Journal of Engineering Science and Technology
Vol. 18, No. 5 (2023) 2411 - 2431

ENHANCING INTRUSION DETECTION SYSTEM PERFORMANCE AGAINST LOW FREQUENT ATTACKS USING FC-ANN ALGORITHM

DINA A. SALIH[1], YASIR A. MOHAMED[2,*], MOHAMED BASHIR[3]

[1]Faculty of Mathematical and Computer Sciences, University of Gezira, Medani, Sudan
[2]Collage of Business Administration, A'Sharqyiah University, Ibra, Oman
[3]Collage of Business Administration, A'Sharqyiah University, Ibra, Oman
*Corresponding author: Yasir.abdulgadir@asu.edu.om

Abstract

The significance of network security within the contemporary corporate landscape cannot be overemphasized. There have been other instances whereby hackers and invaders have made concerted efforts, resulting in successful disruptions of extensive corporate networks and online services. The primary objective of intrusion detection is to discern and recognize malevolent actions inside a networked computer system, with the intention of promptly halting and maybe mitigating such activities. There is a need to investigate novel detection methods in order to improve detection accuracy, since low-frequency attacks often use stealthy and sophisticated strategies that may elude existing IDSs that depend on signature-based detection approaches. The fuzzy clustering-artificial neural network has been proposed by a number of researchers as a means by which intrusion detection systems can outperform more conventional approaches. To that end, researchers have developed a revolutionary hybrid Artificial Neural Network approach that enhances the accuracy, precision, and resilience of intrusion detection systems by combining fuzzy clustering with neural networks. Fuzzy clustering-artificial neural network-modified is a new method that we developed to optimize training time and improve detection precision for low-frequency attacks. This method still has room for improvement, however, especially in terms of the time required to train the entire system and the accuracy of the low-frequency attacks. The proposed technique, a tweaked variation of the Fuzzy Clustering Artificial Neural Network, outperforms the original algorithm in detecting rare assaults by a margin of 39.4 percent points, and it slashes the estimated training time by 99.7 percent points, as shown by the experiments. The effectiveness of the method has been shown in recognizing infrequent attacks that occur sporadically and do not have a well-defined pattern. Furthermore, it is essential to consider the appropriateness of utilization in high-speed networks characterized by substantial traffic quantities. Additionally, it is crucial to acknowledge the adaptability of the system to emerging and changing threats that may not have been encountered before.

Keywords: Fuzzy clustering, FC-ANN algorithm, Intrusion detection system, low frequent attacks.

1. Introduction

Data exchange across the network must be safe for web-based services like shopping on the internet, marketplaces, and banking. To avoid misuse or theft, access to this information must be severely restricted. Antivirus software, intrusion detection systems, and firewalls all serve the same goal. Intrusion detection systems help fix computer network vulnerabilities. Network process data analysis is used to identify cyberattacks. Important metrics [1] for intrusion detection systems are their accuracy and consistency (IDS). The reliability and precision of detectors have been greatly enhanced by several studies. Expert rule systems and preliminary statistical analysis are the primary foci of early-stage study. Expert systems that rely on rules and statistical methods both struggle with larger datasets. There are a plethora of data mining techniques aimed at resolving this issue.

There were a number of techniques used to identify intrusion in computer systems before the development of AI. Signature-based detection and anomaly-based detection are two major categories that describe these techniques. By using signature-based detection, incoming network traffic is compared against a database of known threat signatures. This technique works well for identifying common threats but has little success with novel or unexpected threats. Nevertheless, anomaly-based detection looks for outliers in the system's usual behavior. This technique can effectively identify previously unseen threats, but it takes a large quantity of data to adequately determine the system's baseline behavior [2]. Among the most extensively used algorithms related to machine learning, artificial neural networks (ANNs) that categorize data when it is too complicated to design by other traditional methods. Instead of classifying, the neural network prioritizes class-specific characteristics [3].

Intrusion detection systems are proactive tools that alert system administrators of possible security breaches rather than actively preventing invasions. (IDSs) may be categorized as either host-based or network-based and can operate in an online or offline manner. Furthermore, the assertion pertains to the abuse or aberrations. Host-based intrusion detection systems use data from computer log files, while network-based intrusion detection systems collect and analyse data packets. Offline intrusion detection systems analyses data after the fact and raise a warning if a security violation occurs after the last intrusion detection check. IDS that detect anomalies and abuse detect malicious activity [4].

When the frequency of attacks decreases, artificial neural networks (ANN) exhibit a diminished performance. assaults characterized by low frequency exhibit a limited sample size for analysis. The acquisition of knowledge pertaining to these attacks poses a challenge for artificial neural networks (ANNs), resulting in a decrease in the accuracy of detection. The significance of less frequent acts of aggression should not be underestimated. Nevertheless, these attacks will have catastrophic consequences [5]. In the event that an assailant successfully executes a User-to-Root (U2R) attack, they will get complete administrative privileges over the compromised computer or network device. Infrastructural defence systems (IDS) seldom encounter infrequent or atypical assault patterns. The occurrence of convergence towards a local minimum [6] results in the instability of artificial neural networks. Despite advancements in accuracy and stability, many Intrusion Detection Systems (IDS) continue to face challenges in effectively detecting low-frequency assaults [7].

2. Background

When it comes to intrusion detection systems, evolving malware poses a significant difficulty. There has been an increase in the complexity of malicious assaults, and the main difficulty is in detecting new and complex forms of malware. To evade detection by IDS, malware developers use a number of ways for hiding their true intentions. There have been more zero-day attacks against internet users [8]. Computer security has emerged as a top priority [9] as our reliance on electronic devices grows. An incursion is any action carried out without authority that harms a computer system [10].

An intrusion detection system (IDS), which may be implemented as either software or hardware, detects the presence of a network penetration attempt. Intrusion Detection Systems (IDS) are designed to identify and mitigate security threats that conventional firewalls may fail to detect. The establishment of a robust degree of security against assaults that pose risks to the dependability, legitimacy, or confidentiality of computer systems is an important measure [11].

The confusion matrix for a two-class classifier, used to assess IDS, is shown in Table 1. Classes in the matrix's rows are those that have actually been seen, while those in the matrix's columns are those that have been predicted. Classification Rate (CR), Accuracy, False Negative Rate (FNR), and True Positive Rate (TPR) are some of the metrics used to evaluate IDS performance [12].

Table 1. Confusion matrix for IDS system (updated from [7]).

Actual Class	**Predicted Class**	
Class	Normal	Attack
Normal	True Negative(TN)	False Positive(FP)
Attack	False Negative(FN)	True Positive(TP)

3. Artificial Neural Network (ANN)

Among the most promising machine learning methods, ANN has shown effective in identifying several types of malicious software. There is, nevertheless, potential for development in the accuracy and precision with which ANN-based IDS detects attacks, particularly less common ones. When dealing with fewer prevalent assaults, the training sample is smaller, making it harder for the ANN to correctly identify their properties. As a result, attacks that happen less often will have a lower chance of being discovered. In terms of protecting sensitive digital information [13].

