Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 9s (2025)

Intrusion Detection System in Wireless Sensor Network using Improved Whale Optimization and Enhanced Fuzzy Neural Network

[1*]P. Vijayalakshmi, [2]Dr. P. M. Gomathi

[1*]Assistant professor, Department of computer science, P.K.R. Arts College for Women, Gobichettipalayam -638452

Mail Id: vijiperumalsas@gmail.com

[2]Dean & Associate professor, Department of Computer Science, P.K.R. Arts College for Women, Gobichettipalayam - 638452

Mail Id: gomathipm@pkrarts.org

Article History:

Received: 12-01-2025

Revised: 15-02-2025

Accepted: 01-03-2025

Abstract:

Wireless sensor networks (WSNs) are regularly employed in risky, uncontrolled situations. WSNs are vulnerable to physical intrusion and security threats. Strong security measures must thus be implemented to secure networks where detecting intrusions are generally acknowledged as one of the most effective security methods for protecting a network from malicious assaults and illegal access. Recent study proposes an improved IDS based on modified binary grey wolf optimizer with support vector machine (GWOSVM-IDS). Optimal wolf counts are found using 3,5,7 wolves. The suggested technique attempts to enhance accuracies of intrusion detections while minimizing processing times with lower false alarm rates and feature counts created by IDS in WSNs. However in existing work sensor nodes consumes more energy to perform packet transmission. More energy consumption may lead to network failure because of this reason energy is the very important parameter in WSNs. Additionally, Grey Wolf Optimizer (GWO) performs poorly in local searches and has a slow convergence rate, both of which might affect intrusion detection effectiveness. Support vector machine (SVM) is unsuitable for managing huge data sets. Increased feature counts per data points during training results in poor SVM performances. To address these challenges, the suggested study suggests node clustering, which is accomplished with weighted KMC. Cluster heads (CHs) will be chosen using Mutation Based Improved Butterfly Optimization (MBIBO). To construct secure communications in WSNs, Improved Whale Optimizations (IWO) for feature selections from input network security laboratory dataset is developed, which reduces time consumption and increases intrusion detection efficiency. Experimental findings demonstrate efficacy of the suggested models in terms of packet delivery ratios, end-to-end latencies, throughputs, and attack detection rates.

Keywords: Intrusion Detection System, Grey Wolf Optimizer, time consumption, Whale Optimization and fuzzy neural network.

Introduction

Recent developments in wireless communication and microelectronics technology have made it possible to create multipurpose sensors at low cost and power. These sensor nodes include wireless connection, data processing, and a capturing device. A wide number of autonomous devices that communicate with one another through short-range radio broadcasts make up WSNs [1, 2, 3].

These sensors may be extremely helpful for a wide range of military and civilian applications, including information gathering and processing from challenging to reach and hostile environments for things like environment monitoring and combat surveillance. Since the features of both wireless infrastructure and WSNs might increase the danger of attacks on the network, many researchers have concentrated on the security of WSNs against attacks or malicious behavior. An IDS is one of the security tools used to keep hackers out of WSNs [4,5,6]..

IDS can accurately defend against internal threats and regarded as second lines of defense, This system makes it possible to identify unusual or suspicious activity on the target under analysis and sounds an alert in the event of an intrusion. I firmly feel that IDS is helpful for both internal and external assaults since cryptography is unable to give the required security in WSNs. Many studies for IDS usages in WSNs exist [7, 8].

In recent works an improved IDS utilizing modified binary GWOSVM-IDS using 3,5,7 wolves to determine optimal wolf counts. The suggested technique intends to enhance accuracy of intrusion detections while minimize processing times in the WSNs with reduced false alarm rates and features generated by IDS.

However in existing work sensor nodes consumes more energy to perform packet transmission. Energy is a crucial component in WSNs since excessive energy consumption might result in network failure. Furthermore, GWO's poor local searching capabilities and sluggish convergence rate may have an impact on intrusion detection effectiveness. Large data sets cannot be processed properly by SVM algorithm and performances deteriorate with more features per data point while training data samples.

To avoid these issues in this proposed work introduces node clustering which is performed using weighted KMC. CHs will be selected using MBIBO and IWO selects features from input network security laboratory dataset to reduce time consumptions and increase intrusion detection efficiencies are introduced. EFNN for attack detection in WSNs are also suggested to establish secure communications.

1. Related Works

Alaparthy and Morgera [11] suggested a model to protect WSNs using the immune theory method known as "Danger Theory." Put differently, the design of a multi-level IDS takes into account the roles played by different immune cells. This is achieved by monitoring energies, data volumes, and transfer frequencies in WSNs and developing outputs based on weights and concentrations which act as feasible foundations for IDS designs in WSNs.

Maleh, et al [12] suggested light weighed hybrid IDS for WSNs. Utilizing cluster-based architecture, our intrusion detection methodology lowers energy usage. This idea enables worldwide lightweight

intrusion detection and identification of hazardous activities using anomaly detections based on SVM and signatures. Their simulation results demonstrated their recommended model's high capability to detect abnormalities with reduced false alarms.

Anitha and Kaarthick [13] presented Laplacian grey wolf optimization (GWO) approach with oppositional foundations for grouping attack classes based on similarities, as well as active learning of SVM classifications. To demonstrate the importance of the suggested algorithm, its output has been assessed using common metrics and contrasted with more contemporary methods. Comparing the suggested algorithm's outcomes with those of the current approaches demonstrates its importance.

Borkar, et al[14] suggested the adaptive chicken swarm optimization algorithm for selections of CHs, an effective clustering method. By using this adaptive strategy, the network's lifetime and scalability are both increased while the time consumption is lowered to a larger extent. In addition, IDS includes dual staged classifications called adaptive SVM classifications where acknowledgment based mechanisms detect fraudulent sensors. This acknowledgment describes various attack types discovered by IDS inclusion, such as DOS, probe, U2R, and R2L. On intrusions, high level security mechanisms transmit responses to other sensors allowing secure packet transmissions. Their recommended strategy was implemented on Python platforms, and their comparisons with other techniques showed it outperforms alternative methods.

Mehmood et al [15] developed context aware knowledge based technique for handling malicious invasions. The knowledge bases powering systems are hosted in Base Stations (BS) which store events generated by network nodes. The occurrences are classified, and it is agreed that the CHs prevent intentionally created, repetitive operations. By leveraging their inference engines, the CHs can also obtain informative data on the maliciousness of unauthorized nodes. The BS's event logging and analysis approach impacts performances of network nodes significantly, relieving them of additional security loads.

