Babylonian Journal of Machine Learning
Vol.2026, **pp.** 110-131
DOI: https://doi.org/10.58496/BJML/2026/011; ISSN: 3006-5429
https://mesopotamian.press/journals/index.php/BJML/index

A Preemptive Ensemble Learning Framework for Knowledge-Based Network Intrusion Detection

Maad Kamal Al-anni[1,*], Ammar A Al-Hamadani[1], Rafah M. Almuttairi[3], ,Alharith A Abdullah [2]

[1]*Computer Engineering department, College of Engineering, Al-Iraqia University, 7366 Haifaa, Baghdad, Iraq*
[2]*College of Information Technology, University of Babylon, 51002, Hilla,Iraq*
[3]*College of Information Technology Engineering, Al-Zahraa University for Women, 56001, Karbala,Iraq*

ARTICLEINFO

Article History
Received 20 Jan 2026
Revised 18 Mar 2026
Accepted 10 May 2026
Published 11 Jul 2026

Keywords
Intrusion Detection System (IDS)
Machine Learning
Network Traffic Analysis
Density Peak Clustering (DPC)
Fractal Fuzzy Membership (FFM)

ABSTRACT

The rapid growth of Internet-connected systems has increased the complexity, diversity, and volume of network traffic, making intrusion detection a significant cybersecurity challenge. Traditional machine learning-based intrusion detection systems often suffer from high-dimensional data, severe class imbalance, and poor detection of low-frequency attacks. To address these limitations, this paper proposes a hybrid framework, FD-ANN, which integrates Density Peak Clustering (DPC), Fractal Fuzzy Membership (FFM), and Artificial Neural Networks (ANNs).

The framework employs wrapper-based feature selection to eliminate irrelevant features and reduce dimensionality. DPC partitions the training data into homogeneous clusters, preserving minority attack structures while mitigating class imbalance. Separate ANN classifiers are trained for each cluster, and their outputs are combined through an adaptive FFM weighting mechanism that dynamically assigns weights according to local data distributions.

Experiments on the NSL-KDD and UNSW-NB15 benchmark datasets demonstrate the effectiveness of the proposed approach. FD-ANN achieved accuracies of 82.47%, 92.24%, and 95.59% on NSL-KDDTest-21, NSL-KDDTest+, and UNSW-NB15, respectively. The framework also significantly improved the detection of minority attack classes while reducing false alarm rates. Comparative results against KNN, RF, SVM, DT, and conventional ANN models confirm that FD-ANN provides an effective, scalable, and robust solution for modern network intrusion detection.

1. INTRODUCTION

Expansion of the Internet throughout the world and growing demand in information exchange and access, has increased the volume of data and costs to process it incredibly. With data being frequently kept in cloud systems or across databases [1], it has ended up being necessary to ensure the protection of data. In particular, personal information must be securely protected and authenticated to prevent unauthorized access and potential intrusions [2]. In order to keep things secure and private, most organizations have implemented special policies for data access control and authentication [3]. However, the surge of ransomware, zero-day attacks and similar sophisticated threats are well presented in the media [4]. Zero-day attacks pose a significant challenge in modern intrusion detection systems due to their unknown signatures and evolving behavior, in this work, the term “zero-day attacks” refers to malicious activities exploiting previously unknown vulnerabilities, while “zero-day exploits” denote the

* *Corresponding author. Email: muad.rashed@aliraqia.edu.iq*

underlying techniques or code used to perform such attacks. Conventional protections like antivirus and firewalls are no longer enough. That is, there is a trend towards multi-layered applications which are deployed on different platforms.
Intrusion Detection Systems (IDS) [5] play a critical role in protecting networked environments by detecting and responding to malicious activities. When it monitors network-wide threats, the IDS is known as a Network Intrusion Detection System (NIDS) [6]. Given the context, NIDSs are mainly divided into two broad categories, Signature-Based IDSs (SBIDSs) and Anomaly-Based IDSs (ABIDSs). Any NIDS has the main goal of detecting, tracking, and possibly preventing attacks within the network.
Recent advances in machine learning and deep learning have significantly enhanced the capabilities of intrusion detection systems. Previous studies have demonstrated the effectiveness of these techniques in identifying cyberattacks across diverse environments, including Internet of Things (IoT) platforms [6] and cloud computing systems [7]. Nevertheless, the detection performance of conventional learning models remains challenged by high-dimensional network traffic, class imbalance, and the presence of low-frequency attack categories.

The motivation for this work stems from the need to detect continuous and diverse cyberattacks in network traffic data that is very high-dimensional. We develop an intelligent intrusion detection system to mitigate this challenge by employing a hybrid feature selection, Density Peaks Clustering (DPC), and a deep neural network classifier in order to deliver superior classification performance, minimize false positives, which enhances system adaptability in dynamic network environments via a self-similarity-based estimator. To address these challenges, this paper presents a hybrid multi-classification-based detection framework. In detail, the proposed model improves the low-rate attack recognition through a clustering technique based on data mining and a classification one based on neural networks. A summary of the main contributions of this work are:

i FS to reduce dimensionality without reducing the accuracy of detection; address class imbalance and enable low-frequency attack detection via efficient broadcast detection.

ii application of a powerful data mining clustering algorithm DPC. The use of more complex distance measure implicitly learns in the DPC framework due to the intrinsic complexity of real network traffic.

iii Finally, the installation of a novel Fuzzy Fractal Membership (FFM) model to optimize the cost and complexity of computing processes.

We assess the performance of our proposed method using its capability to classify multiple types of attacks and on the accuracy of low-rate attack detection. The performance of the complete system is shown to outperform existing state-of-the-art methods, finally validating its overall efficiency.
As one of several crucial elements of this area, the Intrusion Detection System (IDS) is an important part in secure the network environment. Based on its points of placement, it is called a Network Level (NIDS) [8]. In general, NIDS can be divided into two types:

- Signature-Based Intrusion Detection Systems (SIDS) detect attacks by matching traffic patterns and behaviors that are known. These patterns match the attack signatures recorded before and can only work against known attacks [1].
- Anomaly-Based Intrusion Detection Systems (AIDS) pertain to analyzing network behavior based on monitoring when certain limits or norms are broken. The system then alerts the whenever suspicious activity is detected.

The main aim of any NIDS is to monitor for attacks taking place in a network; identify, trace, and then stop the attacks. In order to overcome some of the most critical challenges, this research designs, implements and evaluates a hybrid, multi-classification-centric detection system. In this work, a knowledge-based clustering combined with a neural network-based recognition and a representational connectivity system is promoted to improve the detection of low-rate attacks.

1.1 Motivation

Detecting diverse and evolving cyber threats is challenging in high-dimensional network data, but has strong potential for unsupervised detection based on the study of this. The goal is designing an intelligent intrusion detection system based on the hybrid feature selection, Density Peaks Clustering (DPC), and deep neural network classifiers to achieve more accurate classification performance, minimize false positives, and improve adaptability to changing network conditions using a membership estimation method based on similarity patterns in the data.

1.2 Research Gap and Problem Definition

Despite ongoing progress in network intrusion detection systems (NIDS), adapting to changing network conditions remains a significant challenge:

1. Defective Firewalls: Most companies firewalls are useless against volumetric and protocol attacks (DoS or ICMP / TCP flooding which leverage saturation bandwidth via misuse of ports). These types of attacks can exhaust system resources while also evading (initially) firewalls [2] [3].

2. Dataset Limitations To train the NIDS, a well-chosen and powerful dataset has yet to be selected. The performance and usability of any model for intrusion detection primarily relies on the quality and precision of available datasets [4].

3. Handling Missing Values: The network traffic data has been with lots of missing values which reduces the accuracy of detection. Incomplete or noisy data affects a classifier's ability to learn useful patterns.

4. Multi-Class Classification: Network datasets exhibit a major challenge because of highly imbalanced data with so many attack classes vs normal instances Minority-class attacks, which are often of higher importance than easy-to-predict majorities, are therefore underrepresented, leading classifiers to favour the dominated patterns and poorly modeling even very severe intrusions with minority classes [5].

   This does not mean that more general terminology is used to refer the former techniques, but instead, some specific terms as *Fractal Fuzzy Membership (FFM)* are preserved in order to light the model out with its unique design and difference from classic fuzzy approaches.

1.3 Contribution

The main contributions of this work are outlined below. Feature selection mechanism is used to reduce the dimension of dataset and enhance system performance [8]. Secondly, in order to combat class imbalance and further boost the accurate detection of those low-frequency attacks, data mining clustering algorithm: Density Peaks Clustering (DPC) is employed. In addition, it evaluates the real networks'complex data by improving distance metric DPC algorithm. A new novel Fractal Fuzzy Membership (FFM) mechanism is accordingly proposed to minimize both processing time and computation complexities. The proposed method is evaluated in terms of attack classification and detection of low-rate attack. Finally, its general performance is confirmed via a comparison with the rest of relevant contemporary methods.
Unlike conventional approaches that rely on generic density-based clustering or standard fuzzy membership functions, the proposed framework employs Density Peak Clustering (DPC) and a novel Fractal Fuzzy Membership (FFM) mechanism to explicitly capture both structural density variations and multi-scale complexity in network traffic data.

2. RELATED WORKS

In this part some contributions of those researchers who provided IDS techniques based on follow upper methodological techniques are reviewed. Most of the studies use data mining clustering algorithms for assignment and few of the other studies replace data mining clustering with a hybrid framework of a neural or deep learning classifier. Finally, the conversation highlights the importance of hybrid methods, evaluation performance experiment, and the datasets used.
Y. Yang et al. A hybrid IDS based on modified density peak clustering and DBNs were presented in [9]. MDPCA, modified density peak clustering, is used to cluster large-scaleand high-dimensional network data. MDPCA tackles the challenge of excessive training data and has the ability to cope with sample imbalance by grouping data with similar characteristics into subsets, with each subset served by an isolated sub-DBN classifier. Fuzzy membership weighting is employed to combine outputs of these classifiers In this paper, NSL-KDDTest+, NSL-KDDTest-21, and UNSW-NB15 datasets were employed to validate the proposed MDPCADBN model.
Chaofei et al. SAAE-DNN: an intrusion detection framework based on stacked autoencoder and deep neural network with attention mechanism [10]. Detection capability is improved by the attention mechanism, and latent representations are extracted by the SAE. While the SAAE module enhances feature extraction and enhances DNN detection through hidden layer weight initialization. On NSL-KDD dataset SAAE-DNN outperformed traditional ML algorithms like Random Forest and Decision Tree (82.14% accuracy on multi-class detection).
Sydney et al. Based on the above similarity portrayal analysis, Husain and Dimitoglou [11] employed XGBoost on the UNSW-NB15 dataset to develop an efficient intrusion detection model, demonstrating that feature importance

derived from XGBoost significantly enhances classification performance.