To divide the training set into manageable chunks, we employ Fuzzy Clustering, an unsupervised technique. In order to train ANNs, these subsets are necessary. When the amount of the data is reduced, the training time required to train each ANN is also decreased. The detection rate improves when the results from many ANNs are aggregated by a single aggregating ANN, which then corrects the misclassifications produced by the individual ANNs. As a result, we aim to accelerate attack detection while simultaneously decreasing training time [14].

4. Related Work

An intrusion detection system for the cloud that makes use of fuzzy clustering and neural networks is described in [15]. The system restore function allows users to revert the cloud server to a previous state, including all registry keys, system files,

storing and advocating, and the project database, in the event that the server fails, or an intrusion is discovered. In today's cloud computing settings, the system is used to perform a range of security-related activities. While network-based intrusion detection systems can be effective tools for detecting and preventing cyberattacks, they are not the only solution. If the article focuses too narrowly on network-based solutions, it may be oversimplifying the issue.

According to Liu et al. [16], computer security research has focused on huge data security. The security of massive data sets is growing faster than the processing capacity of a single node. The use of distributed computing boosts speed and accuracy. An IDS terminal administration server, a node, an IDS server, a monitoring server, and Hadoop master servers all make up a cloud-based intrusion detection system. Based on the results of the experiments, it seems that the Hadoop-based intrusion detection system can identify threats more effectively. Prior to neural network training, this research explores ideal weights. Cloud-based genetic and neural network techniques are made possible by the Hadoop distributed computing framework. The algorithm is being upgraded to identify intrusions more effectively. Based on these results, it seems that intrusion detection can defend application systems against intrusion threats, although it may not be very useful against infrequent attacks.

Zhang and Xu [17] provided a new technique for intrusion detection that makes use of a Convolutional Neural Network (CNN) and feature selection. The approach uses a combination of traffic flow features and selected time-domain features as inputs to the CNN. Time-domain characteristics are produced from traffic time series data, whereas flow features are taken from the network packet header. An intrusion detection benchmark termed KDDCup99 is used to test the suggested model. The findings demonstrate that the CNN-based model provides superior accuracy, precision, and recall compared to the conventional machine learning techniques. By lowering the dimensionality of the input data and deleting unimportant features, the feature selection strategy enhances the model's performance. Despite having a significant effect, and Although feature selection may assist reduce data dimensionality and boost model performance, it also runs the risk of leaving out crucial information. It may throw off the intrusion detection model's precision.

Brown et al. [18] reported that the most common way for avoiding network intrusion is the creation of attack detection systems based on machine learning and data mining techniques. These technologies improve network security by identifying malicious activity and blocking it. This study compares and contrasts the efficacy of fuzzy logic-based and neural network-based intrusion detection systems using the well-known KDD 99 benchmark dataset. In addition to its overall superiority, the TSK-based intrusion detection system also boasted the best detection performance across all normal, DoS, and probing classes. The issue is that attackers might try to elude detection by changing data in ways that are difficult for machine learning algorithms to spot. To the point where it becomes useless against certain threats.

An intrusion detection system based on weighted K-means clustering and an artificial neural network (WKMC + ANN) is presented in [19]. The article discusses both cluster analysis and intrusion detection. The clustering module uses WKMC to create distinct subsets of the input data. The intrusion detection module

retains the data cluster structure once it has been trained using ANN. The task of testing requires the selection of an ANN classifier, using distance or similarity metrics, that most closely matches the cluster to which the data in question most closely belongs. The experimental assessment was done on the benchmark database, and the results showed that the suggested approach was superior to the state-of-the-art approaches in terms of accuracy. The computational complexity of ANNs and clustering techniques can make it challenging to deploy the intrusion detection system in real-time environments or on resource-constrained devices.

Using a divide-and-conquer approach, Ashfaq et al. [20] divided unlabelled samples and their classification results into subsets according to the level of fuzziness. A new semi-supervised learning (SSL) approach was developed to enhance classifier performance on ID datasets. Neural networks with random weights (NNR) were chosen as the foundational classifier because to their high learning performance at a relatively cheap computing cost. The "hidden nodes" of NNR have their parameters, or weights and biases, chosen at random and independently. On the other hand, the number of sub-problems may grow too enormous to handle successfully in large or complicated networks, making the divide-and-conquer strategy less effective. Moreover, as the size and complexity of the network grow, the overhead associated with coordination between the sub-problems may also grow.

For IoT networks, Aljumah [21] explained an Intrusion Detection System (IDS) built on Convolutional Neural Networks (CNNs). The proposed system uses a CNN-based classifier to differentiate between normal and malicious traffic in IoT networks. The CNN classifier is trained on a dataset of network traffic data that includes both benign and attack traffic. Results from experiments show that the suggested IDS is very effective in detecting various assaults on IoT networks, including Denial of Service (DoS) and Command and Control (C&C) attacks. The authors suggest that the proposed CNN-based IDS can be used as a complementary approach to existing rule-based and signature-based IDS in IoT networks. One limitation of the proposed CNN-based IDS is that it may be vulnerable to adversarial attacks, where attackers can modify the network traffic to evade detection by the CNN-based classifier. Future research could explore ways to address these limitations, such as developing more efficient algorithms for data labelling or incorporating techniques to improve the robustness of the CNN classifier against adversarial attacks.

An overview of various Intrusion Detection Systems (IDS) that have been proposed for Wireless Sensor Networks (WSNs) provided by Butun et al. [22]. TWSNs have several uses, including as in the fields of environment and health care monitoring and control. Nevertheless, owing to intrinsic features like as restricted resources and the scattered structure of the nodes, they are subject to a variety of security risks. The authors review several existing IDS approaches for WSNs, including rule-based, anomaly-based, and hybrid-based methods. The article also discusses some of the key challenges in developing effective IDS for WSNs, such as resource constraints, network topology, and the need for real-time processing. The authors conclude that no single IDS approach is suitable for all WSN applications, and that the choice of IDS should depend on the specific application requirements and security threats.

A multivariate statistical network monitoring strategy for anomaly identification is presented in [23], using Principal Component Analysis (PCA). The aim of the proposed approach is to identify anomalies in network traffic by analysing multiple variables, such as packet size, inter-packet arrival time, and protocol type, using PCA. The authors also use the Mahalanobis distance metric to measure the distance between the observed network traffic and the PCA model. The experimental findings show that the suggested method can reliably identify DDoS assaults, port scans, and worms, among other sorts of network abnormalities. One limitation of the proposed PCA-based approach is that it assumes that the network traffic follows a multivariate normal distribution. This assumption may not hold in some cases, such as when the network traffic contains highly skewed or non-Gaussian distributions. In addition, the PCA-based approach may not be suitable for detecting anomalies in dynamic networks where the traffic patterns change rapidly over time.

To assist in notifying the information security community of new vulnerabilities and to serve as evidence in legal proceedings, Wang et al. [24] introduced a novel intrusion detection technique based on ANN and fuzzy clustering. They dubbed it FC-ANN. The variety of the training set may be divided into multiple groups with the help of the fuzzy clustering technique. As a result, detection performance is enhanced due to the reduced complexity of each sub-training sets. Using the KDD CUP 1999 dataset, they demonstrate that their novel technique is very accurate and stable for identifying low-frequency assaults (such R2L and U2R attacks). The proposed approach does not explicitly model the uncertainty and imprecision in the network traffic data, which can affect the accuracy of the classification results.