Deep learning with recurrent neural networks (RNN-IDS) was recommended for intrusion detection by Yin, et al [16]. They investigated model performances in binary and multiclass classifications and impact of neuron counts and learning rates on recommended model performances. J48, artificial neural network, random forest, support vector machine, and other machine learning (ML) approaches presented by previous researchers were compared using benchmark datasets. According to their experimental results, RNN-IDS outperformed typical ML approaches in binary and multiclass classification, showing that it is an excellent choice for constructing a high-accuracy classification model. The RNN-IDS model improves intrusion detection accuracy while also providing a fresh research technique.

Wazid and Das [17] suggested a novel approach to detecting intrusions in hybrid anomalies that makes advantage of the K-means clustering (KMC) data mining tool. The KMC technique uses training data to automatically create patterns of intrusions for the purpose of detection. Subsequently, intrusions are identified by comparing network activity with these patterns of detection. Examine this method using the Opnet modeler-created WSNs dataset, which includes a variety of variables including end-to-end latency, transmitted and received traffic. The training set had normal network

parameter values while testing set had both normal and abnormal network parameter values created in actual working modes. Blackhole and misdirection nodes are the two categories of malicious nodes that may be found using the suggested method. This method outperforms the equivalent schemes that are already in use, with a detection rate of 98.6% and a false positive rate of 1.2%.

2. Proposed Methodology

The suggested IDS in WSNs is covered in detail in this section. It is divided into four stages: feature selection using enhanced whale optimization is the third phase, CH selection using MBIBO is the second, and node clustering using weighted KMC is the first. The fourth is the use of an EFNN for intrusion detection. Figure 1 depicts the overall architecture of the suggested paradigm.

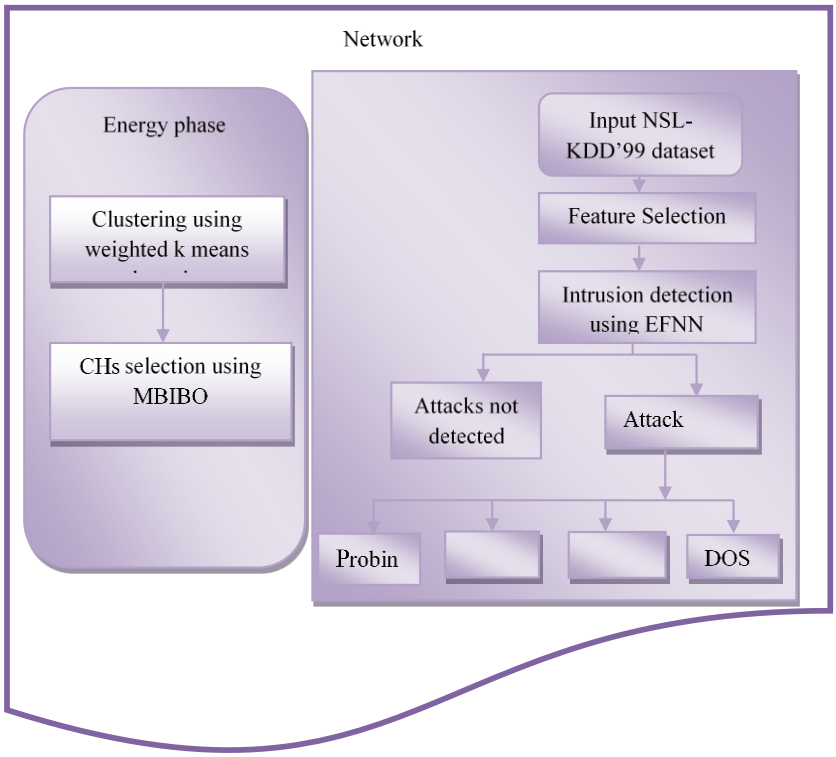

Figure: 1. Overall architecture ofthe proposed model

2.1. Node clustering using weighted KMC

Node clustering in WSNs are primarily used for improving energy efficiency, scalability, and network management. In WSNs, sensor nodes typically have limited power supplies, and their energy consumption must be minimized to prolong the network's lifespan. Clustering helps achieve this by organizing the network into smaller groups, or clusters, where designated CHs handle communications between clustered nodes and central BS.

Clustering is the most prevalent topology management strategy in WSNs, since it clusters nodes for administration and/or distributed task execution, including resource management. The energy consumption in this work is regulated using clustering methods. The network coverage region will determine how these work nodes are grouped. A sensor node should join the Cluster closest to its communication range in order to use less energy. The energy usage will decrease with increasing

distance. KMC is used in this study to cluster nodes. A clustering technique called KMC divides a set of data into k groups. The following stages are repeatedly completed by the algorithm. Analyze the average for each group or cluster and then calculate how far each data point is from the cluster center for each group. Lastly, assign each data point, based on computed distance, to the closest cluster. After the allocation is finished, the cluster center is recalculated and a new distance vector is measured using that center as a base.

2.1.1. Weighted KMC

The KMC technique uses the cluster mean to cluster data and then calculates the distance between any sample and the mean. The sample is indicative of that specific cluster if its distance from the mean is the least. However, the mean tends to go towards dense areas when there are dispersed and dense points in separate clusters, which might lead to some samples being wrongly grouped. Weighted KMC is used in this study to address this problem. Consider a set of x y data that has to be divided into k clusters, with ck serving as each cluster's center. The KMC method's algorithm looks like this:

The weighed means of clusters are computed using weight functions to scan complete data once. In the suggested weighted KMC technique, weights are allocated to each data sample, whereas in the classic k-mean method, the scan is carried out after clusters are completed. The sample's distance from the mean value determines the weights. The following equation is used to calculate the mean as the first step in the suggested procedure.

$$\bar{x} = \frac{\sum_{i=1}^{n} X_i}{n} \quad (1)$$

Where $\bar{x}$ represents means of cluster, n stands for sample counts in particular clusters, and x_i implies samples of clusters.

After determining the mean value, the cluster's radius $\bar{R}$ is computed, the greatest distance between any sample and mean values i.e.

$$\bar{R} = max_i \|x_i - \bar{x}\|_2^2 \quad (2)$$

Cluster radii show areas covered by clusters. Weights of cluster samples are computed using

$$\text{Weight } x_i = \|\bar{x} - x_i\|_2^2 \quad (3)$$

Where $\bar{x}$ represents mean values, and x_i implies samples for which weighs are computed. Weights are assigned to samples for transforming original feature spaces into weighed feature spaces. The weights of samples are computed using:

$$\bar{x}^{(w)} = \frac{\sum_{i=1}^{n} x_i . weightx_i}{\sum_{i=1}^{n} weightx_i} \quad (4)$$

After translating data from their originals to weighted feature spaces, means are calculated again using Equation (5).

$$\mu_k = \frac{\sum_{i=1}^{n} x_i}{n} \quad (5)$$

The ideal cluster is created by minimizing the area that the cluster covers with the weighted mean obtained from the weighted feature space. While the distance within a cluster is measured, the distance between clusters is computed based on the cluster radius.

