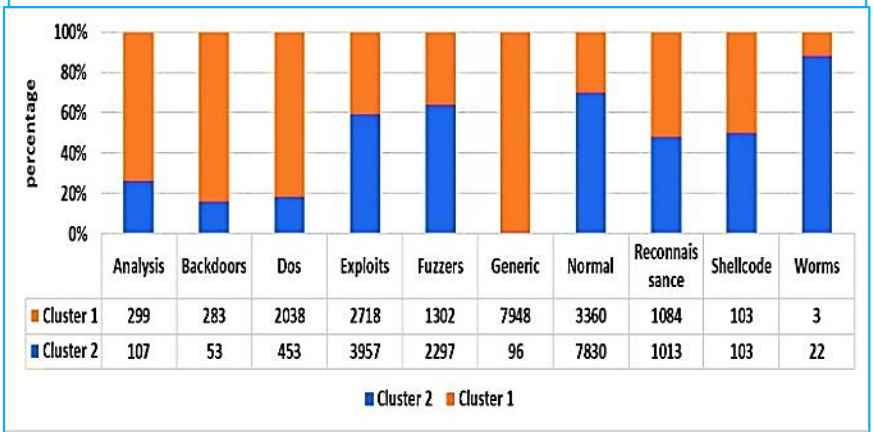
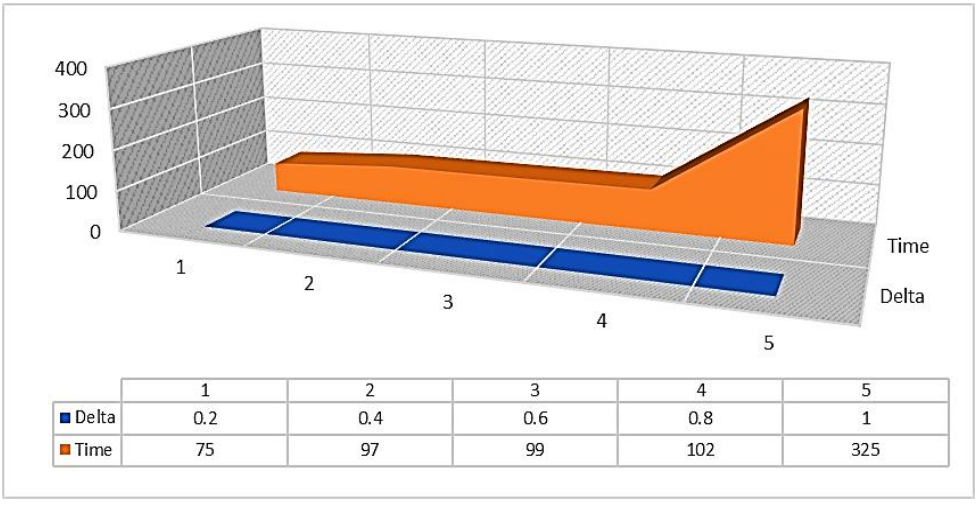


Fig. 7 - Clusters Description of UNSW_NB15 Data Set with Full Features.

After assigning each point in the training data to the most appropriate cluster chosen in the previous steps, the next step is to calculate the fractal membership matrix. In general, the system searches for the nearest point to test point (x_j) in each cluster (c_j). This operation required to visit each point in the cluster i and examine them to select the nearest point. Therefore, it consumes a large amount of time and computational complexity. So, we proposed gathering data with the same approximation behavior of the test point in one sub-cluster of the cluster i . The novel fractal membership matrix (FFM) is proposed for this purpose. It is technically based on the fractal phenomenon, rather than calculating the distance between each test sample and all the training data points in each cluster, which is computationally complex and time-consuming. To solve this problem, we apply the fractal factor (f) between the test sample x_j and each point in clusters x_i. The delta-F determines the similarity ratio between the fractal of (x_j) test and fractal points in the cluster (c_j). The different values of the proposed fractal metric are tested. The weights used to test Delta-F are 0.2, 0.4, 0.6, and 0.8. The optimal value is 0.2. Figure (8) shows the effect value of Delta-F on time required to build the membership matrix in the dataset (20 Percent Training Set of NSL_KDD).