CHAO et al. [12] designed a cascade Intrusion Detection System based on k-means, Random Forest, and deep learning methods to overcome inefficiency in traditional IDS. As normal and malicious activity is distinguished using distributed computing on the Spark platform, DL techniques based on convolutional neural networks (CNN), long short term memory (LSTM) and others classify abnormal traffic into attack categories. On NSL-KDD, at multi-target classification setting, their system achieved 85.24% accuracy.
El-Shafai et al. [13] introduced an architecture for intrusion classification, including a Three-layer CNN (TCNN) and transfer learning models, such as VGG16, VGG19, ResNet50 and ResNet152 on NSL-KDD dataset. The TCNN provides 99.81% accuracy, 100% precision/recall and only 0.004% false alarm rate This shows that a lightweight CNN architecture along with transfer learning can perform exceedingly well for real-time wireless cyber security applications.

3. METHODOLOGY

3.1 Density Peak Clustering (DPC) method

Density Peak Clustering (DPC), a specific density-based clustering algorithm, is employed to identify density peaks based on local density (ρ) and distance to higher-density points (δ).
Clustering techniques such as *Density Peak Clustering* (DPC) have been proposed to identify regions with high local data density [14] [15]. The DPC algorithm operates by analyzing statistical properties of the dataset and is particularly effective in handling areas with continuous data points. The local density is estimated using a Gaussian kernel function defined as in Equation 1:

$$K(x) = \exp\left(-\frac{d^2}{2\sigma^2}\right) \tag{1}$$

where d represents the distance between data points, and σ is the Gaussian kernel bandwidth controlling smoothness; small values yield fragmented clusters, while large values cause over-smoothing.
In its initial formulation, DPC computes the local density and separation distance by first evaluating the pairwise distances between data objects.
Let $X = \{x_1, x_2, \ldots, x_n\}$ denote a dataset consisting of n data points, where each data point x_i is characterized by M attributes. Accordingly, x_{ij} refers to the j-th attribute of the i-th data point. The distance between two data points, x_i and x_j, is defined as shown in Equation 2 [16].

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{M} (x_{ik} - x_{jk})^2} \tag{2}$$

The local density ρ_i of each data point can be defined as in Equation 3:

$$\rho_i = \sum_{j \in S} \exp\left(-\frac{d_{ij}^2}{C_d^2}\right) \tag{3}$$

where d_{ij} denotes the distance between points i and j, and C_d represents the cutoff distance, which is chosen within the neighborhood radius to determine the parameter.
In the second step, the separation distance δ_i is computed according to Equation 4 [17]:

$$\delta_i = \begin{cases} \min\limits_{j:\rho_j > \rho_i} (d_{ij}), & \text{if such a } j \text{ exists} \\ \max\limits_{j} (d_{ij}), & \text{otherwise} \end{cases} \tag{4}$$

Clustering coefficient = $\gamma_i = \rho_i \times \delta_i$, points with the highest values of γ_i are selected as cluster centers. After calculating the local density ρ_i and separation distance δ_i for each data point, cluster centers are identified as those with simultaneously high values of ρ_i and δ_i. To achieve this, you use a 2-dimensional decision graph: local density (x-axis) and separation distance (y-axis) [18]. After determining the cluster centers, all remaining data points are assigned to their nearest cluster center. However, clustering high-dimensional data remains a challenging task due to the well-known curse of dimensionality, where meaningful distance differences gradually disappear and Euclidean distances tend to become increasingly similar. This phenomenon weakens the representation of true density structures and may reduce cluster stability. Furthermore, larger distances amplify the effects of noise and

irrelevant features, leading to misclassification and the loss of local structural information, which is particularly important for distance-based clustering algorithms.
To address these challenges, the proposed framework employs a Gaussian kernel distance measure that transforms Euclidean distances into local exponential similarity values controlled by the kernel parameter σ. This transformation alleviates the adverse effects of high dimensionality by emphasizing local relationships while reducing the influence of distant data points. Consequently, the approach mitigates the uniformity of distances in high-dimensional spaces and enables more reliable density estimation within the Density Peak Clustering (DPC) framework.
As a result, the clustering process becomes less sensitive to dimensionality and more effective at preserving the intrinsic nonlinear manifold structure of the data. This enhancement contributes directly to improving the performance of the subsequent intrusion detection and classification stages.

3.2 Artificial Neural Networks

Artificial Neural Networks (ANNs) [19] are models derived from the working of human brain neurons. These models are classifiers by number of neurons and layers, weights and biases and hyper paramaters like learning rate, epochs, batch size etc. A typical ANN architecture is composed of an input layer, one or more hidden layers, and an output layer. Following the training process, model evaluation on test data is a crucial step to assess its overall effectiveness.

3.3 Multi-Layer Perceptron

A *Multi-Layer Perceptron* (MLP) is a feed-forward neural network distinguished from the simple perceptron by the incorporation of nonlinear activation functions and the presence of multiple layers. Its architecture typically consists of three types of layers: an input layer, one or more hidden layers, and an output layer [20] [21]. Neurons within the MLP employ nonlinear activation functions, while the network is trained using the backpropagation (BP) algorithm.
The functioning of an MLP can be described in two main stages. The first stage, forward propagation, is responsible for generating predictions and is mathematically expressed in Equations 5 and 6 [22]:

$$\text{Net}_i = \sum_{j=1}^{n} w_{ij} x_j + \mu_i, \quad j = 1, 2, \ldots, n \tag{5}$$

$$Y_i' = \varphi(\text{Net}_i) \tag{6}$$

where Net_i denotes the weighted sum of the inputs x_j, w_{ij} represents the connection weights, μ_i is the bias term, φ is the nonlinear activation function, n is the number of inputs, and Y_i' is the resulting output.
The second stage is the training phase, which employs the backpropagation (BP) algorithm to update the weights. During each epoch, the BP algorithm adjusts the weights based on the error, which is computed using Equation 7:

$$E = \frac{1}{n} \sum_{i=1}^{n} \left(y_{\text{actual},i} - y_{\text{desired},i} \right)^2 \tag{7}$$

where E is the mean squared error between the actual output $y_{\text{actual},i}$ and the desired output $y_{\text{desired},i}$.

4. DATA SETS DESCRIPTION

Despite having so many datasets, the two most commonly used ones are NSL-KDD and UNSW-NB15 [23] [24] that consist of a huge amount of network traffic. The NSL–KDD dataset is divided into four attack classes, while the UNSW-NB15 dataset consists of nearly two million network flow records in nine designed attack classes. Both groups are built upon characteristics defined by the TCP/IP protocol stack but again they focus on features in the TCP/IP header. These datasets are described in detail in the following subsections. We performed experiments on the NSL-KDD dataset (v2. 0) on both the test+ subset and the more difficult Test-21 evaluation subset [25]. Despite this it is still a better preparation, on NSL-KDD which main limitation is not being the most recent benchmark of all, compared to such benchmark, an improvement of the original KDD Cup 1999dataset consisting of different types of known limitations, including redundant records or biased distribution of attack types.
The full training set is organized within four main files: the NSL-KDDTrain+ file contains the complete training data, while the reduced 20% Training Set is also utilized for training, as well as two separate testing files: the NSL-KDDTest+ and NSL-KDDTest-21 files. Number of samples from these files is summarized in Table 1.

TABLE 1. Summary of NSL-KDD Dataset: Training (20%) and Testing Samples

Attack Category	Training (20%)	Testing Samples	
		tbl/leaders/begintbl/leaders/end	
		Test-21	Test+
DoS	9,234	4,342	7,458
Normal	13,449	2,152	9,711
Probe	2,289	2,402	2,421
U2R	12	200	200
R2L	197	2,754	2,754
Total	**25,192**	**11,850**	**22,544**

`NSL-KDDTrain+` has a total of 22 attack subtypes but they are grouped into 4 categories of attacks. As a comparison, 37 of the attack subtypes can be found in `NSL-KDDTest+` and `NSL-KDDTest-21` which are not part of the training data, adding in 15 new attack variants. Experiments were conducted using the UNSW-NB15 dataset (Version 1.0), a contemporary benchmark dataset for intrusion detection [26]. This dataset was generated using the IXIA PerfectStorm tool and includes both normal network traffic and nine different categories of attack [24]. UNSW-NB15 was designed by the Australian Centre for Cyber Security (ACCS) which aims to assist with the evaluation of Intrusion Detection Systems and to overcome the shortfalls of previous benchmarks like KDD Cup 1999.

It includes both real and synthetic network attack traffic, offering a broader range of attack categories than NSL-KDD. The dataset consists of two million records, each with 49 attributes and a class label. However, the provided files (`UNSW-NB15_training-set.csv` and `UNSW-NB15_testing-set.csv`) contain only 42 attributes plus the class label, as six attributes (`srcip`, `sport`, `dstip`, `dsport`, `Stime`, and `Ltime`) were removed. For experiments, 20% of the training data (35,069 samples) and 20% of the testing data (16,466 samples) are randomly extracted, as summarized in Table 2.

TABLE 2. Summary of the UNSW-NB15 Dataset Files

Attack Category	Training Samples (20%)	Testing Samples (20%)
Normal	11,190	7,349
Fuzzers	3,599	1,205
Analysis	406	155
Backdoors	336	114
DoS	2,491	833
Exploits	6,675	2,223
Generic	8,044	3,780
Reconnaissance	2,097	730
Shellcode	206	69
Worms	25	8
Total	**35,069**	**16,466**

5. THE PROPOSED SYSTEM

The proposed system model is structured into three main phases, designed to train the system for attack detection and classification. It begins with data preprocessing, followed by clustering using an artificial neural network, and concludes with an aggregation phase to predict the attack types. The overall architecture of the proposed network intrusion detection system is illustrated in Figure 1.

5.1 Preprocessing Phase

In this stage, two datasets (NSL-KDD, UNSW-NB15) are used as inputs to the preprocessing phase of the proposed NIDS. The following subsections explain the preprocessing steps.