Figure:2. Weighted KMC

Input: Nodes

Output: Optimal Clusters

START

Step 1: Set Initial values for cluster counts, k and Cluster centers.

Step 2: Set initial values for cluster centroids ← $\mu_1, \mu_2 \ldots \mu_k \in R^n$;

Step 3:Folds ← Split Data to Folds(S)

Step 4:Compute Mean Value using

$$\mu_k = \frac{\sum_{i=1}^{n} x_i}{n} \qquad (5)$$

Step 5:Calculate centroid distances of clusters d with d=$\sqrt{\sum_{j.k=1}^{n}(\mu_k - x_j)^2}$ (6)

Step 6: Allocate clusters based on minimum distances

Step 7: Calculate Radii using eqn (2)

Step 8: Compute weight for each sample

Step 9: Based on the weight reallocate the nodes into different clusters

Step 10: Repeat the procedure until it converges.

END

The effectiveness of node clustering using weighted KMC directly contributes to the success of the next crucial step: Selections of CHs. In the proposed work calculated a weight value is 0.030 to evaluvated the weighted KMC method. By organizing sensor nodes into optimized clusters, the clustering process ensures that nodes are grouped efficiently based on proximity and energy consumption, laying the foundation for strategic CH selection. The weighted KMC step aids CH selection by providing a balanced and well-structured network topology, ensuring that the most suitable nodes are available for CH roles.

2.2. Selections of CHs using Mutation Based Improved Butterfly Optimization (MBIBO)

The process of CH selections in WSNs is crucial as they impact overall performances and energy efficiencies of networks. CHs are in charge of gathering information from the nodes in their cluster and sending it to the BS. Since CHs perform more communication and processing tasks, their energy consumption is higher, and selecting the most optimal CHs is essential to prolong network lifespan and reduce communication overhead.

One of the key techniques for extending the network lifespan in WSNs is clustering. Sensor nodes are grouped into clusters, and CHs are chosen for each cluster. Data is gathered by CHs from the nodes in their respective clusters and sent to the BS as an aggregate. A significant obstacle with WSNs is choosing the right CHs. In this work, CHs are selected using Mutation Based Improved Butterfly Optimization. The Butterfly Optimization Algorithm generates the best CH selections from the network's nodes by drawing inspiration from the food scavenges of butterflies.

Terminologies

1. S: Set of sensor nodes, i.e., S = {s_1, s_2,_, s_n}.

2. C: Sets of CHs, i.e., C = {CH_1, CH_2,_, CH_m}. where, m<n.

3. lj: Sensor cluster node counts j.

4. d_0: Threshold distance.

5. Es_i: Initial energies of sensor node s_i, $1 \leq i \leq n$.

6. ECH_j: Current energies of CHs CHj, $1 \leq j \leq m$.

7. dis s_i;s_j : Distances between any two sensor nodes s_i and s_j.

2.2.1. Derivation of objective function:

Derivations of objective functions are based on the following parameters:

i). Average intra-cluster distances

These distances are defined as averages of sums of all sensor node distances from their selected CHs. i.e.,$\frac{1}{ij}$ dis$\left(CH_{J,}\, BS\right)$. Every sensor node in an intra-cluster communication system uses energy to transmit data to its cluster hub. The average intra-cluster communication distances must be reduced to save energy consumptions. This indicates that a sensor that is close to each sensor node must be chosen to serve as a CH.

ii). Average sink distance

It is defined as ratios of distances between CHs, CH_j and BS to counts of sensor nodes l_ji n CHj i.e.,$\frac{1}{ij}$ dis$\left(CH_{J,}\, BS\right)$. All CHs must send their aggregated data to BS during data routing phases and thus distances between CHs and BS must be reduced for reduced energy usages.

Optimal selections as per Equation (7) aim to lower average intra-clusters and sink distances of CHs.

Minimize f1 =$\sum_{j=1}^{m}$ $\frac{1}{lj}$ ($\sum_{I=1}^{l_j}$ $dis\left(s_i, CH_j\right)$ + dis$\left(CH_{J,}\, BS\right)$) (7)

iii). Energy parameter

E_{CHj} implies current energies of CHs CHj, $1 \leq j \leq m$ chosen from normal sensors in iterations. $\sum_{j=1}^{m} E_{CH_j}$ implies total current energies of selected CHs making it pertinent to select optimal CHs

and thus maximize total current energies of selected CHs, i.e., minimizing their reciprocals. Hence, the second objective is:

Minimize f2 $=\frac{1}{\sum_{j=1}^{m} E_{CH_j}}$ (8)

The Algorithmic steps followed are:

1. Butterflies produce a scent (score) and are drawn to one another according to this fragrance (the goal function); 1.

2. Butterflies travel randomly to the butterfly that has the least fragrance;

3. The intensity of a butterfly is determined by both the score and the objective function.

The perceived scent (f) of MSBOA may be expressed as a function using the equation (9),

$$f = cI^a \quad (9)$$

Where, c∈ [0, ∞] represents sensory modalities I implies computed stimulus intensities using scores and associated objective functions; and $a \in [0,1]$ are power exponents dependent on mutation operators or varying degree of fragrances. MSBOA has to main steps namely local and global searches in movements of butterflies which are oriented towards stronger fragrances (optimal selections of CHs) and depicted as Equation (10),

$$x_i^{t+1} = x_i^t + (r^2 * g^* - x_i^t) \times f_i \quad (10)$$

Where x_i^t represents locations of i[th] butterflies at times t, g^* stands for current best locations, f_i signifies fragrances of i[th] butterflies, and $r \in [0,1]$ - random number. Localized random walks are depicted in Equation (11),

$$x_i^{t+1} = x_i^t + (r^2 \times x_j^t - x_k^t) \times f_i \quad (11)$$

Where, x_j^t implies positions of j[th] butterflies, x_k^t stands for k[th] butterfly position and r signifies random numbers in intervals [0, 1]. MSBOA additionally uses switch probabilities p, to switch between global and intensive local searches.

Mutation Score Butterfly Optimization Algorithm (MSBOA)

1. **Begin**

2. Generate initial populations P containing n butterflies $pop_i(i = 1, 2, \ldots, n)$,objective functions $f(x), x = (x_1, x_2, \ldots, x_d)^T$, with d signifying dimension counts.