	1	2	3	4	5
Delta	0.2	0.4	0.6	0.8	1
Time	75	97	99	102	325

Fig. 8 - Delta-F Based On Time Required To Building The Membership Matrix.

The experimentation that shows in the table (3) using (25192) records as the (20% from full NSL_KDDTrain+) as training data and (NSL_Test-21), (NSL_ Test+) as testing data to approve how the proposed fractal metric minimizes the time required for building the membership matrix.

Table 3 - The Comparison in time between building membership matrix with and without FFM.

Dataset	Train+Test			Delta F	Silhouette	Time(sc)
	Train	Test	total			
20% from full NSL_KDD training with NSL_ Test+	25192	22544	47736	NaN	0.2966	5049.35
	25192	22544		0.4	0.2965	4792.45
	25192	22544		0.2	0.2974	3361.42

20% from full NSL_KDD training with NSL_Test-21	25192	11850	37042	NaN	0.1912	3113.96
	25192	11850		0.4	0.1912	2568.18
	25192	11850		0.2	0.1926	1798.65

6.4 ANN Classifiers Analysis in a Training Stage for both Data Sets

Each cluster will be trained on a separate ANN classifier. The structure for both NSL_KDD and UNSW-NB15 data sets are:

The number of the hidden layers are two,

The optimizer of each ANN is Adam,

ReLU is the activation function that is used of the hidden layer,

softmax is the activation function that used for the output layer.

The number of neurons for hidden layers in the NSL_KDD data set is [80, 10,], while the number of neurons for hidden layers in the UNSW-N15 data set is [120, 20,].

6.5 Results Comparison

It is clearly shown that the result of the proposed system in Table (4) presents the overall performance based on accuracy, Precision, Recall and f1scor for the system for three testing data sets.

Table 4 - The Accuracy (%) For proposed System Based On Three Data Sets.

Dataset	Accuracy	Precision	Recall	f1scor
NSL_KDDTest-21	80.75	81.28	80.75	80.99
NSL_KDDTest+	89.96	89.94	89.69	89.82
UNSW_NB15	94.92	95.24	94.92	95.08

The experimented on these data sets shows how well our proposed system performed. We compared the results to five established machine learning classifiers such as K-Nearest Neighbor (KNN), Random Forest (RF), Support Vector Machine (SVM), Decision Tree (DT) and Artificial neural network (ANN). The Comparison Results are evaluated based on the four evaluation metrics (Accuracy, Precision, Recall, f1scor). The results are depicted in tables (5), (6) and (7).

Table 5 - Results Comparison Of Different Algorithms Using (NSL_KDDTest-21) Data Set.

Model	Dos	Normal	Probe	R2l	U2r	Accuracy	Precision	Recall	f1scor
KNN	71.2	67.6	64.2	11.7	4.0	54.21	71.47	54.21	61.66
RF	59.4	88.1	61.9	6.7	0.5	51.94	79.63	51.94	62.87
SVM	57.0	68.1	63.2	0	0	46.10	50.64	46.10	48.27
DT	57.8	68.9	62.1	22.4	9.5	51.69	65.06	51.69	57.61
ANN	71.1	82.9	76.4	13.2	4.5	59.81	75.92	59.81	66.91
Our system	73.1	98.0	90.6	74.4	29.5	80.75	81.28	80.75	80.99

As shown in table (5), our proposed system for the (NSL_KDDTest-21) dataset shows a high detection rate of low rate classes such as U2R and R2L attack types, which have detection accuracy of 29.5% and 74.4%, respectively. It is produces a higher accuracy of 80.75% than the other compared methods.

Table 6 - results comparison of different algorithms using NSL_KDDTest+ dataset.

Model	Dos	Normal	Probe	R2l	U2r	Accuracy	Precision	Recall	f1scor
KNN	82.9	92.4	64.5	11.7	4.0	75.68	78.32	75.68	76.98
RF	75.8	97.3	62.2	6.7	0.5	74.56	81.01	74.56	77.67

SVM	74.9	92.8	63.5	0.0	0.0	71.60	65.70	71.60	68.52
DT	75.3	93.0	62.4	22.4	9.5	74.53	75.38	74.53	74.95
ANN	83.2	96.1	76.6	13.2	4.5	78.83	81.03	78.83	79.92
Our system	83.8	99.4	90.7	74.9	22.0	89.96	89.94	89.69	89.82

As shown in table (6), our proposed system for the (NSL_KDDTest+) dataset shows a high detection rate of low rate classes such as U2R and R2L attack types, which has detection accuracy of 22.0% and 74.9%, respectively. It is produces a higher accuracy of 89.96% than the other compared methods.

