I.J. Information Engineering and Electronic Business, **2025, 3, 62-83**
Published Online on June 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijieeb.2025.03.05

DSNFyS: Deep Stacked Neuro Fuzzy System for Attack Detection and Mitigation in RPL based IoT

Prashant Maurya*
Department of Computer Science, Institute of Science, Banaras Hindu University, Varanasi, 221005, India
E-mail: prashant.maurya17@bhu.ac.in
ORCID iD: https://orcid.org/0000-0003-4212-7790
*Corresponding Author

Vandana Kushwaha
Department of Computer Science, Institute of Science, Banaras Hindu University, Varanasi, 221005, India
E-mail: vandanakus@bhu.ac.in
ORCID iD: https://orcid.org/0000-0001-5631-1914

Received: 27 December, 2024; Revised: 13 January, 2025; Accepted: 07 February, 2025; Published: 08 June, 2025

Abstract: The Routing Protocol for Low-Power and Lossy Networks (RPL) is a widely adopted protocol for managing and optimizing routing in resource-constrained Internet of Things (IoT) environments. RPL operates by constructing a Destination-Oriented Directed Acyclic Graph (DODAG) to establish efficient routes between nodes. This protocol is designed to address the unique challenges of IoT networks, such as limited energy resources, unreliable wireless links, and frequent topology changes. RPL's adaptability and scalability render it particularly suitable for large-scale IoT deployments in various applications, including smart cities, industrial automation, and environmental monitoring. However, the protocol's vulnerability to various security attacks poses significant threats to the reliability and confidentiality of IoT networks. To address this issue, a novel deep-stacked neuro-fuzzy system (DSNFyS) has been developed for attack detection in RPL-based IoT. The proposed approach begins with simulating RPL routing in IoT, followed by attack detection processing at the Base Station (BS) using log data. Data normalization is accomplished through the application of min-max normalization techniques. The most crucial features are then identified through feature selection, utilizing information gain and Support Vector Machine-Recursive Feature Elimination (SVM-RFE). Attack detection is subsequently performed using DSNFyS, which integrates a Deep Stacked Autoencoder (DSA) with an Adaptive Neuro-Fuzzy Inference System (ANFIS). Upon detection of an attack, mitigation is carried out employing a DSA trained using the Hiking Optimization Algorithm (HOA). The proposed DSNFyS demonstrated exceptional performance, achieving the better accuracy of 97.41%, True Positive Rate (TPR) of 97.60%, and True Negative Rate (TNR) of 97.12%.

Index Terms: Internet of Things, Routing, Attack Detection, Deep Learning, Attack Mitigation

1. Introduction

The extensive integration of IoT networks into every aspect of our everyday lives has transformed our world into a transformative technological revolution in recent years. This has changed dramatically with the introduction of IoT networks, which have produced self-automated settings designed to facilitate the data transfer between processes. This has been crucial in influencing the Internet's capabilities [1]. Owing to the growing number of smart, ubiquitous, and connected devices, IoT systems are evolving quickly. IP-connected IoT is distributed across various domains, such as the e-health sector, smart homes, industrial cyber-physical systems, autonomous vehicles, and smart cities. It comprises a multitude of connected heterogeneous objects [2]. Allowing for smooth data sharing and automation, IoT has emerged as a new model that integrates various systems, sensors, and smart devices. Billions of people are expected to use IoT devices across a wide range of industries such as manufacturing, smart cities, healthcare, and transportation [3]. Through the Internet, physical objects can be connected to the virtual world, enhancing human lifestyle and enabling countless next-generation applications [4]. IoT networks are more susceptible to attacks owing to the wide range of connected devices, each of which may have different processing speeds, storage capacities, and security measures. The

wireless communication techniques of the devices are vulnerable to eavesdropping and interference, which may compromise private information and sensitive data. IoT network security is complicated owing to the different sensors and protocols used [3].