3. Stimuli intensities I_i at pop_i are found using fitness values f(pop_i) with scores and minimizations in objective functions

4. Defining sensor modalities c, power exponents a by mutation operators, and switch probabilities p.

5. While stop criteria not met do

5.1. For butterflies in populations P do

Compute fragrances f using Equation (9).

5.2. End for

5.3. Evaluating and ranking populations P, and finding best features in populations.

5.4. for butterflies in populations P do

5.4.1. Generate randomized values r ∼U[0, 1].

5.4.2. if (r < p)

Implement global searches with Equation (10).

5.4.3. else

Implement local searches with Equation (11).

5.4.4. end if

5.5. end for

5.6. Update values for power exponents a using mutation operators.

6. End while

7. Output best solutions and optimal values.

8. **End**

The benefits of Combined Role of Sensor Modality and Power Exponent:

- sensor modalities c and power exponents a work in tandem to control the intensity and direction of movement for the butterflies.
  - c regulates how attractive a butterfly is based on its fragrance (fitness).
  - a adjusts how strongly other butterflies move towards this attraction.
- By adapting these parameters, MBIBO efficiently balances between solution space's new explorative and exploitve areas for refinements in selection process of CHs making them more efficient and adaptive.

The M SBOA optimizes the Selections of CHs process by simulating butterfly behaviour to explore and exploit potential solutions (candidate CHs). The algorithm's global and local search mechanisms ensure that the best possible CHs are chosen based on energy availability, network topology, and distance minimization, thus improving the overall performance and lifespan of the WSNs.

2.3. Feature Selections WOA

The next critical step in the methodology is Feature Selection using Improved Whale Optimization Algorithm (IWOA). After clustering and selecting CHs, the focus shifts to reducing the computational complexity and energy consumption involved in detecting intrusions in WSNs, Feature selection is essential because large datasets, like the NSL-KDD'99 dataset, contain numerous

features, many of which may be redundant or irrelevant. Processing all these features increases time complexity, energy consumption, and can lead to less accurate intrusion detection results.

Input NSL-KDD'99 database might have more features and consume more time to get classification results. This work uses feature selections based on Improved Whale Optimization (IWOA). Recently, the revolutionary stochastic population-based optimization approach called WOA which draws inspiration from nature, finds optimal solutions utilizing groups of search agents for optimizations. By using bubble-net hunting technique, WOA mimics actions of humpback whales as they pursue their preys. The three general processes of WOAs are to surround preys, assault using bubble nets, and hunt for best preys [18,19, 20].

Whales enclose their prey including fishes while updating their positions to find optimal solutions. Mathematically Eqs. (12) and (13) depict WOA.

$$X(t+1) = X^*(t) - A.|C.X^*(t) - X(t)| if p < 0.5 \quad (1) \quad (12)$$

$$X(t+1)|C.X^*(t) - X(t)|.e^{bl}\cos(2\pi t) + X^*(t) if p \geq 0.5 \quad (13)$$

Where X implies vectors of whales' positions; t represents times or iteration indices; X* implies best solutions found so far; A=2a. (r-a);C=2.r; a are coefficient vectors that linearly decrease from 2 to 0 in iterations; r stands for random vectors between 0 and 1; b implies constant values that define shapes of logarithmic spirals based on paths and is 1 for this work; l stands for random numbers between - 1 and 1; p implies random numbers between 0 and 1 and used to switch between (12) and (13) while updating whales' positions; Eqs. (12) and (13) have 50% probabilities implying during optimizations whales select paths randomly with equal chances. During bubble-net phases, A's random values are in the range [- 1, 1], however in searching phases, these values may be larger than or less than 1. Search processes are illustrated as Eq. (14).

$$X(t+1) = X_{rand} - A\,.|C.X_{rand} - X(t)\ \ | \quad (14)$$

Random searches with values of |A| > 1 emphasize searches and require WOA algorithm to do global searches. WOA searches begin with generations of random solutions. The responses are then updated in iterations and searches continue until preset max. iterations are reached.

2.3.1. IWOA

The good trade-off between exploration and exploitation, two critical components of an optimization algorithm, allows for a precise solution to be obtained by escaping the local optima. In WOA, a search agent's step size decreases linearly as iteration counts increases. This step size is determined by a parameter known as A. Nonetheless, it has been demonstrated that insufficient divergence restricts WOA's capacity to capture a local optimum in later rounds.

This paper employs an updated whale optimization approach to get around such problems. This changes the value A by introducing the levy flying function. It improves WOA's capacity for simultaneous exploration and exploitation.

The Levy probability distribution function, a power-law function, is utilized in Levy flight to determine jump sizes where Levy distributions can be mathematically formulated as:

$$L(s,\gamma,\mu) = \begin{cases} \sqrt{\frac{\gamma}{2\pi}} \exp\left[-\frac{\gamma}{2(s-\mu)}\right] \frac{1}{(s-\mu)^{3/2}} if\ 0 < \mu < \infty \\ 0 \qquad\qquad if\ s \leq 0 \end{cases} \tag{15}$$

Where μ, γ, and s are positions and scales which control scale distributions and samples in distributions respectively.

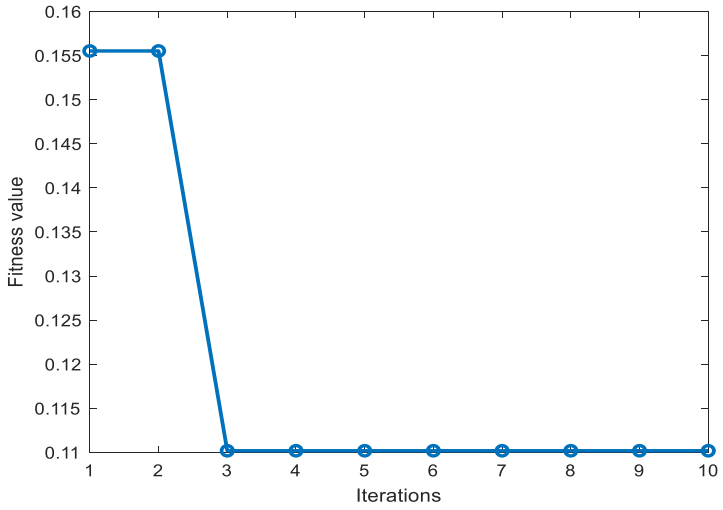

Figure:2. Convergence curve of IWO

The figure 2 depicts a line graph showing the relationship between iterations and fitness values in an optimization algorithm. The graph reveals a significant drop in the fitness value from approximately 0.155 at iteration 1 to about 0.11 at iteration 2. After the second iteration, the fitness value stabilizes and remains constant at approximately 0.11 through iterations 3 to 10, indicating that the optimization process has likely converged. Using Mean Squared Error (MSE) as the fitness function in the Improved Whale Optimization Algorithm not only aids in effectively assessing the quality of solutions but also demonstrates the algorithm's capability to minimize errors in predictive modelling.