Less frequent assaults need less data for learning than more common ones. While Artificial Neural Networks (ANNs) may be trained to recognize cyberattacks, their accuracy is limited by the difficulty of training them to detect the unique features of each attack. To address those issues, Elhag et al. proposed Fuzzy Clustering (FC-ANN), a new ANN-based intrusion detection approach that improves detection accuracy and stability for unusual attacks [25]. One limitation of the proposed approach is that it relies on a set of predefined rules to classify network traffic. These rules may not be suitable for detecting novel attacks or attacks with unknown patterns.

Vinayakumar et al. [26] proposed a deep learning-based approach for detecting malware in computer systems. The authors use a Convolutional Neural Network (CNN) to analyse the behavior of programs and classify them as benign or malicious. The CNN model takes as input the system call sequences generated by running the programs and learns to detect patterns indicative of malware. The authors also propose a technique for generating adversarial examples to evaluate the robustness of the CNN model to malicious attacks. The experimental results demonstrate that the proposed approach achieves high accuracy in detecting malware while also being robust to adversarial attacks. A limitation of the proposed approach is that it relies on system call sequences as the primary source of input for the CNN model. Consequently, this affects the model's ability to detect certain types of malware that do not exhibit anomalous system call patterns. Additionally, the authors did not evaluate the proposed approach using a large-scale dataset, which may limit the ability to assess the scalability and generalizability of the proposed approach.

Li et al. [27] conducted a comprehensive examination of machine learning (ML) techniques used in the detection of network intrusions. The study included an extensive array of machine learning techniques and their applications in the field of network intrusion detection. These techniques included Decision Trees, Support Vector Machines, Naive Bayes, Random Forests, and Artificial Neural Networks (ANNs).Accuracy, performance, and scalability are just some of the topics addressed in this comprehensive overview of supervised and unsupervised learning methods. The authors also emphasize the necessity of feature selection and data pre-processing to increase the machine learning-based intrusion detection systems' efficiency. The article does not include a full analysis of the available machine learning methods, which is one of its drawbacks. The survey only provides a high-level overview of the different techniques, without comparing their relative performance in detail.

The approach described by Hamamoto et al. [28] has the objective of identifying abnormalities in network traffic via the analysis of network flows and the extraction of pertinent information.. Genetic algorithm has been used to optimize the feature selection process, and a fuzzy logic system to classify the network flows as normal or abnormal. The experimental results demonstrate that the proposed approach achieves high accuracy in detecting network anomalies while also being robust to noise and variations in network traffic. The proposed approach relies on handcrafted features, which may limit the ability of the system to detect complex and dynamic network anomalies.

The purpose of Genetic Algorithm and Fuzzy Logic combined to discover abnormalities in a network was outlined by Samrin and Vasumathi [29], Foreseeing how network traffic will behave over a certain time period is the goal of Flow Analysis, which is implemented by applying the Genetic Algorithm to generate a Digital Signature of a Network Segment. Furthermore, a Fuzzy Logic strategy is employed to decide whether or not an event is anomalous, as opposed to other approaches found in the literature. Indeed, an expert system is introduced that can monitor network traffic using IP flows, while at the same time creating projected behaviors at regular time intervals, alerting when a possible problem is present. The authors did not compare their approach to other state-of-the-art intrusion detection techniques, such as deep learning-based approaches. This may limit the ability to assess the effectiveness of the proposed approach in comparison to other methods.

Gao et al. [30] proposed a method to improve classifier performance using less data and a refined version of the dynamic fuzzy C means algorithm. The KDD Cup 99 malware dataset was used. The KDD 99 training and test datasets were normalized before the recommended method was used. When c is fuzzy, it means the effectiveness of the clustering technique has increased. Performance of the proposed strategy is measured against that of competing strategies. Experiments show that the proposed method yields high detection rates and high accuracy. The Java packages of WEKA, a popular open-source machine learning and data mining tool, include things like associations, classifiers, clusters, and more. The suggested method may be limited in its applicability to various network settings since it is based on a narrow use case (cloud-based robotic systems).

It's worth noting that this work is generally considered to be a continuation of the studies found in [31]. Additional details may be found in [32-35].

5.Methods

Initially, a thorough exposition of the novel methodology is presented. Subsequently, a detailed explanation is presented about the fundamental constituents, namely the module for artificial neural network, the module for fuzzy aggregation, and the module for fuzzy clustering. The fundamental framework of a research methodology is shown in Fig. 1.

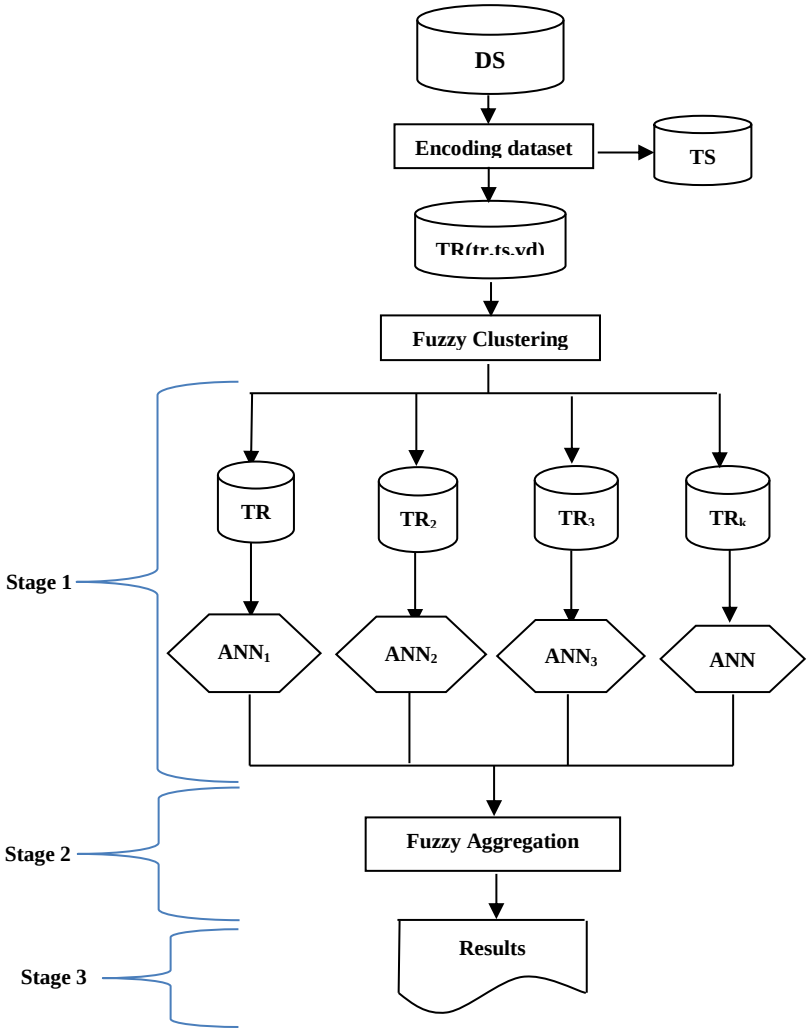

Fig. 1. Research methodology.