Table 7 - results comparison of different algorithms using UNSW_NB15 dataset.

Class	KNN	RF	SVM	DT	ANN	Our system
Analysis	16.1	1.9	00.0	1.2	0.0	88.7
Backdoors	14.0	7.8	00.0	21.0	0.0	96.4
Dos	23.2	19.4	00.0	26.5	25.3	100
Exploits	65.9	80.0	76.3	67.8	79.2	92.2
Fuzzers	55.7	60.3	59.8	59.8	59.8	59.5
Generic	96.1	96.2	97.2	96.7	96.1	99.0
Normal	85.1	94.6	85.1	91.1	87.9	100
Reconnaissance	66.0	83.5	36.7	81.0	88.0	97.0
Shellcode	23.1	62.3	00.0	62.3	30.4	45.2
Worms	00.0	00.0	00.0	12.5	0.0	62.5
Accuracy	77.51	84.59	76.38	81.79	81.72	94.92
Precision	82.54	86.20	75.02	85.91	84.60	95.24
Recall	77.51	84.59	76.38	81.79	81.72	94.92
f1scor	79.95	85.39	75.69	83.80	83.13	95.08

As shown in table (7), our proposed system for the (UNSW_NB15) dataset shows a high detection rate of low rate classes such as Analysis, Backdoors, Shellcode, and Worms attack types, which have detection accuracy of 88.7%, 96.4%, 45.2% and 62.5%, respectively. It produces a higher accuracy of 94.92% than the other compared methods.

6.6 Comparison with the Related Works

To prove the case of handling the imbalance problem of multi classes by enhancing the detection rate of low rate classes such as R2l, U2r, Analysis, Backdoors, Worms, and Shellcode with better performance of our proposed system. We compare our proposed system to nine intrusion detection models. On the data sets (NSL KDDTest-21), (NSL KDDTest+), and (UNSW NB15), our proposed system outperforms based on the Accuracy, Precision, Recall, f1scor, and detection rate of low rate classes. The Comparison results based on these datasets are shown in tables (8), (9), and (10), respectively.

Table 8 - Comparison results with different models based on (NSL_KDDTest-21) Dataset.

Author	Train data	Test-21 data	Accuracy	D.R of R2l	D.R of U2r
T. Ma .et al [21] (2016)	25192	11850	44.55	1.5	0.9
C. Yin .et al [23] (2017)	125,973	11850	64.67	N/A	N/A
Y. Yang .et al [25] (2019)	25192	11850	66.18	34.9	6.0
Y. Yang.et al [25] (2019)	25192	11850	57.45	13.2	0.5
C. Tang .et al [26] (2020)	125,973	11850	77.57	N/A	N/A
Our system	25192	11850	80.75	74.4	29.5

Table 9 - Comparison results with different models based on (NSL_KDDTest+) Dataset.

Author	Train data	Test data	Accuracy	D.R of R2l	D.R of U2r
Y. Yang .et al [25] (2019)	25192	22544	82.08	17.2	6.5
C. Yin .et al [23] (2017)	125,973	22544	81.29	24.6	11.5
Y. Yang.et al [25] (2019)	25192	22544	80.82	12.5	5.5
C. Tang .et al [26] (2020)	125,973	22544	82.14	N/A	N/A
Chao Liu. et al [28] (2021)	125,973	22544	85.24	73.8	25.7
Isra .et al [29] (2021)	125,973	22544	81.44	N/A	N/A
Our System	25192	22544	89.96	74.9	22.0

Table 10 - Comparison results with different models based On (UNSW_NB15) dataset

Author	Train data	Test data	Accuracy	D.R of Analysis	D.R of Backdoors	D.R of Shellcode	D.R of Worms
N.Moustafa .et al [22] (2016)	105204	32932	78.47	N/A	N/A	N/A	N/A
H. M. Anwer .et al [24] (2018)	175341	82332	88.30	N/A	N/A	N/A	N/A
Y. Yang .et al [25] (2019)	35,069	16,466	90.21	0.0	0.8	39.4	11.1
S. M. Kasongo.et al [27] (2020)	131,506	82,332	77.51	N/A	N/A	N/A	N/A
Our system	35069	16466	94.92	88.7	96.4	45.2	62.5