In the IWOA, Levy flights play critical roles in enhancing explorations and exploitations of the algorithm, particularly to avoid issues of getting trapped in local optimums during later iterations.

2.3.2. Role of Levy Flight in IWOA:

1. Exploration and Exploitation Balance:

- Explorations refers to algorithm's abilities to search broadly across solution spaces, while exploitations refer to focusing on refining current best solutions.

- In standard Whale Optimization Algorithm (WOA), the parameter A controls the step size of the whales' movement. However, as iterations increase, A decreases linearly, reducing the step size and thus restricting the ability to explore new regions of the search space.

- To enhance both exploration and exploitation simultaneously, Levy flight is used to introduce random long-distance jumps, which allows for more diverse movements even in later

stages of the search. This nonlinear random walk based on the Levy distribution helps escape local optima by allowing whales to make larger, random steps.

Algorithm for IWO

START

1. import data
2. Set initial locations of whales **X**
3. Compute whales fitness
4. Set initial values of **a** and **r**, calculate **A** and **C**
5. Set initial **X*** as best hunters' whale locations
6. initialize $t = 1$
7. **while** $t \leq$ max iterations **do**
8. **for** each hunting whale **do**
9. **if** $p < 0.5$
10. **if** $|\mathbf{A}| < 1$
11. update existing locations of hunting whales with (12)
12. **else if** $|\mathbf{A}| \geq 1$
13. another search agent randomly
14. update existing locations of hunting whales with (12)
15. **end if**
16. **else if** $p \geq 0.5$
17. update existing locations of hunting whales with (13)
18. **end if**
19. **end for**
20. update **X*** on better solutions
21. $t = t + 1$
22. **end while**
23. output **X*** Best Features
24. **END**

The IWOA is a key component of the proposed methodology for feature selection optimization. By reducing the dataset's dimensionality, IWOA identifies the most relevant features, enhancing the classification model's overall performance while simultaneously reducing time complexity and energy consumption. The inclusion of the Levy flight strategy within IWOA ensures a thorough exploration of the solution space, avoiding local optima and leading to the discovery of the optimal set of features. Out of the 41 initial features, the algorithm selects the 15 most effective for the detection task: Hot, Num_root, Num_shells, Num_access_files, Num_outbound_cmds, Is_host_login, Count, Serror_rate, Rerror_rate, Same_srv_rate, Dst_host_srv_count, Dst_host_same_srv_rate, Dst_host_srv_diff_host_rate, Dst_host_serror_rate, and Dst_host_srv_rerror_rate.The selection of these 15 specific features by the IWOA is based on their strong relevance to distinguishing between normal and malicious network behavior. Each feature plays a unique and crucial role in identifying patterns linked to network intrusions:

- **Hot, Num_root, Num_shells, Num_access_files, and Num_outbound_cmds:** These features are directly tied to unauthorized access attempts, such as gaining root privileges or accessing sensitive files, which are critical indicators of potential system compromise (e.g., Root or User to Root attacks).

- **Is_host_login:** This feature helps detect abnormal login patterns, indicating unauthorized access or probing attempts, which are important in identifying login-related vulnerabilities.

- **Count, Serror_rate, and Rerror_rate:** These features measure the frequency of connections and errors during those connections, helping to identify denial-of-service (DoS) or scanning attacks where a high volume of error-prone connections is typical.

- **Same_srv_rate, Dst_host_srv_count, and Dst_host_same_srv_rate:** These features monitor the frequency and consistency of connections to a particular service or destination, allowing the detection of probing or flooding attacks, which target specific services repeatedly.

- **Dst_host_srv_diff_host_rate, Dst_host_serror_rate, and Dst_host_srv_rerror_rate:** These features track the variance in service errors across different hosts, which are key indicators of network-level anomalies, such as Distributed Denial of Service (DDoS) attacks where multiple hosts are targeted.

Each of these selected features plays a crucial role in differentiating between normal and malicious network behaviours. While some features may exhibit redundancy or high correlation, the IWOA's exploration strategy, enhanced by Levy flight, prevents the selection of redundant features and ensures that the chosen features complement each other. This careful selection of features greatly improves the accuracy of network traffic classification, allowing for more effective intrusion detection while minimizing unnecessary computational costs.

2.4. Intrusion detection using EFNN

After feature selection using the IWOA, the EFNN is employed to categorize network activities as either normal or attacks based on prior history and observed behaviors. The work utilized EFNN for WSN intrusion detection, which combines the benefits of neural networks with fuzzy logic systems. The inherent drawbacks of each technique—such as fuzzy logic systems' subjectivity and neural networks' implicit knowledge representation—are addressed by this hybrid method. This work uses a fuzzy neural network to identify intrusions in wireless sensor networks Neural networks have the problem of being unable to collect implicit information, whereas fuzzy logic systems use subjective and heuristic approaches. The identification of fuzzy rules, input, and The design of a fuzzy logic system is time-consuming since output scaling factors and membership function selection are determined by trial and error. These drawbacks of neural networks and fuzzy logic systems are handled by joining their learning capacities with robustness of fuzzy logics whose principles are embedded in networks' structures. Moreover, they provide natural frameworks for consistently combining linguistic information as IF-THEN rules with numerical information's in input/output pair forms [21, 22].

The network's regular activity and the nodes' past history will be used to identify assaults based on the EFNN technique. Below are definitions of attacks:

- **Probing -Probe Attack:** Before launching an attack, make sure you have access to all network data through probing or probe attacks.
- **User to Root (U2R) Attack:** An attacker first gains access to a regular user account before using system flaws to obtain the root.
- **Remote to user (R2L):** One type of computer network attack known as "remote to user" (R2L) happens when an attacker sends a sequence of packets to a server or other machine across a network to which the attacker does not have local user privileges.
- **Denial-of-Service** (DoS): DoS attacks aim to bring down computers or networks so that intended users cannot access it.