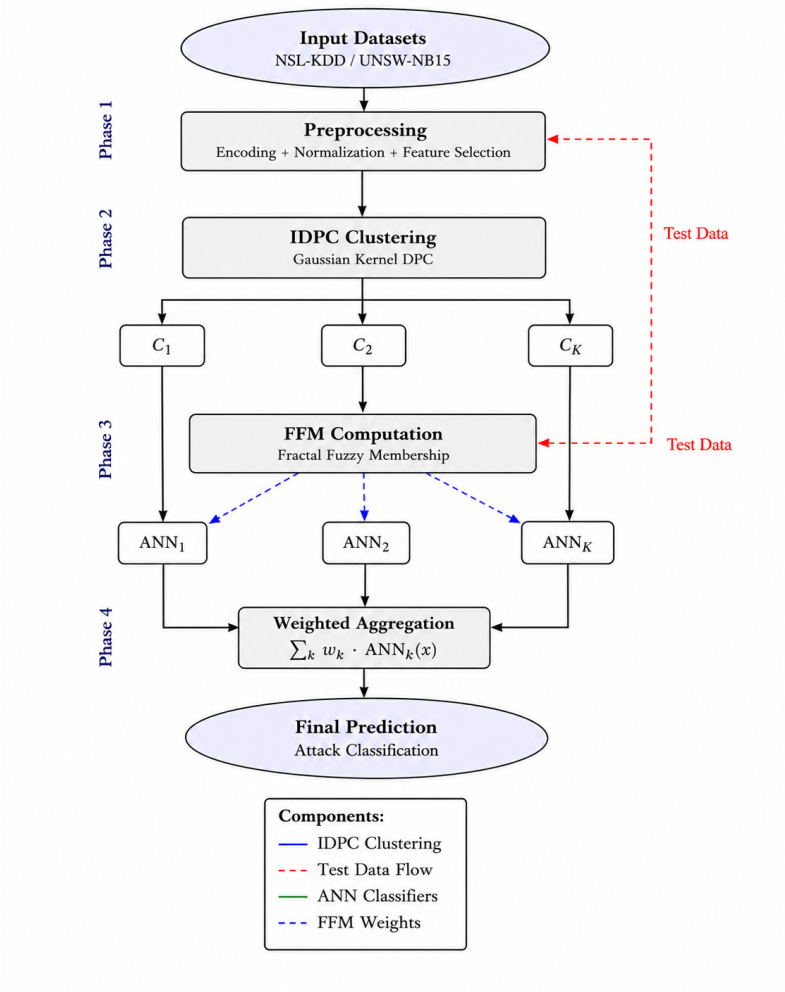


Fig. 1. FD-ANN pipeline for intrusion detection. Four phases: (1) preprocessing, (2) IDPC clustering, (3) FFM computation with parallel ANN training, (4) weighted aggregation for final prediction.

5.1.1. Data Transformation using One-Hot encoding

The datasets were processed using one-hot encoding to ensure the conversion of categorical attributes into machine-readable numerical vectors. This transformation was essential for enabling a comprehensive examination of network traffic behaviors under both benign and adversarial conditions. For the NSL-KDD dataset, the process included emasculating three categorical features blockaded 38 continuous features. Likewise, the same transformation was performed on 39 continuous features in addition to three categorical attributes in the UNSW-NB15 dataset to keep consistent methodology across both datasets.

5.1.2. Data Normalization using Min-Max Method

The preprocessing process requires each attribute to be scaled to a specified range. Bias is removed from the data through scaling without changing its statistical properties.

5.1.3. Feature Selection

The Proactive Bat Search Optimization Algorithm (ABA) is used to choose the most relevant feature before clustering. When stagnation occurs in the search process, it is suitable for sustaining exploration through regeneration of the population by replacing the existing one [8]. *Note on feature selection impact:* Feature selection is not a key contribution of this work. Table 10 presents the ablation study, which indicates that accuracy drops only 1–3% without feature selection for all datasets. This agrees with the evidence discussed previously that the performance gained by the proposed FD-ANN method is via clustering (IDPC), fuzzy weighting (FFM) and aggregation mechanisms, rather than feature reduction.

5.2 Improved Density Peak Clustering(IDPC) Algorithm

As outlined in Section (3.1), Density Peak Clustering (DPC) identifies cluster centers by evaluating local density and separation distance, both computed from pairwise data distances. While conventional DPC relies on Euclidean distance, this often causes misclassification in high-dimensional, complex datasets. To overcome this, Gaussian kernel distance is employed to better capture data geometry, enhance accuracy, and reduce false alarms. The training-stage steps of the IDPC are detailed in Algorithm 1.

Algorithm 1 IDPC for the Training Stage

$data \leftarrow$ training dataset, $\sigma \leftarrow$ Gaussian kernel parameter, $d_c \leftarrow$ neighborhood threshold K sub-training sets: $\{dc_1, dc_2, \ldots, dc_K\}$

In this stage, the definitions of ρ_i, δ_i, and γ_i remain unchanged; only the distance metric is replaced by the Gaussian kernel distance.

for each data point $i \in data$ **do** Compute pairwise distance:

$$d_G(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

Determine cutoff distance: $dc = \lfloor cd \times |data| \rfloor$

Estimate local density:

$$\rho_i = \sum_{j \in S} \exp\left(-\frac{d(x_i, x_j)^2}{dc^2}\right)$$

for each data point i **do** Compute separation distance:

$$\delta_i = \begin{cases} \min(d(x_i, x_j)), & j > i \\ \max(d(x_i, x_j)), & \text{otherwise} \end{cases}$$

for each data point i **do** Sort ρ in descending order Sort δ in descending order Compute $\gamma_i = \rho_i \times \delta_i$: $\gamma_i = \rho_i \times \delta_i$ Select points with highest γ_i as cluster centers Assign remaining points to nearest cluster centers using Gaussian kernel distance **return** $\{dc_1, dc_2, \ldots, dc_K\}$

5.3 Fractal Fuzzy Membership (FFM)

After assigning each point in the training dataset to its best-matching cluster, the traditional Fuzzy Membership (FM) for a test sample is calculated by evaluating its similarity to all points within each cluster. This process can be computationally intensive and time-consuming. To counter this, an alternative approach is proposed that utilizes a fractal factor to limit processing complexity and time. This method identifies representative subclusters (denoted as $\text{FDC}'_1, \text{FDC}'_2, \ldots, \text{FDC}'_n$) from the principal clusters ($\text{dc}_1, \text{dc}_2, \ldots, \text{dc}_n$). These subclusters are characterized by a high behavioral resemblance to the test sample.

Unlike traditional fuzzy and/or fractal techniques, the proposed Fuzzy Fractal Membership (FFM) system directly integrates individual multi-scale dynamic fractal patterns with a fuzzy logic function. This approach provides a preventive, knowledge-based ensemble weighting mechanism that preserves the structural complexity of network intrusions.

The core of this method is an **Fractal Fuzzy Membership (FFM)** which is inversely proportional to the distance (d) from the nearest point in a relevant subcluster (FDC'_n) to the test sample. Formally, this relationship can be expressed as:

$$\text{Weight} \propto \frac{1}{d}$$

Thus, a shorter distance to the test sample results in a higher weight value, signifying a stronger membership association, all distance computations in the FFM module use the Gaussian kernel distance $d_G(\cdot, \cdot)$ defined previously. The complete computational procedure for this adaptive fuzzy weighting mechanism is outlined in Algorithm 2.

The system parameters were adjusted using sensitivity analysis and experimental judgments to ensure stability and efficiency. During the training phase of the Density Peak Clustering (DPC), σ is a parameter used to control the smoothness of local density estimation; small values result in overly fragmented clusters, while large values excessively smooth the data, losing the underlying structure. Therefore, σ was selected empirically through a sensitivity analysis that balances cluster stability with detection performance in subsequent classification stages.

For the Fractal Fuzzy Membership (FFM) module, the parameter β tunes the granularity of fractal-induced membership values. Low values increase sensitivity to complex network dynamics but may introduce noise, whereas high values delay attack detection. To optimize β, an independent validation set was used to minimize the false alarm rate while maintaining high detection capability.

The cutoff distance d_c is defined as $d_c = \lfloor c_d \times |\text{data}| \rfloor$, where c_d is the proximity threshold used to partition the data into K subsets. This scaling ensures density estimation remains stable with varying dataset sizes. Sensitivity analyses confirmed that these parameters exhibit stable performance within reasonable ranges, demonstrating the reliability and portability of the proposed framework.

Algorithm 2 Fractal Fuzzy Membership (FFM) Function

Require: $TD \leftarrow$ test dataset, $\beta \leftarrow$ threshold, $DC_1, DC_2, \ldots, DC_N \leftarrow$ train clusters (The fractal dimension computation employs stability constant ϵ, normalization factor α, and thresholds D_{th} and τ to ensure numerical stability and regulate membership weighting. These parameters were empirically determined through preliminary experiments and cross-validation.)

Ensure: Fractal Fuzzy Membership (FFM)

function FFM($TD, \beta, DC_1, DC_2, \ldots, DC_N$)
  for each $i \in DC$ **do**
    $FDC_i \leftarrow [\,]$
    $R' \leftarrow \frac{1}{N} \sum_{i=1}^{N} DC_i$ ▷ R' is the mean of DC
    $n \leftarrow$ dimensional of point DC_i
    $FDC_i \leftarrow \frac{\left(\sum_{i=1}^{n} (DC - R')^2\right)^2}{\sum_{i=1}^{n} (DC - R')^4}$ ▷ fractal dimension of train clusters
  end for
  $FFM \leftarrow [\,]$
  for each $t \in TD$ **do**
    $FFM_i \leftarrow [\,]$
    $FDT_t \leftarrow \frac{\left(\sum_{t=1}^{n} (TD - R')^2\right)^2}{\sum_{t=1}^{n} (TD - R')^4}$ ▷ fractal dimension of test dataset
    $DC'_1, DC'_2, \ldots, DC'_n \leftarrow \{subcluster_t \mid \frac{|FDT_t - FDC_N|}{FDT_t} \leq \beta\}$
    $d_t \leftarrow [\,]$
    for each DC' in $subcluster_t$ **do**
      $d_t[j] \leftarrow$ the nearest distance (DC', t)
    end for
    for each $d \in d_t$ and $k \in$ cluster no. **do**
      $FFM_i[k] \leftarrow \frac{1}{\sum_{k=1}^{n} \left(\frac{d}{d_t[k]}\right)^2}$ ▷ n is the number of clusters
    end for
    Append FFM_i to FFM
  end for
  return FFM Function

5.4 ANN Classification

Two separate sub-clusters of the planning matrix, dc_1 and dc_2, are created from the training dataset, and a different ANN classifier is trained for each sub-cluster. Due to training across different data subsets, the full and truncated ANNs possess unique architectural features with respect to their input, hidden, and output layers.

5.5 Aggregation Model

The aggregation of the system is done with a series connection between modules, where the output of one module is used as input to the next. In the proposed architecture first, you multiply each Artificial Neural Network (ANN) output with its corresponding fuzzy membership value and this product, then multiplies by the next ANNs followed by this same process, and then multiplying this product by the output of the subsequent ANN in the sequence. Formally, the prediction for a test sample x_j made by a sub-classifier ANN_i is denoted as $\text{ANN}_i(x_j)$. The overall aggregation process for generating the final prediction is illustrated in Figure 2.

6. EXPERIMENTAL WORK

A hybrid intrusion detection system (IDS) has been developed for attack detection, utilizing IDPC clustering and artificial neural network (ANN) algorithms along with a fractal fuzzy membership function. Two datasets were employed in the experiments.