Low-Power and Lossy Networks (LLNs) have limited processing, storage, and energy resources. RPL is a protocol designed for such networks [1]. Routing protocol security is essential owing to the widespread use of RPL in IoT applications. However, the introduction of RPL-based IoT has resulted in major security vulnerabilities that must be fixed, making it imperative to solve these problems in RPL-based IoT [5]. RPL can easily be optimized to meet the network requirements of any IoT deployment owing to its versatile design and customizable functionality[6]. Moreover, IoT security issues cannot be resolved by the most recent security laws that only address data encryption, identity-authenticated key agreements, digital signatures, and spam detection. These techniques have a coercive effect on device performance and utilize numerous IoT resources. Furthermore, the internal authentication of nodes renders cryptographic systems vulnerable to external threats [7]. IoT security solutions are insufficient, which has led to a number of disruptive attacks recorded in recent years. Furthermore, the rapid development and expansion of IoT will raise future security threats, risks, and impact of attacks [8]. Thus, more significant and practical improvements in RPL security are required to improve its suitability for practical IoT applications [6]. Attackers are now more interested in IoT devices than ever before owing to their increasing popularity. For an IoT environment to function properly, new security vulnerabilities must be addressed [9].

Version number attacks, blackhole, greyhole, wormhole, and rank decrease attacks are a few types of these attacks [9]. The most harmful types of attacks are sinkhole and Sybil attacks, which can disrupt and avoid network-device connections [7]. Therefore, creating more effective protection against internal routing attacks is essential to increase the resilience of RPL networks [6]. Machine Learning (ML) algorithms are used to track network behaviour, categorize network traffic, detect and prevent network intrusions, etc. However, in the IoT, these intelligent security models present memory and computational power challenges because they require considerable memory and processing power to function [5], [10]. By first learning the behavior of nodes and then identifying normal and abnormal conduct, Deep Learning (DL) is thought to be an efficient technique for solving various types of problems [11]. The accuracy and effectiveness of IoT attack detection schemes increase when Convolutional Neural Networks (CNNs) are used to choose feature sets that aid in the detection of IoT attacks[12]. Attack detection requires a CNN to identify and prioritize critical features [3]. Reinforcement Learning (RL), cloud-based security mechanisms, cross-layer intrusion detection, IoT data analysis, and ML are modern solutions that rely on DL techniques [9]. Intrusion detection using DL has proven to be a promising method for RPL-IoT attack detection, owing to its detection accuracy [2]. Massive heterogeneous IoT data can be handled by DL algorithms, which are capable of identifying advanced attacks with high attack detection efficiency [4].

Existing attack detection systems in RPL-based IoT networks face unique challenges, including limited resources, complexity of the RPL protocol, scalability issues, and a lack of standardization. The dynamic nature of IoT networks, limited visibility, and potential for false positives and negatives further complicate detection efforts. Additionally, sophisticated attackers may employ evasion techniques, and network congestion can affect the detection system performance. To overcome these challenges, innovative solutions such as machine learning-based detection, distributed detection systems, anomaly based detection, and collaborative detection are being explored to improve the detection accuracy and adapt to evolving threats.

A novel DSNFyS is proposed in this work for attack detection in RPL-based IoT. IoT was simulated using the RPL routing protocol. After routing, attack detection is performed at the Base Station based on the log file data. Here, the input log data is taken from specified database [13] to train model. Then, the min-max normalization technique is used to perform data normalization to convert data into a structured format. Thereafter, feature selection was performed by employing SVM-RFE and information gain to identify the most relevant features. Subsequently, attack detection is performed by exploiting DSNFyS, which is the unification of DSA and ANFIS. Finally, attack mitigation is performed when an attack is detected by employing DSA, which is tuned using the HOA.

The proposed DSNFyS algorithm offers several improvements over the existing algorithms for attack detection and mitigation in RPL-based IoT networks. By integrating DSA and ANFIS, DSNFyS achieves improved accuracy and effective mitigation. The algorithm's feature selection process, using SVM-RFE and information gain, helps to identify the most relevant features, leading to robust detection performance. Additionally, DSNFyS is more scalable, flexible, and reduces the number of false positives compared with existing algorithms. Overall, the DSNFyS provides a more effective and efficient solution for attack detection and mitigation in RPL-based IoT networks.

The primary contributions of this research are listed below.

- ***Proposed DSNFyS for Attack Detection in RPL-based IoT:*** This paper proposes an effective technique for attack detection in RPL-based IoT. Attack detection is carried out by exploiting DSNFyS, which is a unification of DSA and ANFIS.
- ***Proposed DSNFyS+DSA_HOA for attack mitigation in the RPL routing protocol:*** An effective technique was introduced for attack mitigation in the RPL routing protocol. Attack mitigation is performed when an attack is detected using DSA, which is trained using HOA.