To train the EFNN, both legitimate network activity and malicious activity will first be monitored. And then, based on that training, this algorithm will examine every node's past and present behavior during the testing phase, classifying it as an attacker node if it has engaged in any attacker activity, such as the aforementioned DOS, U2R, R2L, and probing attacker movements. If not, it will categorize them as either regular activity or an assault that was missed.

2.4.1. EFNN

Gaussian membership function is used by Basic FNN to calculate membership in the first layer. However, there is a compromise with these systems' accuracy. This study computes the membership value using a uniform distribution in order to get around such problems.

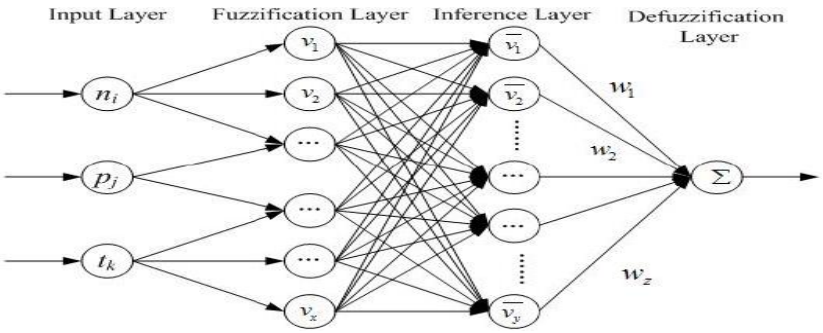

Figure.3. Structure of the four-layered fuzzy neural network

Fuzzy neural network has four layers,

Layer 1- Input layers: No computations are done in these layers. Nodes in this layer correspond to input variables and only transmit input values to next layers directly [23]. i.e.

$$o^{(1)} = a_i^{(1)} = x_i \tag{16}$$

Where $x_i, i = 1,2, \ldots., M,$ are the input variables of the FNN.

Layer 2—Membership function layer: Fuzzification is the process of converting crisp input values into fuzzy values. In the EFNN, the fuzzification is done in Layer 2 where the input values from Layer 1 are passed to membership functions. Instead of using a Gaussian membership function, the EFNN uses a uniform distribution to calculate the membership values. Nodes in this layer are membership functions with uniform distributions corresponding to labels of input in Layers 1.

$$F(x) = \frac{1}{B-A} \quad (17)$$

For $A \leq x \leq B$

Where A is the location parameter and (B - A) is the scale parameter. The case where A = 0 and B = 1 is called the standard uniform distribution.

Layer 3—Rule layer: The inference procedure takes place at Layer 3 of the EFNN. This layer represents fuzzy rules, and the model matches their preconditions using an AND operation. A node in this tier represents one fuzzy logic rule and conducts precondition matching on that rule. Use the AND operation for each Layer 2 node.

$$o^{(3)} = \prod_{i=1}^{M} \quad a_i^{(3)} = e^{-[D_j(x-m_j)]^T [D_j(x-m_j)]} \quad (18)$$

Where $D_j = \text{diag}\left(\frac{1}{\sigma 1_J}, \ldots \ldots, 1/\sigma MJ\right), m_j = [m_{1j}, m_{2j}, \ldots . m_{Mj}]^T$ X= $[x_1, x_2, \ldots . x_M]^T$ the FNN input vector. The output of a Layer-3 node represents the firing strength of the corresponding fuzzy rule.

Layer 4—Output layer: The defuzzification process happens in Layer 4, where the fuzzy outputs from the inference process are converted back into a single crisp value. The single node $o^{(4)}$ in this layer is labelled with Σ, which computes the overall output as the summation of all input signals.

$$o^{(4)} = \sum_{j=1}^{N} d_j \times a_j^{(4)} \times d_o \quad (19)$$

Where connecting weights are output action strengths of Layer 4 outputs associated with Layer 3 rules and scalars are bias values. Based on the above procedure irrigation requirement will be finding for all attack [24,25]. One key enhancement is replacing the Gaussian membership function with a uniform distribution for calculating the membership values in Layer. Here's a breakdown of the role of the Gaussian membership function and how the EFNN improves upon it.

Advantages of EFNN:

- Smooth transition: The smooth curve of the uniform function offers a gradual transition between different membership degrees.
- Simplicity: uniform functions are easy to compute and implement.
- Reduced Complexity: The uniform distribution simplifies the membership function calculations, reducing the overall computational complexity, which is beneficial for applications requiring real-time decision-making, such as intrusion detection.

The EFNN effectively improves the classification of network activities and attack detection in WSNs. By utilizing uniform membership functions and optimized rules, the EFNN enhances accuracy while reducing the complexity of design, making it a powerful tool for intrusion detection in complex network environments.

3. RESULTS AND DISCUSSION

The experiments conducted on the suggested model are analyzed in this section. The use of MATLAB facilitates the implementation of this concept. To assess the suggested method, a comparison is made between the already available KMC and SVM algorithms and the suggested EFNN in terms of accuracy, specificity, time complexity, energy consumption, and error rate using the network security laboratory dataset (NSL-KDD'99). Here, dividing the dataset into 70% training and 30% testing is a standard method for assessing the effectiveness of the model. One may obtain the open-source dataset from the internet. Features from [16] in the following table were implemented in this work:1.

Table.1. Features of NSL-KDD dataset.

	FEATURES	TYPES	NO	FEATURES	TYPES
1	duration	Continuous	22	is_guest_login	Symbolic
2	Protocol_type	Symbolic	23	count	Continuous
3	service	Symbolic	24	srv_count	Continuous
4	flag	Symbolic	25	serror_rate	Continuous
5	src_bytes	Continuous	26	Srv_serror_rate	Continuous
6	dst_bytes	Continuous	27	rerror_rate	Continuous
7	land	Symbolic	28	srv_rerror_rate	Continuous
8	'wrong_fragment'	Continuous	29	same_srv_rate	Continuous
9	urgent	Continuous	30	diff_srv_rate	Continuous
10	hot	Continuous	31	srv_diff_host_rate	Continuous
11	num_failed_logins	Continuous	32	dst_host_count	Continuous
12	logged_in	Symbolic	33	dst_host_srv_count	Continuous
13	num_compromised	Continuous	34	dst_host_same_srv_rate	Continuous
14	root_shell	Continuous	35	dst_host_diff_srv_rate	Continuous
15	su_attempted	Continuous	36	dst_host_same_src_port_rate	Continuous
16	num_root	Continuous	37	dst_host_srv_diff_host_rate	Continuous
17	num_file_creatio	Continuo	38	dst_host_serror_rate	Continuous

	ns	us			
18	num_shells	Continuous	39	dst_host_srv_serror_rate	Continuous
19	num_access_files	Continuous	40	dst_host_rerror_rate	Continuous
20	num_outbound_cmds	Continuous	41	dst_host_srv_rerror_rate	Continuous
21	is_host_login	Symbolic			

The confusion matrix presented below summarizes the performance of a classification model on a four-category experiment using the intrusion detection dataset. It allows us to analyse how well the model distinguishes between different types of network attacks: Denial-of-Service (DoS), Probing, Remote-to-User (R2L), and User-to-Root (U2R). Table 2 describes the confusion matrix of the proposed work. The diagonal entries of the matrix represent the true positives for each class, showcasing the model's ability to accurately identify attacks. For instance, the model correctly classified a substantial 17,225 instances as DoS attacks, demonstrating its effectiveness in recognizing this prevalent attack type. Similarly, it identified 3,897 instances of Probing correctly, along with 399 instances of R2L attacks and an impressive 18,898 instances of U2R attacks.