7. Conclusion

The proposed system is crucial when the resources of an organization are connected to the Internet, where the security objectives (CIA) are required, because the problem of network attacks in an organization leads to a degradation of network performance. The expected results of this study will present a new proposed system of hybrid techniques. It presents a new framework that combines more than one of the data mining techniques that can classify network attacks multiple times and provide better detection rate of low rate attacks. The performance of the proposed system will provide a new metric for distributing the test data; this performance is necessary to reduce the time and computational complexity. Several metrics are used to evaluate the proposed system. The obtained results have shown that the best recognition system is achieved by using hybrid techniques based on DPC clusters algorithm using fractal membership degree function with the help of ANN classification.

References

[1] S. Aljawarneh, M. Aldwairi, and M. Bani, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," J. Comput. Sci., vol. 25, pp. 152–160, 2018, doi: 10.1016/j.jocs.2017.03.006.

[2] S. M. Hadi, A. H. Alsaeedi, M. I. Dohan, R. R. Nuiaa, S. Manickam, and A. S. D. Alfoudi, "Dynamic Evolving Cauchy Possibilistic Clustering Based on the Self-Similarity Principle (DECS) for Enhancing Intrusion Detection System," Int. J. Intell. Eng. Syst., vol. 15, no. 5, pp. 252–260, 2022, doi: 10.22266/ijies2022.1031.23.

[3] I. Škrjanc, S. Ozawa, T. Ban, and D. Dovžan, "Large-scale cyber attacks monitoring using Evolving Cauchy Possibilistic Clustering," Appl. Soft Comput. J., vol. 62, pp. 592–601, 2018, doi: 10.1016/j.asoc.2017.11.008.

[4] D. B. Rawat, R. Doku, and M. Garuba, "Cybersecurity in Big Data Era: From Securing Big Data to Data-Driven Security," IEEE Trans. Serv. Comput., vol. 14, no. 6, pp. 2055–2072, 2021, doi: 10.1109/TSC.2019.2907247.

[5] R. B. Basnet, R. Shash, C. Johnson, L. Walgren, and T. Doleck, "Towards detecting and classifying network intrusion traffic using deep learning frameworks," J. Internet Serv. Inf. Secur., vol. 9, no. 4, pp. 1–17, 2019, doi: 10.22667/JISIS.2019.11.30.001.

[6] S. Mohammadi, H. Mirvaziri, M. Ghazizadeh-Ahsaee, and H. Karimipour, "Cyber intrusion detection by combined feature selection algorithm," J. Inf. Secur. Appl., vol. 44, pp. 80–88, 2019, doi: 10.1016/j.jisa.2018.11.007.

[7] N. Moustafa and J. Slay, "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," Inf. Secur. J., vol. 25, no. 1–3, pp. 18–31, 2016, doi: 10.1080/19393555.2015.1125974.

[8] A. Singh, K. Chatterjee, and S. C. Satapathy, "An edge based hybrid intrusion detection framework for mobile edge computing," Complex Intell. Syst., 2021, doi: 10.1007/s40747-021-00498-4.

[9] V. Kumar, D. Sinha, A. K. Das, S. C. Pandey, and R. T. Goswami, "An integrated rule based intrusion detection system: analysis on UNSW-NB15 data set and the real time online dataset," Cluster Comput., vol. 23, no. 2, pp. 1397–1418, 2020, doi: 10.1007/s10586-019-03008-x.