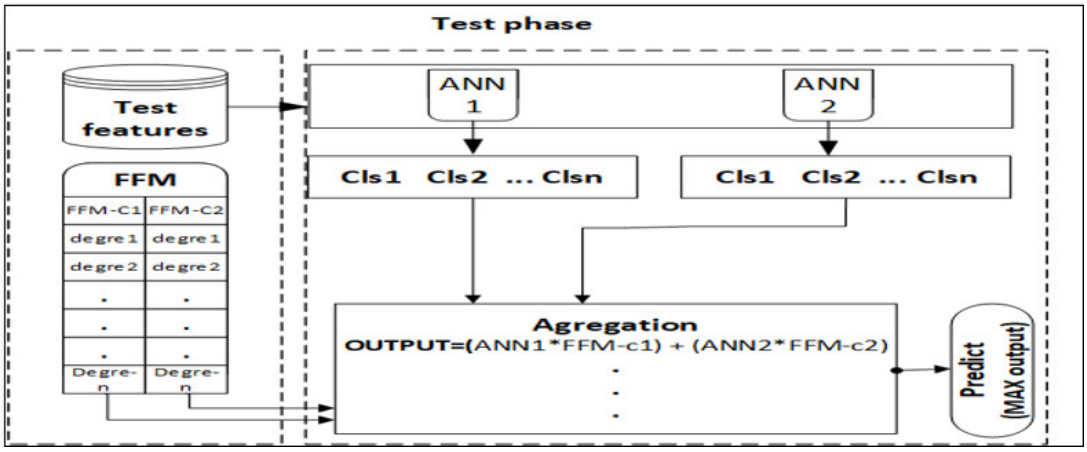


Fig. 2. Illustrates The Aggregation Method Predict Final Output.

The use of 20% of the total dataset for both training and testing was adopted based on methodological soundness as well as practical considerations related to efficiency and reproducibility, To ensure methodological clarity, 20% of the training dataset was randomly selected for model training, while a separate 20% of the predefined testing dataset was used exclusively for evaluation, with no overlap between the two subsets.

First, the data sets used are large enough so that they repeat their frequency under similar operating conditions. Thus, a 20% subset is sufficient to maintain the general statistical properties as well as behavioral structure of network traffic without serious loss of information. Preliminary experiments suggest stability of performance metrics with larger number of samples, justifies this choice.

Second, by resizing the datasets we help reduce the overhead of training and improve throughput of each respective stage – DPC, FFM and ANN – to achieve a rather more efficient overall execution. Such efficiency improvements are important for running more than one independent run to assess statistical significance and conduct a sensitivity analysis. This also makes the reproducibility given a fixed-seed randomization possible by running experiments over different sub-samples. In such cases reporting results over several runs help avoid sampling bias and makes our reported results more believable and stronger against idiosyncratic behaviour of particular data-strategies combinations.

For its first experiment, only 20% of the NSL-KDD training data was used in addition to other datasets that did scale as testing data. The second experiment conducted on dataset UNSW-NB15, using 20% of its training set and 20% of its test set for all the tests to be trained. The association between data pre conditions (DPC), fuzzy logic based feature measurement (FFM) and artificial neural network (ANN) is possible only through a thorough substructure which typical of the overall structure due to an inherent provision by them for essential needs of intrusion detection [27] [28].

DPC minimizes overlaps and improves data representation by purely focusing on local high-density patterns in the geometric structure of heterogeneous network traffic without any prior assumption about cluster shapes. Finally, FFM quantifies uncertainty as structure and multi-layer in the network dynamics by embedding fractal nature into fuzzy membership values, generating flexible weights reflecting the intrinsic complexity of network function. Finally, the stored representations with structure and weights are used by ANN to learn more generalizable and stable decision boundaries via non-linear mappings. This complementary combination steadily diminishes complexity while maintaining discriminative power, hence attaining better attack detection capability over the case of applying any of these components solely; In this research, The 20% subset was randomly selected from UNSW-NB15 dataset and in order to achieve the statistical significance and prevent sampling bias, several repetitive experiments were performed.

6.1 Metrics of evaluation

The overall computational cost of the proposed FD-ANN framework remains efficient despite its hybrid design. Feature selection and DPC clustering are performed offline during training, with complexities of $\mathcal{O}(Nd)$ and $\mathcal{O}(N^2)$, respectively. During inference, the Fractal Fuzzy Membership (FFM) computation operates on representative subclusters, reducing the test-time complexity to $\mathcal{O}(kd)$ per sample, where $k \ll N$. The ANN classifier incurs a standard forward-pass cost of $\mathcal{O}(Hd)$. As a result, FD-ANN achieves improved detection performance with modest computational overhead, which is consistent with the observed low execution times reported in the

experimental results.
To assess the robustness of the experimental results, each experiment was repeated N independent times using different random seeds. The average and standard deviation of performance metrics were recorded. Statistical significance between the proposed method and baseline models was evaluated using a paired t-test (and the Wilcoxon signed-rank test when normality assumptions were not satisfied), with a significance level of $p < 0.05$.
A confusion matrix is used to evaluate the performance of intrusion detection systems by providing a comprehensive view of a classifier's performance on test datasets. It includes key metrics such as True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). In cybersecurity, TN represents the number of normal instances correctly identified, while TP indicates the number of attack instances correctly detected. FP corresponds to normal activities incorrectly classified as attacks, and FN represents attacks misclassified as normal activities. To assess the proposed system, standard performance metrics including accuracy, precision, recall, and F1-score are employed. Additionally, the Silhouette index is used to evaluate the quality of generated clusters [25] [29].

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{8}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{9}$$

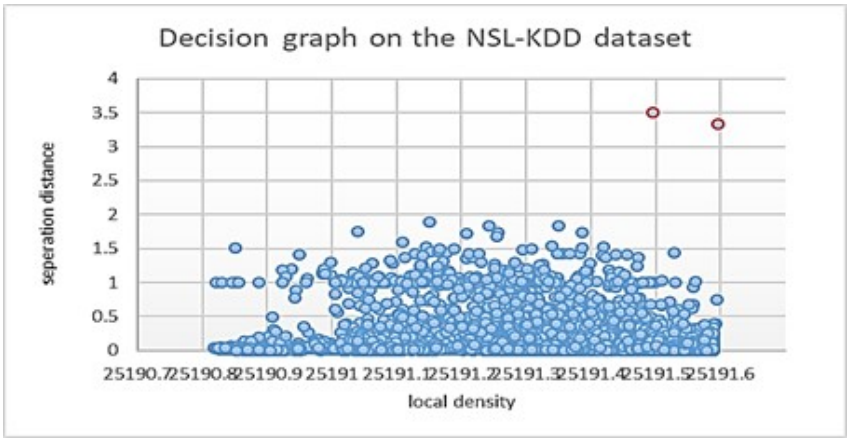
$$F1\text{-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}$$

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

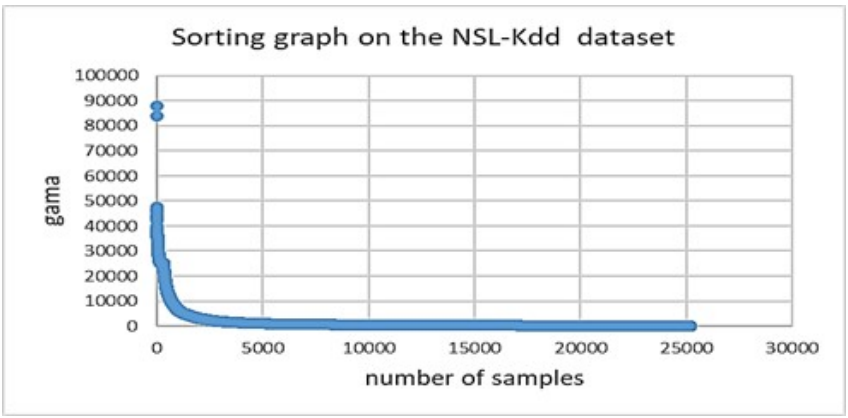
where $x(i)$ denotes the shortest distance between an object and the cluster it belongs to, and $y(i)$ represents the largest distance between that object and all other clusters.
It should be noted that performance metrics are reported under different evaluation settings (binary vs. multi-class classification, and across distinct dataset splits such as NSL-KDDTest+ and NSL-KDDTest-21). Therefore, reported values are not directly comparable across all experiments.

6.2 IDPC Clustering Analysis in a Training Stage

Following the preprocessing stage, the IDPC algorithm was applied to dynamically partition the training dataset into two clusters. This was achieved using decision and sorting graphs, which were constructed based on feature similarity. These graphs helped address data imbalance and improved the detection of minority classes. Peak points were identified by analyzing the decision and sorting graphs for each data point, as illustrated in Figure 3 for the NSL-KDD dataset(Fig. 3a- 3b : Density-distance (a) and gamma ranking (b) plots for NSL-KDD dataset) and Figure 4 for the UNSW-NB15 dataset(Fig. 4a- 4b : Density-distance (a) and gamma ranking (b) plots for NSL-KDD dataset).
The DPC algorithm handles clustering based on local density and distance to the highest-density points, allowing for the identification of density peaks in the data. However, it fails to completely deal with stratification irregularities, because classes that do not frequently occur often generate small and low-density clusters that may go unnoticed during the center selection process. These classes can therefore be under-represented, motivating the use of resampling techniques such as SMOTE [30], which can be integrated into DPC through density-based weighting. Within this framework, FFM increases the influence of minority classes by assigning adaptive weights according to their structural complexity, ensuring that they are adequately preserved during the propagation process of neural network training. This approach enables a higher detection rate for minority classes without heavily perturbing the dense data distribution.

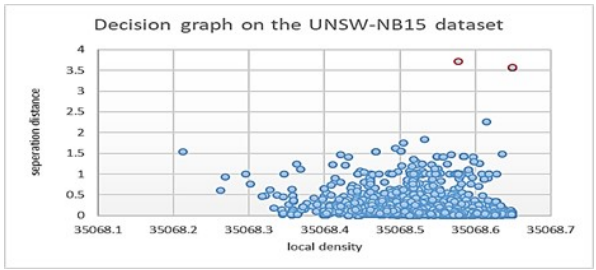


(a) Density–Distance Decision Plot for the NSL-KDD Training Dataset.

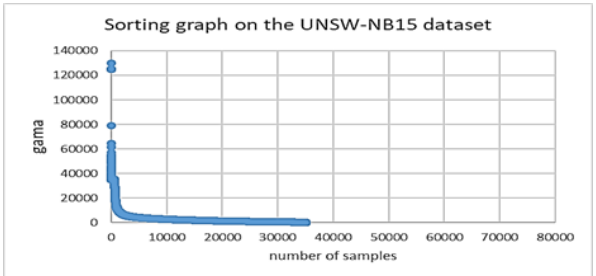


(b) Gamma Ranking Plot for the NSL-KDD Training Dataset

Fig. 3. Density–Distance Decision Plot (a) and Gamma Ranking Plot (b) for the UNSW-NB15 training dataset. The plots illustrate cluster center identification based on local density, distance, and ordered values, showing clearer separation and improved cluster discrimination.