Table.2. Confusion Matrix for the Four-Category Experiments on Dataset

Actual Class / **Predicted Class**	**DOS**	**Probing**	**R2L**	**U2R**
DOS	17225	89	2	119
Probing	220	3897	9	255
R2L	232	40	399	359
U2R	102	27	2	18898

Table.3. Performance Comparison Results

Metrics		**Methods**				
	NB	**RF**	**MLP**	**SVM**	**GWOSVM**	**EFNN**
Accuracy (%)	78.9272	80.2451	85.4561	87.8216	98.0565	98.3108
Error rate (%)	21.0728	19.7549	14.5439	12.1784	1.9435	1.6892
Specificity (%)	87.6578	89.8912	91.0152	93.8615	94.5671	98.8255
Time complexity(sec)	35.6712	30.0788	28.2731	25.5231	20.1248	3.8459

Table.4. Comparison of the Energy Consumption with the other methods

No. of nodes	Energy consumption (J)					
	NB	RF	MLP	SVM	GWOSVM	EFNN
30	18	16	15	14	13	8
60	38	27	21	17	15	10
90	41	37	29	25	20	15
120	52	48	41	35	28	22
150	54	50	44	39	35	28

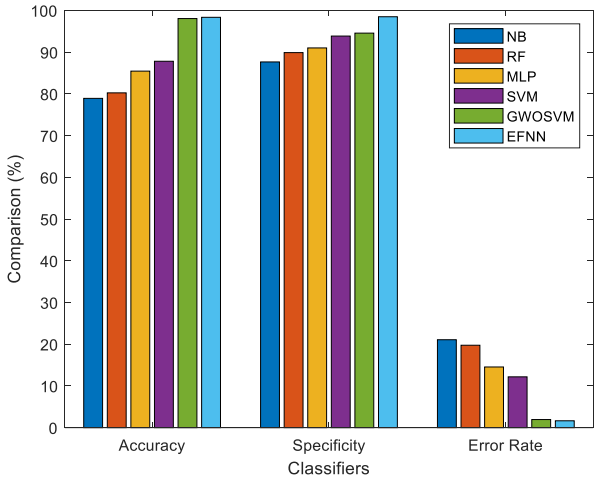

Figure: 4. Accuracy, Error rate and specificity results Comparison of Various Classifiers

The figure above compares the performance of the classifiers NB, RF, MLP, SVM, GWOSVM, and the suggested EFNN schemes in terms of accuracy, error rate, and specificity. In the graph above, several approaches are displayed on the X-axis, while accuracy, error rate, and specificity numbers are represented on the Y-axis. The findings show that the newly presented EFNN model outperformed the other existing models. For example in the above figure proposed EFNN produces higher accuracy which is 98.3108 (%) while available, NB, RF, MLP, SVM and GWOSVM technique yields only 78.9272(%),80.2451(%),85.4561(%),87.8216(%) and 98.0565 (%) respectively.

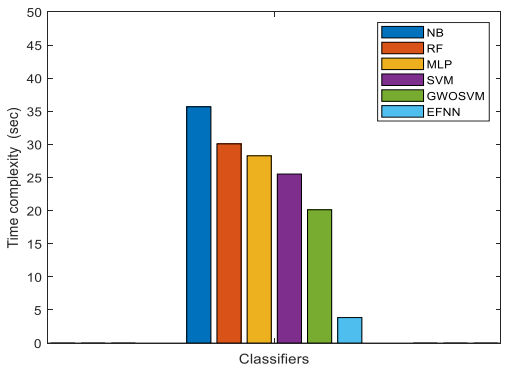

Figure: 5. Time complexity results Comparison of Various Classifiers

Performance comparison for **Time complexity** metrics with the existing classifier NB, RF, MLP, SVM, GWOSVM, proposed EFNN techniques is depicted in the above figure. In the proposed work, IWO employs fitness to identify relevant features, reducing the temporal complexity of the EFNN. on the following graph, several methodologies are depicted on the X and Y axes. The time complexity values are represented. As seen in the findings, the newly presented EFNN model gave reduced time complexity results 3.8459(sec), whereas the available NB, RF, MLP, SVM, and GWOSVM techniques yielded 35.6712(sec), 30.0788(sec), 28.2731(sec), 25.5231(sec), and 20.1248(sec), respectively.

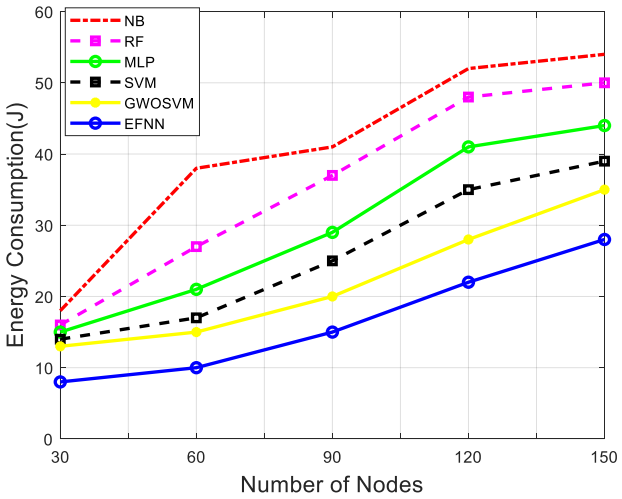

Figure:6. Energy consumption results

The figure above compares the energy consumption results of the proposed EFNN to current classifier methods such as NB, RF, MLP, SVM, and GWOSVM. The X and Y axes in the graph above reflect the number of nodes. Energy usage figures are displayed. As seen in the data, the newly adopted EFNN model yielded reduced energy usage results 28(J)while available NB, RF, MLP, SVM and WOSVM technique consumes 54(J), 50(J), 44(J), 39(J), 35(J) respectively.