Like other machine learning frameworks, FC-ANN features a training and a testing phase. Mostly, we may divide machine learning process into three eras: Before anything else, a DS is converted into a training set. When used with training set, the fuzzy clustering module may provide a wide variety of test data sets.

During the training and clustering step, we took in the data set and cleaned it up. The first phase, fuzzy clustering, will begin shortly. While analysing the data

in this way, patterns of different connectivity densities and kinds emerge as distinct subgroups. Furthermore, the subset is divided into three groups at different rates: the training set accounts for 70% of the training, the test set for 15%, and the validation set for 15%. Figure 2 illustrates the fuzzy clustering module.

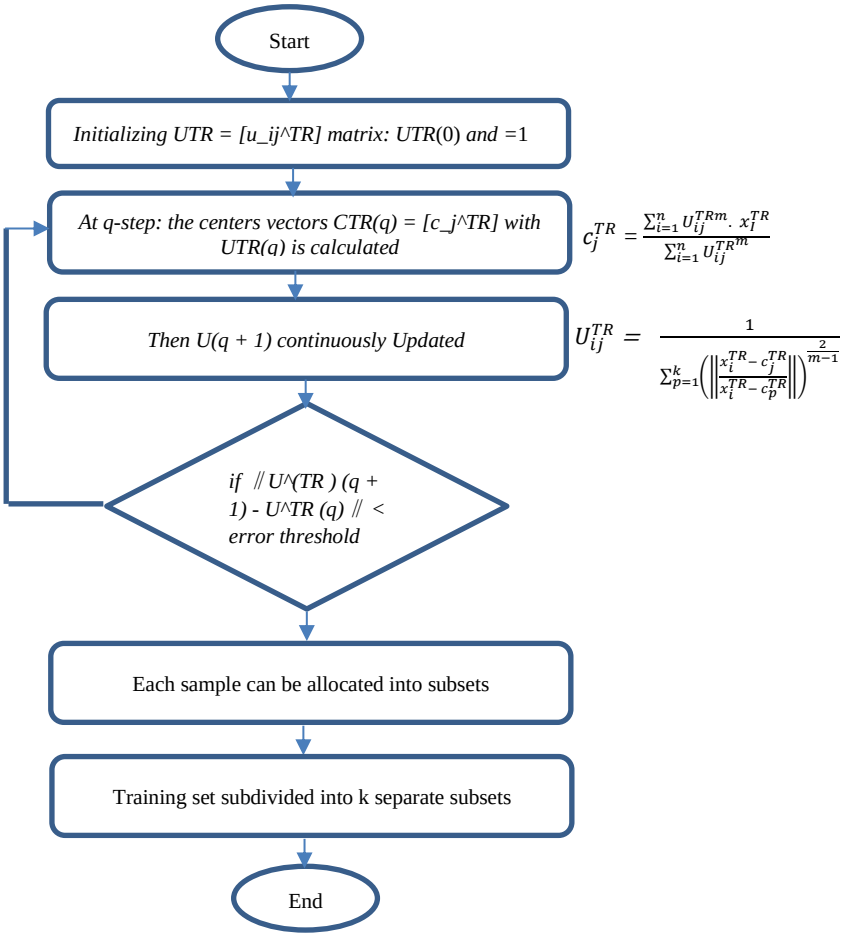

Fig. 2. Fuzzy clustering module.

Several ANN models are trained using different learning algorithms and data. Towards this end, a conventional feed-forward neural networks that have been trained by back-propagation will be used. This is a high-level description of the proposed algorithm:

A. *Establish an ANN with a number of input nodes equal to the number of features in the dataset.*

B. *Set the weights to a few random digits.*

C. The input is sent across the network for each sample used in training.

(1) *The weighted sum of inputs and bias is distributed to each hidden node:*

$z = b + \Sigma_i w_i x_i$

where z is the total input to the hidden node, b is the bias term

a. A non-linear activation function is then applied to this. A unipolar sigmoid activation function (*logsig, output* [0, 1]) *is utilized:*

$f(z) = 1 / (1 + exp(-z))$

(2) *With the FC-ANN method, the output of the ANN layers is transformed using the Tan-Sigmoid function (tansig, output (+1,-1)) rather than the Purelin function:*

$f(z) = (2 / (1 + exp(-2z))) - 1$

where z is the total input to the neuron, computed as the weighted sum of the inputs plus a bias term.

(3) *Thereafter, the ANN's computed output is contrasted to the goal value, and the error is determined by applying the error function, which is:*

$E = 1/2 * \Sigma_i(y_i - \hat{y}_i)^2$

where E is the error, y_i is the true output for the i^{th} example, and $\hat{y}_i$ is the predicted output for the i^{th} example.

(4) *The ANN then updates its weights based on this phrase once the error has been propagated back through the system, that incorporates two* steps:

(a) Computing the error at the output layer:

$\delta^{(n)} = \nabla aE \odot f'(z^{(n)})$

where $\delta^{(n)}$ represents the error at the output layer, ∇aE represents the gradient of the error with respect to the output, $f'(z^{(n)})$represents the derivative of the activation function at the output layer, and $\odot$ signifies element-wise multiplication.

(b) Propagating the error backwards to the hidden layers:

$\delta^{(l-1)} = (w^{(l)})^T\delta^{(l+1)} \odot f'(z^{(l-1)})$

$\Delta w^{(l)} = -\eta\delta^{(l+1)}(a^{(l-1)})^T$

where $\delta^{(l-1)}$ is the error at the (l-1)-th hidden layer, $(w^{(l)})^T$ is the transpose of the weight matrix between layers l and (l-1), η is the learning rate, $a^{(l-1)}$ is the output of the (l-1)-th layer, and $\Delta w^{(l)}$ is the update to the weight matrix between layers l and (l-1).

b. To speed up the learning process, the velocity parameter (0, 1) *is used.*

If the error E_m exceeds the predefined level, training will be halted. Otherwise, return to the third step. To complete the feed-forward neural networks training process using the back-propagation approach, each ANN*i* may utilize a different subset of TR *K*. Nevertheless, the next challenge is figuring out how to pool data from different ANN*i* base models.

By simulating the ANN*i* on the full TR, we may find the optimal parameters for the ANN*i* to reduce its error. The data is then compiled using the membership grades determined by the fuzzy clustering component. Next, we train a new ANN using the combined data. Final results may be generated using the last remaining fuzzy aggregation module. To see how we teach a new ANN to see and fix errors

in its predictions, it is assumed that every trained ANN_i will use the whole training set *TR* as input data:

$y = f(W_2 * f(W_1 * x + b_1) + b_2)$

where x is the input vector, W_1 and W_2 are weight matrices, b_1 and b_2 are bias vectors, and $f()$ is the activation function used by the network. To train the next ANN, Y_{input} can be used as input and use the whole training set TR's class label as output to train the new ANN.