[10] A. Cheema, M. Tariq, A. Hafiz, M. M. Khan, F. Ahmad, and M. Anwar, "Prevention Techniques against Distributed Denial of Service Attacks in Heterogeneous Networks: A Systematic Review," Secur. Commun. Networks, vol. 2022, 2022, doi: 10.1155/2022/8379532.
[11] A. Heidari and M. A. Jabraeil Jamali, "Internet of Things intrusion detection systems: a comprehensive review and future directions," Cluster Comput., 2022, doi: 10.1007/s10586-022-03776-z.
[12] A. K. Balyan et al., "A Hybrid Intrusion Detection Model Using EGA-PSO and Improved Random Forest Method," Sensors, vol. 22, no. 16, pp. 1–20, 2022, doi: 10.3390/s22165986.
[13] R. R. Nuiaa, A. H. Alsaeedi, S. Manickam, and D. E. J. Al-Shammary, "Evolving Dynamic Fuzzy Clustering (EDFC) to Enhance DRDoS_DNS Attacks Detection Mechnism," Int. J. Intell. Eng. Syst., vol. 15, no. 1, pp. 509–519, 2022, doi: 10.22266/IJIES2022.0228.46.
[14] R. R. Nuiaa, S. Manickam, and A. H. Alsaeedi, "Distributed reflection denial of service attack: A critical review," Int. J. Electr. Comput. Eng., vol. 11, no. 6, pp. 5327–5341, 2021, doi: 10.11591/ijece.v11i6.pp5327-5341.
[15] Y. Yang, K. Zheng, C. Wu, X. Niu, and Y. Yang, "Building an effective intrusion detection system using the modified density peak clustering algorithm and deep belief networks," Appl. Sci., vol. 9, no. 2, 2019, doi: 10.3390/app9020238.
[16] C. Tang, N. Luktarhan, and Y. Zhao, "Saae-dnn: Deep learning method on intrusion detection," Symmetry (Basel)., vol. 12, no. 10, pp. 1–20, 2020, doi: 10.3390/sym12101695.
[17] C. Liu, Z. Gu, and J. Wang, "A Hybrid Intrusion Detection System Based on Scalable K-Means+ Random Forest and Deep Learning," IEEE Access, vol. 9, pp. 75729–75740, 2021, doi: 10.1109/ACCESS.2021.3082147.
[18] I. Al-Turaiki and N. Altwaijry, "A Convolutional Neural Network for Improved Anomaly-Based Network Intrusion Detection," Big Data, vol. 9, no. 3, pp. 233–252, 2021, doi: 10.1089/big.2020.0263.
[19] J. L. Lin, "Accelerating density peak clustering algorithm," Symmetry (Basel)., vol. 11, no. 7, pp. 1–18, 2019, doi: 10.3390/sym11070859.
[20] A. Qiu and Z. Wang, "Optimization of Density Peak Clustering Algorithm Based on OpenMP," J. Softw., vol. 13, no. 3, pp. 168–179, 2018, doi: 10.17706/jsw.13.3.168-179.
[21] N. Oliveira, I. Praça, E. Maia, and O. Sousa, "Intelligent cyber attack detection and classification for network-based intrusion detection systems," Appl. Sci., vol. 11, no. 4, pp. 1–21, 2021, doi: 10.3390/app11041674.
[22] A. H. Alsaeedi, A. H. Aljanabi, M. E. Manna, and A. L. Albukhnefis, "A proactive metaheuristic model for optimizing weights of artificial neural network," Indones. J. Electr. Eng. Comput. Sci., vol. 20, no. 2, pp. 976–984, 2020, doi: 10.11591/ijeecs.v20.i2.pp976-984.
[23] S. M. Kasongo and Y. Sun, "Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset," J. Big Data, vol. 7, no. 1, 2020, doi: 10.1186/s40537-020-00379-6.
[24] A. S. Saljoughi, M. Mehvarz, and H. Mirvaziri, "Attacks and intrusion detection in cloud computing using neural networks and particle swarm optimization algorithms," Emerg. Sci. J., vol. 1, no. 4, pp. 179–191, 2017, doi: 10.28991/ijse-01120.
[25] A. L. A. A. H. A. A. H. A. Mehdi Ebady Manna, "A proactive metaheuristic model for optimizing weights of artificial neural network," Indones. J. Electr. Eng. Comput. Sci., vol. 20, no. 2, pp. 976–984, 2020, doi: 10.11591/ijeecs.v20.i2.pp976-984.
[26] UNB, "NSL-KDD."
[27] A. S. Choudhary, P. P. Choudhary, and S. Salve, "A Study on Various Cyber Attacks and A Proposed Intelligent System for Monitoring Such Attacks," Proc. 3rd Int. Conf. Inven. Comput. Technol. ICICT 2018, pp. 612–617, 2018, doi: 10.1109/ICICT43934.2018.9034445.