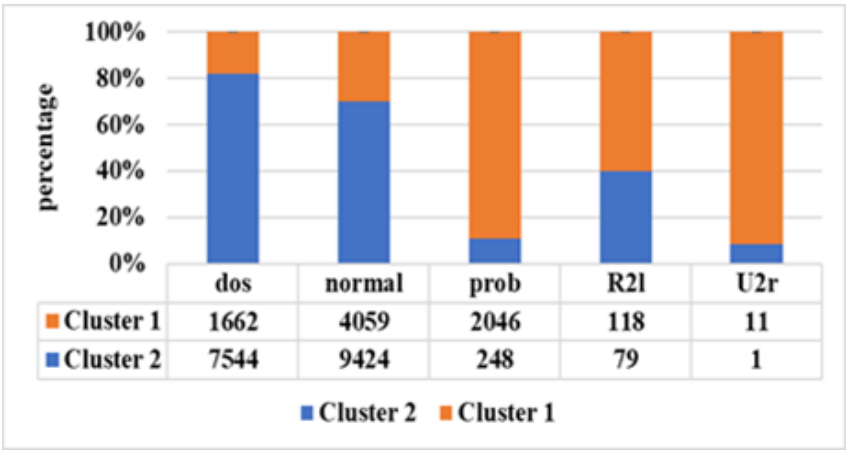
(a) Density–Distance Decision Plot for the UNSW-NB15 Training Dataset.



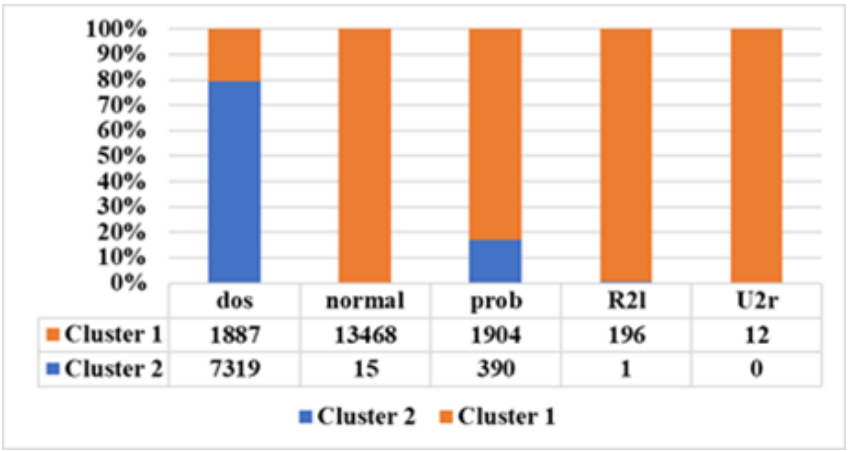
(b) Gamma Ranking Plot for the UNSW-NB15 Training Dataset

Fig. 4. Density–Distance Decision Plot (a) and Gamma Ranking Plot (b) for the UNSW-NB15 training dataset. The plots illustrate cluster center identification based on local density, distance, and ordered values, showing clearer separation and improved cluster discrimination.

Applying proactive feature selection during the preprocessing stage improves the clustering process by eliminating irrelevant features that negatively impact detection performance. The data is distributed across each peak based on the highest level of feature similarity, which helps partially balance the dataset. Experiments were conducted to evaluate the IDPC algorithm with proactive feature selection for both datasets. This approach enhances detection accuracy and promotes greater homogeneity in data distribution throughout clustering. Figure 5a shows the two clusters of the NSL-KDD training data using all features, while Figure 5b illustrates the two clusters generated from the NSL-KDD training data using 94 selected features, improving the distribution of the Normal, R2L, and U2R classes.



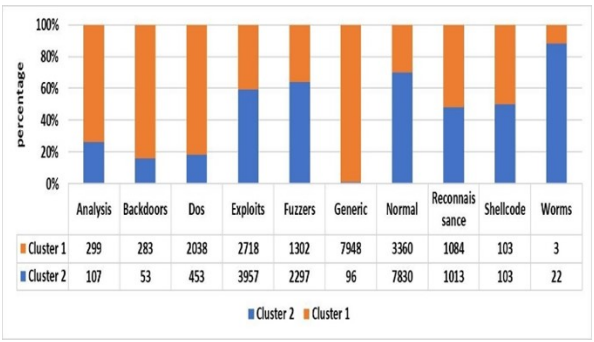
(a) full features of NSL-KDD Data Clusters.



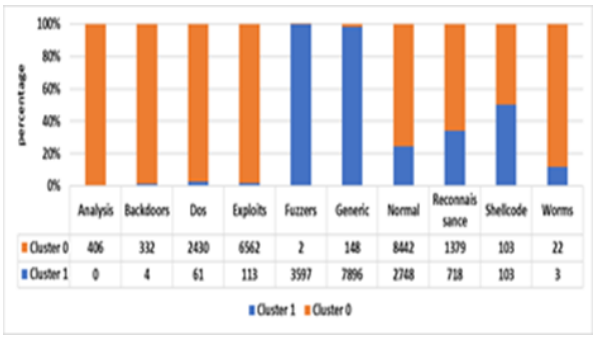
(b) 94 Features Of NSL-KDD Data Clusters.

Fig. 5. NSL-KDD training data clusters (a) using all 122 features — minority classes (R2L, U2R) are dispersed and overlapping with majority classes; (b) using 94 selected features — improved cluster compactness and separation, particularly for low-frequency attack classes.

Figure 6a illustrates two clusters of the UNSW-NB15 training dataset using all features. In contrast, Figure 6b presents two clusters of the UNSW-NB15 training dataset generated with 158 selected features, which improves the distribution of training data across most classes, particularly the Analysis, Backdoor, and Fuzzer classes. After each data point has been allocated to the best-fit cluster as identified in the previous step, the next step is to calculate the fractal fuzzy membership matrix.



(a) Clusters of the UNSW-NB15 dataset using all features.



(b) 158 Features Of UNSW-NB15 Data Clusters.

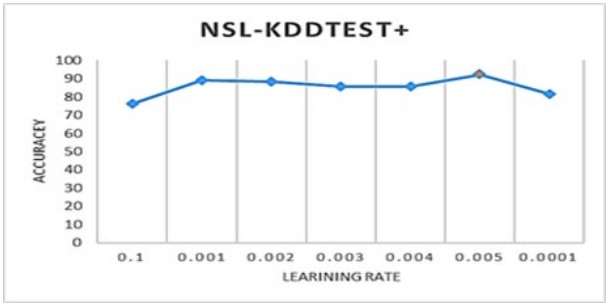
Fig. 6. UNSW-NB15 training data clusters (a) using all features — minority classes (Analysis, Backdoor, Shellcode, Worms) are highly dispersed and overlapping with majority traffic; (b) using 158 selected features — significantly improved cluster compactness and separation, enabling better detection of low-frequency attack classes.

6.3 Training phase and Analysis of ANN Classifiers

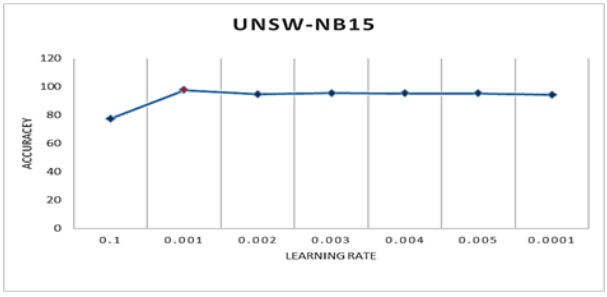
Each of the generated clusters is trained using a separate ANN classifier. The architectures applied to both the UNSW-NB15 and NSL-KDD datasets share the following characteristics:

- Each network is composed of two hidden layers.
- The Adam optimizer is utilized for training.
- The ReLU activation function is employed in the hidden layers.
- The output layer uses the softmax activation function.

For the NSL-KDD dataset, the two hidden layers contain $[80, 10]$ neurons, while the UNSW-NB15 dataset adopts $[120, 20]$ neurons in its hidden layers. Two important hyperparameters that must be tuned are the learning rate and the number of neurons. An excessively high learning rate during training may cause oscillations, preventing convergence, whereas a very small learning rate can slow down convergence. The default learning rate of the Adam optimizer in `sklearn` is 0.001. Therefore, several candidate values are considered for experimentation: $\{0.1, 0.001, 0.002, 0.003, 0.004, 0.005, 0.0001\}$, as demonstrated in Figure 7a and Figure 7b.



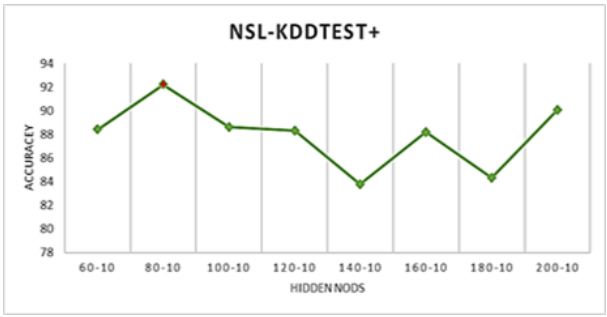
(a) Comparison Accuracy With Different Learning Rates Based On NSL_KDD Data Set.



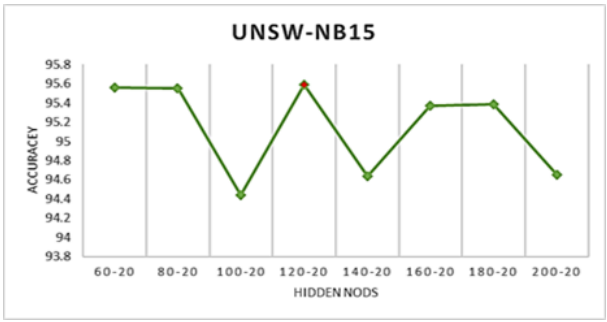
(b) Comparison Accuracy With Different Learning Rates Based On UNSW-NB15 Data Set.

Fig. 7. Comparison Accuracy With Different Learning Rates Based On NSL-KDD and UNSW-NB15 Data Set.

Experimental results indicate that an appropriate learning rate is 0.005 for the NSL-KDD dataset and 0.001 for the UNSW-NB15 dataset. The second critical hyperparameter is the number of neurons, evaluated as shown in Figure 8a and Figure 8b. Using too few neurons may lead to underfitting, as the network cannot adequately capture the underlying patterns. Conversely, employing too many neurons can result in overfitting, where the model memorizes the training data and fails to generalize. Thus, an optimal intermediate number of neurons is necessary to achieve effective training.



(a) Accuracy comparison with different numbers of hidden neurons on the NSL-KDD dataset.



(b) Comparison Accuracy With Different Hidden Neurons Based On UNSW-NB15 Data Set.

Fig. 8. Comparison Accuracy With Different Hidden Neurons Based On NSL-KDD and UNSW-NB15 Data Set.

6.4 Testing phase and Analysis of ANN Classifiers

In the testing stage, with the assistance of the fractal fuzzy membership degree, the predicted values of each sub-ANN$_i$ algorithm are aggregated according to the chosen aggregation method. The test sample x_j in each sub-ANN$_i$ classifier is represented as ANN$_i(x_j)$.

6.4.1. ANN Analysis in a Testing Stage for NSL-KDD Data Set

The NSL-KDD dataset was employed to evaluate the performance of the proposed system. Table 3a presents the confusion matrix multiclass results for the NSL-KDDTest-21 dataset during the testing phase.