6. Implementation

Tests have been conducted using the KDD CUP 1999 dataset. The MIT Lincoln Laboratory creates and manages the KDD CUP 1999 dataset, which is based on the original DARPA software for evaluating intrusion detection from 1998. The KDD set, containing approximately 4,898,431 connections, is available for download from the website http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html; the KDDCUP dataset includes five different types of connections, the characteristics of which are summarized in Table 2. Matlab 2017a was used to implement and test the suggested approaches.

Each connection has its own unique set of 41 characteristics, which are used to determine the safe or risky nature of the records kept on that connection. Variables in these characteristics can be either continuous or discrete or even symbolic. The 41 characteristics of each link are broken down in Table 3.

Many uses have made use of random selection to cut down on the size of the dataset. We chose R2L and U2R attacks because they represent a small percentage of the KDD dataset.

Table 2. Type of KDD set connection.

High frequent attacks/ inner types		**Low frequent attacks/ inner types**		**Normal**
Dos	Probe	R2l	U2R	-
6 types	5 types	8 types	4 types	

Table 3. Some of the 41 features (individual connections) [36].

Basic features of individual TCP connections			
1	feature name	description	type
2	duration	length (number of seconds) of the connection	continuous
3	Protocol_ type	type of the protocol, e.g. tcp, udp, etc.	discrete
4	service	network service on the destination, e.g., http, telnet, etc.	discrete
5	Src_ bytes	number of data bytes from source to destination	continuous
6	Dst_ bytes	number of data bytes from destination to source	continuous
7	flag	normal or error status of the connection	discrete
8	land	1 if connection is from/to the same host/port; 0 otherwise	discrete
9	wrong_ fragment	number of ``wrong'' fragments	continuous
10	urgent	number of urgent packets	continuo
Content features within a connection suggested by domain knowledge			
11	hot	number of ``hot'' indicators	continuous

12	num_ failed _logins	number of failed login attempts	continuous
13	Logged _in	1 if successfully logged in; 0 otherwise	discrete
14	Num _compromised	number of ``compromised'' conditions	continuous
15	Root _shell	1 if root shell is obtained; 0 otherwise	discrete
16	Su _attempted	1 if ``su root'' command attempted; 0 otherwise	discrete
17	Num _root	number of ``root'' accesses	continuous
18	Num _file _creations	number of file creation operations	continuous
19	Num _shells	number of shell prompts	continuous
20	Num _access _files	number of operations on access control files	continuous
21	num_ outbound_ cmds	number of outbound commands in an ftp session	continuous
22	is_ hot_ login	1 if the login belongs to the ``hot'' list; 0 otherwise	discrete
23	is_ guest _login	1 if the login is a ``guest'' login; 0 otherwise	discrete
Traffic features computed using a two-second time window			
24	count	number of connections to the same host as the current connection in the past two seconds	continuous
25		Note: The following features refer to these same-host connections.	
26	Serror _rate	% of connections that have ``SYN'' errors	continuous
27	Rerror _rate	% of connections that have ``REJ'' errors	continuous
28	same_srv_rate	% of connections to the same service	continuous
29	diff_srv_rate	% of connections to different services	continuous
30	srv_count	number of connections to the same service as the current connection in the past two seconds	continuous
31		Note: The following features refer to these same-service connections.	
32	srv _serror_rate	% of connections that have ``SYN'' errors	continuous
33	srv_rerror_rate	% of connections that have ``REJ'' errors	continuous
34	srv_diff_host_rate	% of connections to different hosts	continuous

Some string nodes may be found in the KDD data collection. For example, the nntool does not recognize string values for the protocol type and services nodes. To do so, we have converted 18,543 string connections to numeric ones, where each digit represents a distinct string value. The converted and original data set are shown in Table 4.

Table 4. KDD 1999 data set before and after convert to numerical value.

	protocol type	**Service**					**protocol type**	**Service**			
0	"tcp"	"private"	"REJ"	0	0	0	1	6	2	0	0
0	"tcp"	"private"	"REJ"	0	0	0	1	6	2	0	0
0	"tcp"	"private"	"REJ"	0	0	0	1	6	2	0	0
0	"tcp"	"private"	"REJ"	0	0	0	1	6	2	0	0
0	"tcp"	"private"	"REJ"	0	0	0	1	6	2	0	0
0	"tcp"	"ftp"	"RSTO"	0	0	0	1	34	3	0	0
0	"tcp"	"ftp"	"RSTO"	0	0	0	1	34	3	0	0
0	"tcp"	"ftp"	"RSTO"	0	0	0	1	34	3	0	0
0	"tcp"	"private"	"RSTO"	0	0	0	1	6	3	0	0
0	"tcp"	"ftp_data"	"REJ"	0	0	0	1	30	2	0	0

0	"tcp"	"private"	"REJ"	0	0	0	1	6	2	0	0
0	"tcp"	"telnet"	"RSTO"	0	0	0	1	31	3	0	0
0	"tcp"	"telnet"	"RSTO"	0	0	0	1	31	3	0	0
0	"tcp"	"ssh"	"REJ"	0	0	0	1	7	2	0	0
0	"tcp"	"ssh"	"REJ"	0	0	0	1	7	2	0	0
0	"tcp"	"private"	"REJ"	0	0	0	1	6	3	0	0
0	"tcp"	"private"	"REJ"	0	0	0	1	6	3	0	0
0	"tcp"	"smtp"	"RSTO"	0	0	0	1	2	3	0	0
0	tcp'	"smtp"	"RSTO"	0	0	0	1	2	3	0	0

Each object or link is characterized by a vector of 41 characteristics. It should be noted that certain characteristics are continuous while others are nominal (Table 4). Since continuous values are required for the clustering and classification methods, these nominal values will be converted to continuous values first.

Researchers have recommended utilizing true positives, true negatives, false positives, and false negatives to measure the effectiveness of an IDS's detection capabilities [37, 38]. "True positive" means an identified attack in intrusion detection. "True negative" means an IDS correctly identified an expected state. False positives occur when an IDS reports an attack that didn't happen. False positives are caused by loss of recognition, methodological constraints, and environmental phenomena. It measures the detection system's effectiveness. Administrators will continue to ignore system warnings if it stays unsafe. False negatives occur when an intrusion detection system misses an attack. This is probably because the intrusion detection system doesn't have enough information about the specific intrusion, or the recognition information needed to identify the event. The detection system is comprehensive. Because there are so few U2R and R2L attacks in the training set, these numbers are insufficient as a benchmark [39]. Using these quantities may skew performance test results. Precision, recall, and other measures of performance are independent of training and testing sample sizes. Here are some explanations of what they mean:

Precision = $TP/(TP+FP)$.

Recall = $TP/(TP+FN)$.

F-value = $((1+\beta 2)*Recall*Precision)/(\beta 2*(Recall + Precision))$.

As a result of its frequent convergence to the local minimum and inability to train, ANN is inherently unstable. One of the major factors affecting IDS detection reliability. Since the detection consistency is an important factor for ANN-based IDS, we also measure the percentage of successful training as a metric.