The remainder of this paper is organized as follows. Section 2 provides a detailed description of the current approaches and the challenges encountered. Section 3 presents the system model of IoT. Section 4 discusses the DSNFyS model that was developed for attack detection. The results for the introduced DSNFyS and DSNFyS+DSA_HOA are discussed in Section 5. Finally, Section 6 concludes the paper.

2. Motivation

RPL is extensively utilized in low-power and resource-constrained networks such as IoT environments. High loss rates, limited bandwidth, and energy constraints are characteristics of networks in which RPL is intended to support routing. RPL is susceptible to different types of security attacks because of the resource constraints of IoT devices, although it works well in these scenarios. Although the RPL protocol is effective at controlling and maximizing routing in limited settings, it has serious safety problems that may harm network performance and integrity. To solve this problem, an efficient technique called DSNFyS and DSNFyS+DSA_ HOA was introduced for attack detection and mitigation in the RPL routing protocol using DL.

2.1 Literature survey

Albishari et al. [14] devised a DL-based early stage detection (DL-ESD) model for routing attacks in IoT networks. This approach performed better, was lightweight, had a lower error rate, and better detection efficacy. Unfortunately, the computational complexity and cost of this method are extremely high. A Security, Mobility, and Trust-based (SMTrust) model was developed by Muzammal et al. [8] to provide security in IoT against RPL attacks. This approach offers more flexibility, better scalability, a lower packet loss rate, high throughput, secure communication, dependability, and trustworthiness. Nevertheless, this approach had a significant delay and was unable to reduce the power consumption. A Multi-Mobile agent-based trust framework for Mitigating-RPL (MMTM-RPL) was developed by Farooq et al. [7] to mitigate internal attacks and enhance RPL security. This technique reduces the processing time, increases the storage capacity, and consumes less energy while improving the RPL security. However, real-time scenarios are not applicable to this method. A Collaborative and Distributed security scheme for RPL (CDRPL) was developed by Alsukayti et al. [6] to mitigate RPL version-number attacks in IoT networks. This approach requires less energy, has no extra entities, is scalable and secure, and requires less communication overhead to maintain network stability. However, the execution of a physical testbed with various IoT device types running real data traffic was unsuccessful using this approach.

An Intrusion Detection System (IDS) for RPL-based IoT networks was developed by Ribera et al. [9] . This technique allows rapid and high-accuracy detection of version number, blackhole, greyhole, and flooding attacks. Nevertheless, this approach does not automatically extract accurate data regarding intrusions from their observed characteristics. A Trust-based Hybrid Cooperative RPL protocol (THC-RPL) was introduced by Arshad et al. [15] in an RPL-based IoT network to identify malicious Sybil nodes. This technique easily detects malicious nodes and enhances the lifespan of the IoT network. However, this approach was unable to assess the THC-RPL in an actual testbed setup. A moth-flame Optimization-based secure method for RPL (MFO-RPL) was devised by Seyfollahi et al. [16] to improve the routing procedure and rank attack detection in RPL. This technique has fewer lost packets, less rank switching, and less convergence time. However, this approach is not suitable for other common RPL attacks, such as replay, version, and Destination Oriented Directed Acyclic Graph (DODAG) inconsistency attacks. To detect RPL attacks, Bokka and Sadasivam [10] established a DL-based Gated Recurrent Unit (GRU) model. This approach was better, with a lower rate of false positives, accurate prediction, and stability. However, this model does not investigate the application of GRU models to multiclass classification.

2.2 Major challenges

The subsequent are the main difficulties with current attack detection in the RPL-based IoT.

- MMTM-RPL [7] improved RPL security, extended network lifetime, and reduced internal attacks in IoT-based Wireless Sensor Networks (WSNs). Unfortunately, this method was unable to handle complex attack patterns for security-critical applications.
- CDRPL was devised in [6] to mitigate RPL version-number attacks in IoT networks. Here, this model had less communication complexity and efficiently maintained network stability. However, this approach was unable to identify the necessary characteristics for managing complicated data connections in attack detection.
- In [15], the THC-RPL technique was established to identify malicious Sybil nodes. In this case, the approach increases network lifetime and consumes less energy. However, this approach does not consider the quantitative measurement of attack detection features to improve the network security.
- MFO-RPL [16] used a trust mechanism and next-hop node selection to reduce the impact of rank attacks. Nevertheless, this model did not consider attack severity based on the observed network behaviors and activities.