TABLE 3. Confusion Matrix Results on the NSL-KDDTest-21 Dataset During the Testing Phase.

(a) Confusion Matrix Results For Five Classes For (NSL-Kddtest-21) In A Testing Phase.					
	Predicted Class				
Actual Class	**DoS**	**Normal**	**Probe**	**R2L**	**U2R**
DoS	3468	213	593	68	0
Normal	6	2131	15	0	0
Probe	226	60	2021	95	0
R2L	455	144	67	2088	0
U2R	0	131	4	0	65

(b) Binary Confusion Matrix Results Using (NSL-KDDtest-21) In A Testing Phase		
	Normal	**Attack**
Normal	TN (2131)	FP (21)
Attack	FN (548)	TP (9150)

Table 3b presents the binary confusion matrix results for the NSL-KDDTest-21 dataset derived from Table 3a. The detection accuracy was 82.47%, the detection rate(recall) was 82.47%, the system achieved a misclassification rate of 17.53%.

Table 4a shows the confusion matrix results for the `NSL-KDDTest+` dataset during the testing phase, the multi-class confusion matrix for both test datasets consists of five classes: one class representing normal transactions and four classes representing malicious detections (`DoS`, `Probe`, `R2L`, and `U2R`) attacks.

TABLE 4. **Confusion Matrix Results on the `NSL-KDDTest+` Dataset During the Testing Phase.**

(a) Five-Class Confusion Matrix					
Actual Class	**DoS**	**Normal**	**Probe**	**R2L**	**U2R**
DoS	6494	209	681	74	0
Normal	6	9688	16	1	0
Probe	166	56	2104	95	0
R2L	118	148	36	2452	0
U2R	0	143	0	0	57

(b) Binary Confusion Matrix		
Actual / Predicted	**Normal**	**Attack**
Normal	TN (9688)	FP (23)
Attack	FN (556)	TP (12277)

Table 4b presents the binary confusion matrix results for the `NSL-KDDTest+` dataset, derived from Table 4a. The overall accuracy detection was **92.24%**, the detection rate was **92.24%**, and the misclassification rate was **7.76%**.

6.4.2. ANN Analysis in a Testing Stage for UNSW-NB15 Data Set

The `UNSW-NB15` dataset was used to evaluate the performance of the proposed system. Table 5a shows the multi-class confusion matrix results for the `UNSW-NB15` dataset in the testing phase, using a confusion matrix with ten classes: one class representing normal transactions and nine classes representing malicious detections (`Generic`, `Exploits`, `Fuzzers`, `DoS`, `Reconnaissance`, `Analysis`, `Backdoor`, `Shellcode`, and `Worms`).

TABLE 5. presents the confusion matrix results for the UNSW-NB15 dataset during the testing phase. The model achieved a high number of true positives (9094) with zero false positives, demonstrating high detection accuracy and reliability.

(a) Predicted Class (Ten-Class Confusion Matrix)										
Actual	**Analysis**	**Backdoor**	**Dos**	**Exploits**	**Fuzzers**	**Generic**	**Recon.**	**Shellcode**	**Worms**	**Normal**
Analysis	**136**	0	0	0	0	0	19	0	0	0
Backdoor	0	**110**	1	0	0	0	3	0	0	0
Dos	0	6	**826**	0	0	0	1	0	0	0
Exploits	0	0	0	**2051**	0	0	0	163	0	9
Fuzzers	0	0	0	485	**710**	0	0	0	0	10
Generic	0	0	0	0	0	**3780**	0	0	0	7
Reconnaissance	0	0	0	22	0	0	**708**	0	0	0
Shellcode	0	0	0	0	0	0	0	**65**	0	4
Worms	1	0	0	1	0	0	1	0	**5**	0
Normal	0	0	0	0	0	0	0	0	0	**7349**

(b) Binary Confusion Matrix			
		Actual Class	
		Normal	**Attack**
Predicted	**Normal**	TN (7349)	FP (0)
	Attack	FN (23)	TP (9094)

Table 5b presents the binary confusion matrix results for the `UNSW-NB15` dataset, derived from Table 5a. The binary detection accuracy was **99.8%**, while the multiclass classification accuracy for the same dataset is **95.59%**. The overall accuracy detection was **99.8%**, the detection rate was **99.7%**, and the misclassification rate was **0%**. The difference in reported values is due to evaluation settings, where **95.59%** represents multiclass classification accuracy, while **99.8%** corresponds to binary detection accuracy derived from the confusion matrix.

6.5 Results Comparison

Table 6 presents the overall performance of the proposed approach, measured by *Accuracy*, *Precision*, *Recall*, and *F1-score* on three different test datasets. The experimental results on these datasets demonstrate the effectiveness and computational efficiency of the proposed approach. The outcome was compared with the results of five prominent classifiers: *K-Nearest Neighbor (KNN)*, *Random Forest (RF)*, *Support Vector Machine (SVM)*, *Decision Tree (DT)*, and *Artificial Neural Network (ANN)*. The comparison is made on the basis of the four evaluation metrics—*Accuracy*, *Precision*, *Recall*, and *F1-score*—with the results given in Tables Table 7, Table 8, and Table 9.

TABLE 6. Percentage of system's accuracy and time inference using three datasets.

Dataset	Accuracy	Precision	Recall	f1score	Time (sec)
NSL-KDDTest-21	82.47	83.44	82.47	82.95	0.1504
NSL-KDDTest+	92.24	92.68	92.24	92.46	0.3103
UNSW-NB15	95.59	96.68	95.59	96.13	0.4537

TABLE 7. Different AI comparisons using NSL-KDD Test-21

Model	DoS	Normal	Probe	R2L	U2R	Accuracy (%)	Precision	Recall	F1-score
KNN	71.2	67.6	64.2	11.7	4.0	54.21 ± 0.63	71.47	54.21	61.66
RF	59.4	88.1	61.9	6.7	0.5	51.94 ± 0.71	79.63	51.94	62.87
SVM	57.0	68.1	63.2	0	0	46.10 ± 0.58	50.64	46.10	48.27
DT	57.8	68.9	62.1	22.4	9.5	51.69 ± 0.66	65.06	51.69	57.61
ANN [120–10]	71.1	82.9	76.4	13.2	4.5	59.81 ± 0.74	75.92	59.81	66.91
Proposed system	**79.8**	**99.0**	**84.1**	**75.8**	**32.5**	$82.47 \pm 0.41^*$	**83.44**	**82.47**	**82.95**

***Statistical significance.** Over 10 independent runs, we compared FD-ANN against each baseline using a paired t-test ($p < 0.05$, normality verified via Shapiro–Wilk). Asterisks ($*$) indicates statistically significant improvements.

As seen from Table 7, the system proposed has high performance on the NSL-KDDTest-21 data, especially in the detection of low-frequency classes U2R and R2L with **32.5%** and **75.8%** detection rates, respectively. The system is even better than the other evaluated methods with an improved overall accuracy of **82.47%**.

TABLE 8. Different AI comparisons using NSL-KDD Test+ dataset

Model	DoS	Normal	Probe	R2L	U2R	Accuracy (%)	Precision	Recall	F1-score
KNN	82.9	92.4	64.5	11.7	4.0	75.68 ± 0.69	78.32	75.68	76.98
RF	75.8	97.3	62.2	6.7	0.5	74.56 ± 0.73	81.01	74.56	77.67
SVM	74.9	92.8	63.5	0.0	0.0	71.60 ± 0.62	65.70	71.60	68.52
DT	75.3	93.0	62.4	22.4	9.5	74.53 ± 0.71	75.38	74.53	74.95
ANN [120–10]	83.2	96.1	76.6	13.2	4.5	78.83 ± 0.65	81.03	78.83	79.92
Proposed system	**87.0**	**99.7**	**86.9**	**89.0**	**28.5**	$92.24 \pm 0.38^*$	**92.68**	**92.24**	**92.46**

***Statistical significance.** Over 10 independent runs, we compared FD-ANN against each baseline using a paired t-test ($p < 0.05$, normality verified via Shapiro–Wilk). Asterisks ($*$) indicates statistically significant improvements.

As evidenced by Table 8, the system proposed detects at high rates for the classes U2R and R2L within the NSL-KDDTest+ dataset at 28.5% and 89.0% accuracy, respectively. The system also achieves an overall accuracy of 92.24% that is higher than other methods considered.

TABLE 9. Performance comparison of different models on the UNSW-NB15 dataset

Class / Metric	KNN	RF	SVM	DT	ANN	Proposed FD-ANN
Analysis	16.1	1.9	0.0	1.2	0.0	**87.7**
Backdoors	14.0	7.8	0.0	21.0	0.0	**96.4**
DoS	23.2	19.4	0.0	26.5	25.3	**99.1**
Exploits	65.9	80.0	76.3	67.8	79.2	**92.2**
Fuzzers	55.7	60.3	59.8	59.8	59.8	**85.5**
Generic	96.1	96.2	97.2	96.7	96.1	**100.0**
Normal	85.1	94.6	85.1	91.1	87.9	**100.0**
Reconnaissance	66.0	83.5	36.7	81.0	88.0	**96.9**
Shellcode	23.1	62.3	0.0	62.3	30.4	**94.2**
Worms	0.0	0.0	0.0	12.5	0.0	**62.5**
Accuracy (%)	77.51 ± 0.82	84.59 ± 0.76	76.38 ± 0.69	81.79 ± 0.71	81.72 ± 0.74	$95.59 \pm 0.33^{*}$
Precision	82.54	86.20	75.02	85.91	84.60	**96.68**
Recall	77.51	84.59	76.38	81.79	81.72	**95.59**
F1-score	79.95	85.39	75.69	83.80	83.13	**96.13**

***Statistical significance.** Over 10 independent runs, we compared FD-ANN against each baseline using a paired t-test ($p < 0.05$, normality verified via Shapiro–Wilk). Asterisks ($*$) indicates statistically significant improvements.

As seen from Table 9, the system proposed shows an impressive detection ratio for the low frequency classes of attacks for the UNSW-NB15 dataset, such as *Analysis*, *Backdoors*, *Shellcode*, and *Worms*, with accuracies of **87.7%**, **96.4%**, **94.2%**, and **62.5%**, respectively. The system achieves a multiclass accuracy of **95.59%**, which is better compared to the other methods considered.

TABLE 10. Numerical ablation study of FD-ANN components on NSL-KDD and UNSW-NB15 datasets.