7. Results and Discussion

A separate three-layer network is used by both the artificial neural network (ANN) and fuzzy aggregation parts of the system. Previously, we mentioned an empirical formula $\sqrt{(I + O)} + \alpha$ (α = 1-10), that was used to calculate the total number of unseen nodes. Thus, in the experiment, the neural network structure during the ANN stage is denoted by the notation [41; 17; 3], while in the second experiment, it is denoted by the notation [5; 13; 5]. Logsig was used as a transformative function at the input and hidden nodes, while Tansig was used at the output node. In the training phase, the MSE was 0.001. Our momentum factor was 0.2, and our learning rate was 0.01. By following the sampling rules outlined in Section 3.3, we conduct 10 separate experiments.

Initially, following the completion of ANN training for each cluster, error data was extracted and contrasted with MSE **[***If AB(error) 0.001, then TRUE else FALSE***]**, as shown in Table 5.

Table 5. Calculation of fail and successful experiments.

#	Connection Type	Error	Absolute (Error)	Train Results	
1	10	1.913095	1.913094987	FALSE	***48-U2R***
2	10	0.066538	0.066537974	FALSE	***False=9, FN=9***
3	10	0.02912	0.029120247	FALSE	***True=39***
4	1.00E+01	0.007023	0.007022689	FALSE	***TP=39***
5	10	0.020401	0.020401415	FALSE	
6	10	0.027554	0.027553722	FALSE	
7	10	3.378912	3.378912367	FALSE	
8	10	0.372415	0.372415337	FALSE	
9	1.00E+01	0.008834	0.008834305	FALSE	
10	10	0	0	TRUE	
11	10	0	0	TRUE	
12	10	5.17E-09	5.16502E-09	TRUE	
13	10	0	0	TRUE	
14	10	5.68E-09	5.68212E-09	TRUE	
15	10	0	0	TRUE	
16	10	3.05E-12	3.05178E-12	TRUE	
17	10	5.09E-09	5.09197E-09	TRUE	
18	10	0	0	TRUE	
19	1.00E+01	0	0	TRUE	
20	10	0	0	TRUE	
21	10	0	0	TRUE	
22	10	0.000206	0.00020586	TRUE	
23	10	0	0	TRUE	
24	10	0.000262	0.000261882	TRUE	
25	10	0	0	TRUE	

Figures 3 to 6 show the accuracy, recall, F-value, and success rate of training for both the FC-ANN and FC-ANN-MD algorithms when applied to the same set of attacks.

Figure 3 depicts the accuracy results of Wang's algorithm (FC-ANN) [28], which applies Purelin function (PF) as a transformation function to the ANN's final output layer. The prices fall into specific categories on the KDD dataset for a range of cluster sizes. Our findings corroborate the findings of Wang's algorithm, which showed that the best precision results may be attained by splitting the whole cluster population into six equal halves (k=6).

The results of the precision ratio for the modified algorithm (FC-ANN-MD) that employs the Tan-sigmoid function (TF) as a transformation function on the output layer of the ANN structure are depicted in Fig. 3. There is a correlation between the number of clusters and a specific class that the rates belong to in the KDD dataset. Six clusters, we now know, yield the highest precision results.

Comparison of the detection accuracy achieved using the Wang algorithm (FC-ANN) and the modified version (FC-ANN-MD) is shown in Figs. 4 and 5, respectively. The accuracy for different cluster sizes is also provided, from 2

clusters up to 12 clusters, Based on the provided data, it is evident that the use of the Tan-sigmoid transformation function in the output layers yields a higher level of detection accuracy compared to the linear function. Equation (11) is used to calculate accuracy levels.

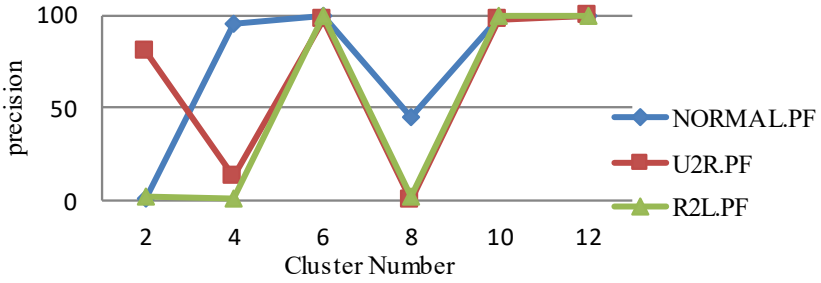

Fig. 1. Precision results using Purelin function.

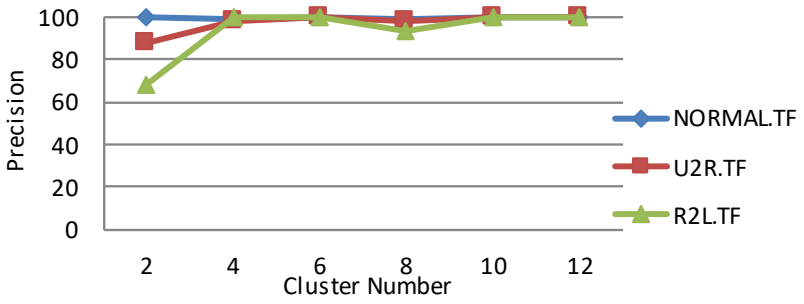

Fig. 2. Precision results use Tansig function.

Figure 5 presents a graphic illustrating the recall value attained by Wang's technique (FC-ANN) while using the Purelin function (PF) as the transformation function on the artificial neural network's output layer. Rates may be classified into several groups depending on the quantity of clusters included in the KDD dataset.

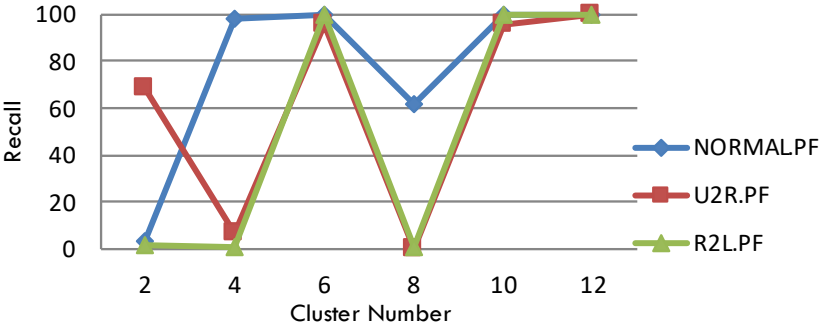

Fig. 3. Recall results use Purelin function.

The Wang's algorithm (FC-ANN) stability, which applies purelin function (PF) as a transformation function to the ANN's output layer, is depicted in Fig. 6. There

is a correlation between the number of clusters and a specific class that the rates belong to in the KDD dataset. This confirms the findings of Wang's algorithm, which showed that dividing the number of clusters into six (k=6) yields the best results for stability, but we also find that the best results can be obtained at twelve clusters (K=12).

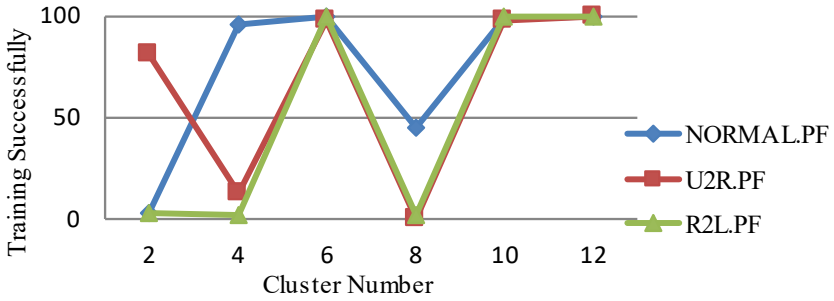

Fig. 4. stability of Wang's algorithm (FC-ANN).