4. CONCLUSION AND FUTURE WORK

WSNs are constructed using little microscopic nodes which are oftentimes heavily distributed in open and unprotected environment. WSNs are vulnerable to several forms of assault and are of interest to enemies in numerous applications, especially in the military. Even when precautions are taken to guard against assaults, certain attacks are unavoidable and cannot be stopped by recognized precautions. In order to stop the hacker from damaging the network, the intrusion detection system (IDS) can gather data on the attack methods and aid in the creation of a defense mechanism. Weighted KMC is used in this suggested study to conduct node clustering. MBIBO will be used to choose the CHs.To aid in feature selection, an IWO is presented. and then suggested using EFNN to identify attacks in WSNs and provide secure connection. The suggested model performs better in terms of accuracy, specificity, error rate, and energy consumption, according to experimental data. But, because this EFNN yields approximations, future classification efforts will need to employ alternative classifiers.

REFERENCES

1. Sajjad, S.M., Bouk, S.H. and Yousaf, M., 2015. Neighbor node trust based intrusion detection system for WSN. Procedia Computer Science, 63, pp.183-188.
2. Can, O. and Sahingoz, O.K., 2015, May. A survey of intrusion detection systems in wireless sensor networks. In 2015 6th international conference on modeling, simulation, and applied optimization (ICMSAO) (pp. 1-6). IEEE.
3. Hammoudeh, M., Al-Fayez, F., Lloyd, H., Newman, R., Adebisi, B., Bounceur, A. and Abuarqoub, A., 2017. A wireless sensor network border monitoring system: Deployment issues and routing protocols. IEEE Sensors Journal, 17(8), pp.2572-2582.
4. Ramson, S.J. and Moni, D.J., 2017, February. Applications of wireless sensor networks—A survey. In 2017 international conference on innovations in electrical, electronics, instrumentation and media technology (ICEEIMT) (pp. 325-329). IEEE.
5. Nayak, P. and Vathasavai, B., 2017. Energy efficient clustering algorithm for multi-hop wireless sensor network using type-2 fuzzy logic. IEEE Sensors Journal, 17(14), pp.4492-4499.
6. Aljawarneh, S., Aldwairi, M. and Yassein, M.B., 2018. Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. Journal of Computational Science, 25, pp.152-160.
7. Vijayanand, R., Devaraj, D. and Kannapiran, B., 2018. Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection. Computers & Security, 77, pp.304-314.
8. Bhushan, B. and Sahoo, G., 2018. Recent advances in attacks, technical challenges, vulnerabilities and their countermeasures in wireless sensor networks. Wireless Personal Communications, 98(2), pp.2037-2077.
9. Tao, P., Sun, Z. and Sun, Z., 2018. An improved intrusion detection algorithm based on GA and SVM. Ieee Access, 6, pp.13624-13631.
10. Hajisalem, V. and Babaie, S., 2018. A hybrid intrusion detection system based on ABC-AFS algorithm for misuse and anomaly detection. Computer Networks, 136, pp.37-50.
11. Alaparthy, V.T. and Morgera, S.D., 2018. A multi-level intrusion detection system for wireless sensor networks based on immune theory. IEEE Access, 6, pp.47364-47373.
12. Maleh, Y., Ezzati, A., Qasmaoui, Y. and Mbida, M., 2015. A global hybrid intrusion detection system for wireless sensor networks. Procedia Computer Science, 52, pp.1047-1052.
13. Anitha, P. and Kaarthick, B., 2021. Oppositional based Laplacian grey wolf optimization algorithm with SVM for data mining in intrusion detection system. Journal of Ambient Intelligence and Humanized Computing, 12(3), pp.3589-3600.
14. Borkar, G.M., Patil, L.H., Dalgade, D. and Hutke, A., 2019. A novel clustering approach and adaptive SVM classifier for intrusion detection in WSN: A data mining concept. Sustainable Computing: Informatics and Systems, 23, pp.120-135.
15. Mehmood, A., Khanan, A., Umar, M.M., Abdullah, S., Ariffin, K.A.Z. and Song, H., 2017. Secure knowledge and cluster-based intrusion detection mechanism for smart wireless sensor networks. IEEE Access, 6, pp.5688-5694.
16. Yin, C., Zhu, Y., Fei, J. and He, X., 2017. A deep learning approach for intrusion detection using recurrent neural networks. Ieee Access, 5, pp.21954-21961.

17. Wazid, M. and Das, A.K., 2016. An efficient hybrid anomaly detection scheme using K-means clustering for wireless sensor networks. Wireless Personal Communications, 90(4), pp.1971-2000.
18. Sharawi, M., Zawbaa, H.M. and Emary, E., 2017, February. Feature selection approach based on whale optimization algorithm. In 2017 Ninth international conference on advanced computational intelligence (ICACI) (pp. 163-168). IEEE.
19. Wu, X., Zhang, S., Xiao, W. and Yin, Y., 2019. The exploration/exploitation tradeoff in whale optimization algorithm. IEEE Access, 7, pp.125919-125928.
20. Ling, Y., Zhou, Y. and Luo, Q., 2017. Lévy flight trajectory-based whale optimization algorithm for global optimization. IEEE access, 5, pp.6168-6186.
21. Han, H.G., Wu, X.L., Liu, Z. and Qiao, J.F., 2017. Design of self-organizing intelligent controller using fuzzy neural network. IEEE Transactions on Fuzzy systems, 26(5), pp.3097-3111.
22. Figueroa-García, J.C., Ochoa-Rey, C.M. and Avellaneda-González, J.A., 2015. Rule generation of fuzzy logic systems using a self-organized fuzzy neural network. Neurocomputing, 151, pp.955-962.
23. Tang, J., Liu, F., Zou, Y., Zhang, W. and Wang, Y., 2017. An improved fuzzy neural network for traffic speed prediction considering periodic characteristic. IEEE Transactions on Intelligent Transportation Systems, 18(9), pp.2340-2350.
24. Lin, F.J., Lu, K.C., Ke, T.H., Yang, B.H. and Chang, Y.R., 2015. Reactive power control of three-phase grid-connected PV system during grid faults using Takagi–Sugeno–Kang probabilistic fuzzy neural network control. IEEE Transactions on Industrial Electronics, 62(9), pp.5516-5528.
25. Wang, Z. and Fei, J., 2021. Fractional-Order terminal sliding mode control using self-evolving recurrent chebyshev fuzzy neural network for MEMS gyroscope. IEEE Transactions on Fuzzy Systems.