Dataset	Model Variant	FS	DPC	FFM	ANN	Accuracy (%)	F1-score
	Full FD-ANN(UNSW-NB15)					**95.59**	**96.13**
	w/o Feature Selection					92.30	93.00
	w/o DPC Clustering					93.10	93.80
	w/o FFM					94.20	94.50
	ANN only					89.70	90.20
	Full FD-ANN(NSL-KDDTest-21)					**82.47**	**82.95**
	w/o Feature Selection					78.60	79.10
	w/o DPC Clustering					79.40	80.00
	w/o FFM					80.80	81.20
	ANN only					74.90	75.30
	Full FD-ANN(NSL-KDDTest+)					**92.24**	**92.46**
	w/o Feature Selection					88.70	89.10
	w/o DPC Clustering					89.50	89.90
	w/o FFM					90.80	91.20
	ANN only					85.60	86.10

6.6 State of the art's comparison

To assess the contribution of each component in the proposed FD-ANN framework, a numerical ablation study was conducted on the NSL-KDD and UNSW-NB15 datasets, as reported in Table 10. Each core module—feature selection, IDPC clustering, fractal fuzzy membership (FFM), and ANN—was removed individually while keeping all other settings fixed. The full FD-ANN model consistently achieves the highest accuracy and F1-score, while the observed performance degradation upon removing any component confirms the necessity of the hybrid architecture.

To illustrate the efficiency of handling the issue of the imbalanced classes in multi-class problems, the proposed system FD-ANN (Fractal Density-Peak Clustering and Artificial Neural Network) enhances the rates of detection of the low-frequent classes like "R2L, U2R, Analysis, Backdoors, Worms, and Shellcode". We compare it with nine other intrusion detection models. In the datasets "NSL-KDDTest-21, NSL-KDDTest+, and UNSW-NB15", the system outcompetes others under Accuracy, Precision, Recall, F1-score, as well as the detection of the low-frequent classes. Baseline results (KNN, RF, SVM, DT, ANN) were re-implemented under the same experimental

conditions as our proposed system. State-of-the-art results are reproduced from the cited references and may reflect different dataset splits or evaluation settings. The results of comparison for the datasets are provided by Tables 11, **??**, and 12 respectively.

TABLE 11. Comparison of State-of-the-art models based on (NSL−KDDTest−21).

Author	Model	Train data	Test-21 data	Accuracy (%)	D.R of R2L	D.R of U2R
Y. Yang et al.(2020) [31]	SAVAER-DNN	125,973	11,850	80.30	77.88	58.33
Y. Li et al. (2020) [32]	Multi-CNN	125,973	11,850	64.81	35.15	23.50
Y. Yang et al. (2019) [9]	MDPCA-DBN	25,192	11,850	66.18	34.93	6.00
H. Wang et al. (2025) [33]	DPC-MDNN	125,973	11,850	91.8	55.1	26.5
P. Illy et al. (2019) [34]	Ensemble Model	25,192	11,850	78.33	N/A	N/A
S. M. Kasongo. (2023) [35]	RNN variants with XGBoost	125,973	11,850	N/A	N/A	N/A
M. Uçar et al. (2021) [36]	Stacking Ensemble (DT-LR-ANN + SVM)	125,973	11,850	84.32	N/A	N/A
C. M. Hsu et al. (2021) [37]	CNN-LSTM	125,973	11,850	99.98	N/A	N/A
Proposed system	**FD-ANN**	**25,192**	**11,850**	**82.47**	**75.8**	**32.5**

TABLE 12. Comparing State of the art's models based on (UNSW−NB15).

Author	Model	Train data	Test data	Accuracy (%)	D.R of Analysis	D.R of Backdoor	D.R of Shellcode	D.R of Worms
N. Moustafa et al. (2016) [42]	EM clustering	175,341	82,332	78.47	N/A	N/A	N/A	N/A
H. M. Anwer et al. (2018) [43]	filter ranking - FS-J48	175,341	82,332	88.30	N/A	N/A	N/A	N/A
Y. Yang et al. (2019) [9]	MDPCA-DBN	175,341	82,332	90.21	0.00	9.34	11.1	N/A
S. M. Kasongo et al. (2020) [44]	XGBoost+ANN	175,341	82,332	77.51	N/A	N/A	N/A	N/A
Ahmad et al. 2022 [45]	Adaboost	175,341	82,332	99.30	N/A	N/A	N/A	N/A
More et al. 2024 [46]	Random Forest	175,341	82,332	98.63	N/A	N/A	N/A	N/A
Wu et al.2025 [28]	TCG-IDS	175,341	82,332	91.48	N/A	N/A	N/A	N/A
Proposed system	**FD-ANN**	**35,069**	**16,466**	**95.59**	**87.7**	**96.4**	**94.2**	**62.5**

Results represented in Tables 11, **??**, and 12 indicate that the employed FD-ANN modeling architecture not only shows a high level of accuracy, but also offers a balanced quality of predicting diverse kinds of attacks as compared to state-of-the-art technical systems. Note that for some of the competing systems, their G-scores offer little or no benefit in that they vary only slightly with respect to the weak performance in R2L and U2R; see a number of studies which did not even present these rates. However, FD-ANN demonstrates improved and stable detection rates for low-frequency attacks across the NSL-KDD Test-21, NSL-KDD Test+, and UNSW-NB15 datasets, indicating its effectiveness in handling low-frequency, high-risk attacks. However, these results are specific to the benchmark datasets and are influenced by dataset characteristics such as feature distribution and class imbalance. Variations in data distribution, feature space, and attack patterns may affect performance in other settings. Therefore, the findings should be interpreted within this experimental context, and further validation on diverse, real-world datasets is necessary to confirm generalizability.

The advantage is not a result of employing more complex models or training with higher data volume. FD-ANN is less dependent on the training samples than deep or reinforcement learning models. Instead, it is inherent in the design of the system. In this work, density-based clustering is employed to retain the structural characteristics of micro-attacks, while a fractal fuzzy method is used to address irregular or low-intensity behaviors prior to classification. Consequently, the mode of operation from balanced samples and difficult data are learned together by the neural network, so that it generalizes better and performs more robustly.

Thus, FD-ANN not only improves the overall accuracy of intrusion detection systems but also has higher sensitivity for rare attacks, effectively uses the collected data, and achieves reliable operation, which are major challenging issues in most existing intrusion detection systems.

6.6.1. Practical Deployment and Scalability Analysis

In practice, the proposed FD-ANN scheme is suitable for real-time deployment under real-world conditions due to its modular architecture and balanced distribution of computational load across processing stages. Computationally intensive operations, including feature selection and Density Peak Clustering (DPC) aggregation, are executed offline during the training phase and therefore do not introduce overhead during online detection. Moreover, the computation of the Fractal Fuzzy Membership (FFM) relies on localized clustering structures rather than the entire data space, ensuring low computational complexity and scalability as network traffic volume increases. As reported in Table 6, the average inference time per experiment ranges from 0.15 s to 0.45 s, demonstrating low-latency detection. The ANN classifiers employ shallow architectures with two hidden layers ($[80, 10]$ for NSL-KDD and $[120, 20]$ for UNSW-NB15), resulting in a modest number of trainable parameters compared to deep learning models.

The experiments were all run on standard commodity hardware without GPU acceleration, demonstrating its viability for deployment in resource limited environments. Runtime detection process is mainly based on a single forward pass through the Artificial Neural Network (ANN), which is known to run with low latency and can be trained using commodity hardware or even GPU, leading towards near real-time response performance. Improving scalability to an extent, adaptive weight mechanism allows you to use smaller training samples so that every batch

of historical data never need to be completely re-trained fitting easily under modern network traffic setting! In addition, clear separation of training detection phases by design enable deployment across distributed and hybrid infrastructures including data-centers and Software-Defined Networks (SDN).
In addition to scalability, therefore, the FD-ANN framework balances detection performance and computational efficiency, allowing for real-world application of practical intrusion detection as opposed to experimentation in a laboratory environment.
In the longer term, the focus will be on full time complexity analysis, learning performance and run-time scalability.

7. CONCLUSION

The proposed system is crucial when an organization's resources are connected to the Internet, where the security goals (*Confidentiality, Integrity, and Availability (CIA)*) are required. Network attacks within an organization can result in severe degradation of network performance. The expected results of this study present a new proposed system based on hybrid techniques, representing a framework that combines multiple data mining methods capable of multi-classifying network attacks and providing a better detection rate for low-frequency attacks.
The performance of the proposed system also introduces a new metric for distributing testing data, which is essential to reduce both time and computational complexity. Several evaluation measures are used to assess the proposed system. The obtained results indicate that the best detection performance is achieved using hybrid techniques based on the IDPC clustering and ANN classification algorithm, enhanced by the fractal-based fuzzy membership function.
The overall accuracy for the `NSL-KDDTest-21` dataset is **82.47%**, with a precision of **83.44%**, recall (detection rate) of **82.47%**, and an F1-score of **82.95%**. For the `NSL-KDDTest+` dataset, the overall accuracy is **92.24%**, precision **92.68%**, recall (detection rate) **92.24%**, and F1-score **92.46%**. Meanwhile, for the `UNSW-NB15` dataset, The overall multiclass accuracy for UNSW-NB15 is **95.59%**, while the binary detection accuracy reaches **99.8%**. These results are obtained efficiently in terms of computation time (measured in seconds).
This study has five main limitations: (1) reliance on older datasets (NSL-KDD, UNSW-NB15) that may not reflect modern attacks; (2) lack of validation on real-world live networks; (3) no evaluation of adversarial robustness; (4) limited computational scalability analysis; and (5) no mechanism for concept drift adaptation.
In addition, the limitations and recommendations for future works in addition to achieving a low success rate of FD-ANN system proposed are discussed. Experimental results illustrated that the proposed framework provides a well performance on traditional benchmark datasets, however, if it can continuously maintain decent performance trend in more recent and realistic datasets remains uncertain especially with respect to the ever-evolving nature of contemporary network traffic along with the advent of novel attack strategies. Additionally, the proposed system was not rigorously tested against adversarial attacks, which is vital to ensure functionality in hostile environments. Furthermore, additional research may explore adaptive procedures for updating models in an online incremental fashion, finally resulting with a procedure of continuous learning to adapt and adjust themselves to concept drift on streaming data. As you can see these are the fundamental steps for increasing robustness and generalization in much more real world complicated elevation.

Conflicts of Interest

The authors declare no conflicts of interest.

Funding

No funding was received for this study.

Acknowledgment

The authors would like to express gratitude to the institution for their invaluable support throughout this research project.