The Precision, Recall and F-value rates for all classes are shown in Table 6. The accuracy rates of R2L, U2R, and Probe are opposite of one another.

The figures clearly indicate the differences in each assessment criterion under various algorithms. This suggests that the FC-ANN-Modification method improves accuracy for low-frequency attacks (U2R, R2L). Tables 6-10 compare the results obtained from implementing the algorithms with varying numbers of clusters ranging from 2 to 12.

Table 6. Performance comparison of the various methods (R2L attack) from PF to TF.

Evolution criteria	FC-ANN	FC-ANN M.D
Precision (%)	51	93.5
Recall (%)	50.5	89.8
F-value (%)	50.6	91.5

Table 1. Performance comparison of various methods (U2R attack) from PF to TF.

Evolution criteria	FC-ANN	FC-ANN M.D
Precision (%)	64.9	97.2
Recall (%)	61.2	94.2
F-value (%)	62.8	95.5

Table 8. Performance comparison of various methods (NORMAL) from PF to TF.

Evolution criteria	FC-ANN	FC-ANN M.D
Precision (%)	73.5	99.8
Recall (%)	77	99.9
F-value (%)	75	99.8

Table 9 presents the statistical data pertaining to the mean accuracy, percentage of successful training, and length required for the transition from PF (Promiscuous

Mode Filtering) to TF (Transparent Firewall) in the context of low-frequency attacks, namely u2r (user-to-root) and r2l (remote-to-local) attacks.

Table 2. Average accuracy, percentage of successful training, and training time from PF to TF for (u2r,r2l) attacks.

Evolution criteria	FC-ANN	FC-ANN M.D	Increase(In)/ decrease(De)
Average accuracy (%)	56.7	93.5	39.4/IN
Percentage of training successfully (%)	58	95.5	39.3/IN
Training time (s)	1808s	6s	99.7/de

Table 10 displays the data pertaining to the accuracy, percentage of successful training, and the duration necessary to advance from a novice to an expert level in the context of high-frequency assaults that are considered representative.

Table 10. Average accuracy, percentage of training successfully, and training time from PF to TF for high frequent attacks (normal).

Evolution criteria	FC-ANN	FC-ANN M.D	Increase(In)/ decrease(De)
Average accuracy (%)	75	99.8	24.8/In
Percentage of training successfully (%)	73.8	99.9	26.1/In
Training time (s)	1808s	6 s	99.7/De

Confusion Matrix Results are another type of evaluation criteria used to determine the accuracy of (U2R and R2L) as low frequent attack and data norm using FC-ANN algorithm and FC-ANN-MD, respectively. Tables 11 and 12 show the accuracy of Confusion Matrix findings when the FC-ANN algorithm and FC-ANN-modified are used.

An in-depth evaluation of the precision of the outputs of the confusion matrix using the FC-ANN method is shown in Table 11.

Table 11. Detailed accuracy of Confusion matrix results with the use FC-ANN algorithm.

Actual class	Predicted class	
Class	Normal	Attack
Normal	0.818	0.226
Attack	0.484	0.543

The improved FC-ANN yields precise results in the Confusion matrix, according to the details provided as depicted in Table 12.

Table 12. Detailed accuracy of Confusion matrix findings using modified FC-ANN.

Actual class	Predicted class	
Class	Normal	Attack
Normal	0.998	0.001
Attack	0.062	0.937

The aforementioned modifications made to the FC-ANN algorithm reveal that the FC-ANN-MD variant enhances the accuracy of detecting rare attacks such as "U2R and R2L" by a significant margin of 39.4 percentage points. The recorded statistic for high-frequency attacks is 24.8. FC-ANN-MD improves the success rate of training, which is expected to be 39.3 percent for low-frequent assaults. There is a significant gap in training time between the FC-ANN system and the modified ones, especially for very frequent attacks (calculated at 26.1). The FC-ANN-MD cuts training time by 99.7 percent, from 1808 seconds to only 6 seconds. Based on these findings, we know that low-frequency assaults are becoming more precise and taking less time to perfect.

8. Conclusion

Many researchers believe Artificial Neural Networks (ANNs) can outperform traditional intrusion detection systems (IDS). For low-frequency attacks, ANN-based IDS needs the detection precision and stability to be improved. A technique called FC-ANN was proposed. This technique utilizes Artificial Neural Networks and fuzzy clustering, with the aim of enhancing the accuracy and stability of intrusion detection systems. However, it is worth noting that the issue of low-frequency assaults continues to persist in the majority of IDS systems.

The ANN structure of Wang's method has been modified to enhance the precision and reliability of low-frequency attacks. The experiment included the use of the FC-ANN method to train specific classes from the KDD1999 dataset.

A comparison was made between the outcomes obtained from this approach and those achieved by the application of the FC-ANN-MD technique on an equal number of classes. The FC-ANN-MD method demonstrates enhanced accuracy and stability in mitigating low-frequency assaults, while also achieving optimal training time. Although it's true that there are more recent datasets available, KDD Cup 1999's rich variety makes it ideal for training machine learning models that can spot low-frequency attacks that may be rare in actual network traffic.

Additionally, it is annotated with attack labels, meaning that each network connection is labelled as normal or one of several types of attacks, and it has become a benchmarking standard in the area of intrusion detection, so many researchers have used it to compare different machine learning algorithms performance. Future work intends to employ additional current datasets to compare the outcomes to what has been accomplished in this study.

References

1. Srilatha, D.; and Shyam, G.K. (2021). Cloud-based intrusion detection using kernel fuzzy clustering and optimal type-2 fuzzy neural network. *Cluster Computing*, 24(3), 2657-2672.
2. Khraisat, A.; Gondal, I., Vamplew, P., and Kamruzzaman, J. (2019). Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity*, 2(1), 2-22.
3. Rashid, O.F.; and Al-Hakeem, M.S. (2022). Hybrid intrusion detection system based on DNA encoding, Teiresias algorithm and clustering method. *Webology*, 19(1), 508-520.

4. Walling, S.; and Lodh, S. (2022). A survey on intrusion detection systems: Types, datasets, machine learning methods for NIDS and challenges. *Proceedings of the 2022 13th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Kharagpur, India, 1-10.
5. Balasaraswathi, V.R.; Shamala, L.M.; Hamid, Y.; Alias Priya, M.P.; Shobana, M.; and Sugumaran, M. (2022). An efficient feature selection for intrusion detection system using B-HKNN and C2 search based learning model. *Neural Processing Letters*, 54(6), 5143-5167.
6. Liao, H.-J.; Richard Lin, C.-H.; Lin, Y.-C.; and Tung, K.-Y. (2013). Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1), 16-24.
7. Da Costa, K.A.P.; Papa, J.P.; Lisboa, C.O.; Munoz, R.; and de Albuquerque, V.H.C. (2019). Internet of things: A survey on machine learning-based intrusion detection approaches. *Computer Networks*, 151, 147-157.
8. Alazab, A.; Abawajy, J.; Hobbs, M.; Layton, R.; and Khraisat, A. (2013). Crime toolkits: The productization of cybercrime. *Proceedings of the 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, Melbourne, Australia, 1626-1632.
9. Lim, J.-H.; and Kim, T.-S. (2020). Optimization of information security investment portfolios based on data breach statistics: A genetic algorithm approach. *Information Systems Review*, 22(2), 201-217.
10. Xin, Y.; Kong, L.; Liu, Z.; Chen, Y.; Li, Y.; Zhu, H.; Gao, M.; Hou, H.; and Wang, C. (2018). Machine learning and deep learning methods for cybersecurity. *IEEE Access*, 6, 35365-35381.
11. Eltanbouly, S.; Bashendy, M.; AlNaimi, N.; Chkirbene, Z.; and Erbad, A. (2020). Machine learning techniques for network anomaly detection: A survey. *Proceedings of the 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies* (*ICIoT*), Doha, Qatar, 156-162.
12. Meiners, C.R.; Patel, J.; Norige, E.; Torng, E.; and Liu, A.X. (2010). Fast regular expression matching using small TCAMs for network intrusion detection and prevention systems. *Proceedings of the* 19th *USENIX Security Symposium*, Washington, USA, 1-8.
13. Kreibich, C.; and Crowcroft, J. (2004). Honeycomb: creating intrusion detection signatures using honeypots. *Computer Communication Review*, 34(1), 51-56.
14. Vigna, G.; and Kemmerer, R.A. (1999). NetSTAT: A network-based intrusion detection system. *Journal of Computer Security*, 7(1), 37-71.
15. Srilatha, D.; and Shyam, G.K. (2021). Cloud-based intrusion detection using kernel fuzzy clustering and optimal type-2 fuzzy neural network. *Cluster Computing*, 24(3), 2657-2672.
16. Liu, L.; Wang, P.; Lin, J.; and Liu, L. (2021). Intrusion detection of imbalanced network traffic based on machine learning and deep learning. *IEEE Access*, 9, 7550-7563.
17. Zhang, L.; and Xu, C. (2022). A intrusion detection model based on convolutional neural network and feature selection. *Proceedings of the 2022 5th International Conference on Artificial Intelligence and Big Data (ICAIBD)*, Chengdu, China, 162-167.

18. Brown, C.; Cowperthwaite, A.; Hijazi, A.; and Somayaji, A. (2009). Analysis of the 1999 DARPA/Lincoln Laboratory IDS evaluation data with NetADHICT. *Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, Ottawa, ON, Canada, 1-7.
19. Samrin, R.; and Vasumathi, D. (2018). Hybrid weighted K-means clustering and artificial neural network for an anomaly-based network intrusion detection system. *Journal of Intelligent Systems*, 27(2), 135-147.
20. Ashfaq, R.A.R.; Wang, X.-Z.; Huang, J.Z.; Abbas, H.; and He, Y.-L. (2017). Fuzziness based semi-supervised learning approach for intrusion detection system. *Information Sciences*, 378, 484-497.
21. Aljumah, A. (2021). IOT-based intrusion detection system using convolution neural networks. *PeerJ Computer Science*, 7:e721, 16-34.
22. Butun, I.; Morgera, S. D.; and Sankar, R. (2014). A survey of intrusion detection systems in wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 16(1), 266-282.
23. Camacho, J.; Pérez-Villegas, A.; García-Teodoro, P.; and Maciá-Fernández, G. (2016). PCA-based multivariate statistical network monitoring for anomaly detection. *Computers & Security*, 59, 118-137.
24. Wang, G.; Hao, J.; Ma, J.; and Huang, L. (2010). A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering. *Expert Systems with Applications*, 37(9), 6225-6232.
25. Elhag, S.; Fernández, A.; Bawakid, A.; Alshomrani, S.; and Herrera, F. (2015). On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on intrusion detection systems. *Expert Systems With Applications*, 42(1), 193-202.
26. Vinayakumar, R.; Alazab, M.; Soman, K.P.; Poornachandran, P.; and Venkatraman, S. (2019). Robust intelligent malware detection using deep learning. *IEEE Access*, 7, 46717-46738.
27. Li, J., Qu, Y., Chao, F., Shum, H.P.H., Ho, E.S.L., and Yang, L. (2018). Machine learning algorithms for network intrusion detection. *AI in Cybersecurity. Intelligent Systems Reference Library*, 151, 151-179.
28. Hamamoto, A.H.; Carvalho, L.F.; Sampaio, L.D.H.; Abrao, T.; and Proena, M.L. (2018), Network anomaly detection system using genetic algorithm and fuzzy logic. *Expert Systems with Applications*, 92, 390-402.
29. Samrin, R.; and Vasumathi, D. (2018). Hybrid weighted K-means clustering and artificial neural network for an anomaly-based network intrusion detection system. *Journal of Intelligent Systems*, 27(2), 135-147.
30. Gao, Y.; Liu, Y.; Jin, Y.; Chen, J.; and Wu, H. (2018). A novel semi-supervised learning approach for network intrusion detection on cloud-based robotic system. *IEEE Access*, 6, 50927-50938.
31. Mohamed, Y.A.; Salih, D.A.; and Khanan, A. (2023). An approach to improving intrusion detection system performance against low frequent attacks. *Journal of Advances in Information Technology*, 14(3), 472-478.
32. Mohamed, Y.A. (2012). I2MANET security logical specification framework. *The International Arab Journal of Information Technology* (*IAJIT*), 9(6), 495-503.

33. Mohamed, Y.A.; and Abdullah, A.B. (2008). Biologically inspired model for securing hybrid mobile ad hoc networks. *Proceedings of the 2008 International Symposium on High Capacity Optical Networks and Enabling Technologies*, Penang, Malaysia, 187-191.
34. Mohamed, Y.A.; and Abdullah, A.B. (2009). Security mechanism for Manets. *Journal of Engineering Science and Technology*, 4(2), 231-242.
35. Alhag, N.M.M.; and Mohamed, Y.A. (2018). An enhancement of data encryption standards algorithm (DES). *Proceedings of the 2018 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*, Khartoum, Sudan, 1-6.
36. KDD Cup 1999. (1999). Computer network intrusion detection. Retrieved February 17, 2023, from https://kdd.org/kdd-cup/view/kdd-cup-1999#:~:text=The%20task%20for%20the%20classifier,connections%20in%20a%20computer%20network.
37. Kemmerer, R.A., and Vigna, G. (2002). Intrusion detection: A brief history and overview. *Computer*, 35(4), supl27 - supl30.
38. Guimaraes, M.V.C.; and Murray, M. (2008). Overview of intrusion detection and intrusion prevention. *Proceedings of the 5th annual conference on Information security curriculum development (InfoSecCD '08)*. Association for Computing Machinery, New York, NY, USA, 44-46.
39. Kuzmanovic, A.; Knightly, E.W. (2006). Low-rate TCP-targeted denial of service attacks and counter strategies. *IEEE/ACM Transactions on Networking*, 14(4), 683 - 696.