REFERENCES

[1] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," *Journal of Computational Science*, vol. 25, pp. 152–160, 2018.
[2] S. Kostopoulos, D. Papatsaroucha, I. Kefaloukos, and E. K. Markakis, "eidps: A comprehensive comparative analysis of packet-level and

flow-level intrusion detection and prevention," in *2025 IEEE International Conference on Cyber Security and Resilience (CSR)*, 2025, pp. 372–378.
[3] U. B. Clinton, N. Hoque, and K. R. Singh, "Classification of ddos attack traffic on sdn network environment using deep learning," *Cybersecurity*, vol. 7, no. 1, p. 23, 2024, doi: https://doi.org/10.1186/s42400-024-00219-7.
[4] J. Lansky, S. Ali, M. Mohammadi, M. K. Majeed, S. H. T. Karim, S. Rashidi, M. Hosseinzadeh, and A. M. Rahmani, "Deep learning-based intrusion detection systems: A systematic review," *IEEE Access*, vol. 9, pp. 101 574–101 599, 2021, doi: https://doi.org/10.1109/ACCESS.2021.3097247.
[5] A.-A. Maiga, E. Ataro, and S. Githinji, "Intrusion detection with deep learning classifiers: A synergistic approach of probabilistic clustering and human expertise to reduce false alarms," *IEEE Access*, vol. 12, pp. 17 836–17 858, 2024.
[6] B. R. Kikissagbe and M. Adda, "Machine learning-based intrusion detection methods in iot systems: A comprehensive review," *Electronics*, vol. 13, no. 18, p. 3601, 2024, doi: https://doi.org/10.3390/electronics13183601, publisher = MDPI.
[7] D. Gowthami, S. E. Raja, and K. Srinivas, "Comprehensive analysis on machine learning and deep learning based intrusion detection in the internet of things," in *2025 5th International Conference on Mobile Networks and Wireless Communications (ICMNWC)*, 2025, pp. 1–7, doi: https://doi.org/10.1109/ICMNWC66779.2025.11354174.
[8] D. K. K. Reddy, J. Nayak, H. S. Behera *et al.*, "A systematic literature review on swarm intelligence based intrusion detection system: Past, present and future," *Archives of Computational Methods in Engineering*, vol. 31, no. 2, pp. 965–1002, 2024, doi: https://doi.org/10.1007/s11831-023-10059-2.
[9] 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," *Applied Sciences*, vol. 9, no. 2, p. 238, 2019.
[10] C. Tang, N. Luktarhan, and Y. Zhao, "Saae-dnn: Deep learning method on intrusion detection," *Symmetry*, vol. 12, no. 10, p. 1695, 2020, doi: https://doi.org/10.3390/sym12101695, publisher = MDPI.
[11] A. Husain and G. Dimitoglou, "Development of an efficient network intrusion detection model using extreme gradient boosting (xgboost) on the unsw-nb15 dataset," in *IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, 2019, pp. 1–6.
[12] 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. 75 729–75 740, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9437227
[13] W. El-Shafai, E.-S. El-Rabaie, F. E. Abd El-Samie, and S. A. Alshebeili, "Optimized convolutional neural network frameworks for automatic intrusion detection systems in wireless cybersecurity applications," *Wireless Personal Communications*, vol. 143, no. 1, pp. 307–342, 2025. [Online]. Available: https://doi.org/10.1007/s11277-024-11535-z
[14] J.-L. Lin, "Accelerating density peak clustering algorithm," *Symmetry*, vol. 11, no. 7, pp. 1–18, 2019, doi: https://doi.org/10.3390/sym11070859, articleno = 859, publisher = MDPI AG.
[15] B. Duan, L. Han, Z. Gou, Y. Yang, and S. Chen, "Clustering mixed data based on density peaks and stacked denoising autoencoders," *Symmetry*, vol. 11, no. 2, pp. 1–22, 2019, doi: https://doi.org/10.3390/sym11020163, articleno = 163, publisher = MDPI AG, month = feb,.
[16] Z. Jiang, X. Liu, and M. Sun, "A density peak clustering algorithm based on the K-Nearest Shannon entropy and Tissue-Like P system," *Mathematical Problems in Engineering*, vol. 2019, pp. 1–13, 2019.
[17] C. Ren, C. Wang, Y. Yu, W. Yang, and R. Guo, "Network intrusion detection based on relative mutual k-nearest neighbor density peak clustering," *Frontiers in Physics*, vol. 13, 2025.
[18] X. Sun, X. Liu, C. Deng, H. Chu, G. Wang, and H. Zhao, "An enhanced density peak clustering algorithm with dimensionality reduction and relative density normalization for high-dimensional duplicate data," *IEEE Access*, vol. 13, pp. 147 242–147 264, 2025, doi: https://doi.org/10.1109/ACCESS.2025.3596983.
[19] F. Foukalas, "A survey of artificial neural network computing systems," *Cognitive Computation*, vol. 17, no. 1, 2025, doi: https://doi.org/10.1007/s12559-024-10383-0.
[20] N. Oliveira, I. Praça, E. Maia, and O. Sousa, "Intelligent cyber attack detection and classification for network-based intrusion detection systems," *Applied Sciences*, vol. 11, no. 4, p. 1674, 2021.
[21] A. Drewek-Ossowicka, M. Pietrołaj, and J. Rumiński, "A survey of neural networks usage for intrusion detection systems," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 1, pp. 497–514, 2021.
[22] S. J. Mousavirad, G. Schaefer, K. Rezaee, D. Oliva, D. Zabihzadeh, R. K. Chakrabortty, H. Mohammadigheymasi, and M. Pedram, "A novel metaheuristic population algorithm for optimising the connection weights of neural networks," *Evolving Systems*, vol. 16, no. 1, p. 18, 2025, doi: https://doi.org/10.1007/s12530-024-09641-1, publisher = Springer.
[23] Z. Maseer *et al.*, "Meta-analysis and systematic review for anomaly network intrusion detection systems: Detection methods, dataset, validation methodology, and challenges," *IET Networks*, 2024, doi: https://doi.org/10.1049/ntw2.12128.
[24] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*. Canberra, ACT, Australia: IEEE, November 2015, pp. 1–6.
[25] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*. Ottawa, ON, Canada: IEEE, 2009, pp. 1–6, doi: https://doi.org/10.1109/CISDA.2009.5356528.
[26] N. Moustafa, "Designing an online and reliable statistical anomaly detection framework for dealing with large high-speed network traffic," PhD Thesis, University of New South Wales (UNSW), Canberra, Australia, 2017, doi: https://doi.org/10.26190/unsworks/3298, note = Introduces the UNSW-NB15 dataset. [Online]. Available: http://hdl.handle.net/1959.4/58748
[27] C. Strickland, M. Zakar, C. Saha, S. Soltani Nejad, N. Tasnim, D. J. Lizotte, and A. Haque, "Drl-gan: A hybrid approach for binary and multiclass network intrusion detection," *Sensors*, vol. 24, apr 2024.
[28] C. Wu, J. Sun, J. Chen, M. Alazab, Y. Liu, and Y. Xiang, "Tcg-ids: Robust network intrusion detection via temporal contrastive graph learning," *IEEE Transactions on Information Forensics and Security*, vol. 20, p. 1475, jan 2025, doi: https://doi.org/10.1109/TIFS.2025.3530702.
[29] K. M. Sujon, R. Hassan, K. Choi, and M. A. Samad, "Accuracy, precision, recall, f1-score, or mcc? empirical evidence from advanced statistics, ml, and xai for evaluating business predictive models," *Journal of Big Data*, vol. 12, no. 1, p. 268, 2025, doi: https://doi.org/10.1186/s40537-025-01313-4.
[30] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
[31] Y. Yang, K. Zheng, B. Wu, Y. Yang, and X. Wang, "Network intrusion detection based on supervised adversarial variational auto-encoder with regularization," *IEEE Access*, vol. 8, pp. 42 169–42 184, 2020.
[32] Y. Li, Y. Xu, Z. Liu, H. Hou, Y. Zheng, Y. Xin, Y. Zhao, and L. Cui, "Robust detection for network intrusion of industrial iot based on

multi-cnn fusion," *Measurement*, vol. 154, p. 107450, 2020. [Online]. Available: https://doi.org/10.1016/j.measurement.2019.107450

[33] H. Wang, J. Zhang, Y. Shen, S. Wang, B. Deng, and W. Zhao, "Improved density peak clustering with a flexible manifold distance and natural nearest neighbors for network intrusion detection," *Scientific Reports*, vol. 15, no. 1, p. 8510, 2025. [Online]. Available: https://www.nature.com/articles/s41598-025-92509-4

[34] P. Illy, G. Kaddoum, C. M. Moreira, K. Kaur, and S. Garg, "Securing fog-to-things environment using intrusion detection system based on ensemble learning," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2019, pp. 1–7, also available as arXiv preprint arXiv:1901.10933. [Online]. Available: https://doi.org/10.1109/WCNC.2019.8885534

[35] S. M. Kasongo, "A deep learning technique for intrusion detection system using a recurrent neural networks based framework," *Computer Communications*, vol. 199, pp. 113–125, 2023.

[36] M. Uçar, M. O. İncetaş, and E. Uçar, "A stacking ensemble learning approach for intrusion detection system," *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, vol. 9, no. 4, pp. 1329–1341, 2021. [Online]. Available: https://doi.org/10.29130/dubited.737211

[37] C. M. Hsu, M. Z. Azhari, H.-Y. Hsieh, S. W. Prakosa, and J.-S. Leu, "Robust network intrusion detection scheme using long-short term memory based convolutional neural networks," *Mobile Networks and Applications*, vol. 26, no. 3, pp. 1137–1144, 2021, tested on NSL-KDD dataset with KDDTest+ and KDDTest-21.

[38] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21 954–21 961, 2017.

[39] I. Ahmad, I. T. Al-Shaikhli, H. M. Dweik, A. Shayea, and H. Ahmed, "A convolutional neural network for improved anomaly-based network intrusion detection," *Sensors*, vol. 21, no. 12, p. 4145, 2021, doi: https://doi.org/10.1089/big.2020.0263, publisher = MDPI.

[40] Y. Imrana, Y. Xiang, L. Ali, Z. Abdul-Rauf, Y.-C. Hu, S. Kadry, and S. Lim, "χ^2-bidlstm: A feature driven intrusion detection system based on χ^2 statistical model and bidirectional lstm," *Sensors*, vol. 22, no. 5, p. 2018, 2022. [Online]. Available: https://doi.org/10.3390/s22052018

[41] M. Vishwakarma and N. Kesswani, "A new two-phase intrusion detection system with naïve bayes machine learning for data classification and elliptic envelop method for anomaly detection," *Decision Analytics Journal*, vol. 7, p. 100233, 2023.

[42] 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," *Information Security Journal: A Global Perspective*, vol. 25, no. 1-3, pp. 18–31, 2016. [Online]. Available: https://dl.acm.org/doi/10.1080/19393555.2015.1125974

[43] H. M. Anwer, M. Farouk, and A. Abdel-Hamid, "A framework for efficient network anomaly intrusion detection with features selection," in *2018 9th International Conference on Information and Communication Systems (ICICS)*. IEEE, 2018, pp. 157–162.

[44] S. M. Kasongo and Y. Sun, "Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset," *Journal of Big Data*, vol. 7, no. 1, p. 105, 2020.

[45] I. Ahmad, Q. E. Ul Haq, M. Imran, M. O. Alassafi, and R. A. AlGhamdi, "An efficient network intrusion detection and classification system," *Mathematics*, vol. 10, p. 530, feb 2022, the proposed method is an AdaBoost-based approach for network intrusion detection using selected features, achieving an accuracy of 99.3.

[46] S. More, M. Idrissi, H. Mahmoud, and A. T. Asyhari, "Enhanced intrusion detection systems performance with unsw-nb15 data analysis," *Algorithms*, vol. 17, feb 2024.