- The need for security solutions in the IoT domain is highlighted by the exponential growth in the number of smart devices. Nonetheless, full support for routing security remains an unresolved issue in the RPL design. Some studies have ignored routing attacks and security issues in favor of concentrating only on routing behavior and network performance. In addition, trust-based network security solutions do not assess critical security threats, particularly routing attacks in RPL.

3. System Model of IoT

Smart tools exchange data and process data in the IoT. The number of IoT devices and applications is increasing rapidly [8]. However, end users face challenges when implementing IoT, mainly because of security and privacy problems. In an IoT network, information exchange is crucial but is also vulnerable to hacking attacks. The Internet Engineering Task Force (IETF) established the Internet Protocol version 6 (IPv6) Routing Protocol for RPL to efficiently address the routing needs of IoT. Owing to the correlation between the IoT and WSN fields, a significant number of routing attacks, such as RPL attacks and threats acquired from WSNs, are certainly probable in RPL. The requirements and necessities of IoT methods and services make it challenging to secure a network against routing threats in an IoT network. Considering the traditional use of RPL in the smart world, it is important to address security risks. The structure of the RPL network is illustrated in Fig. 1.

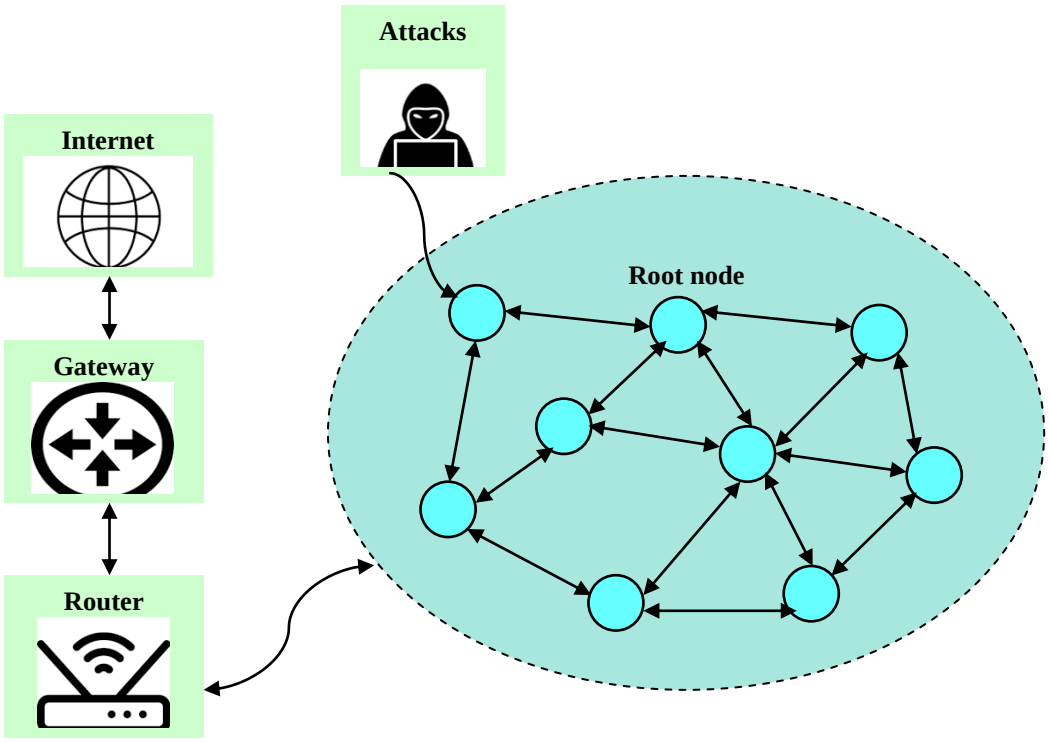


Fig. 1. System model of RPL network

3.1 Energy model

It is essential to assess the energy used in each node during packet processing when implementing energy-efficient routing protocols [17]. This energy encompasses the power required to transmit, receive, or forward a packet along a designated path. In addition, the node requires energy to listen to incoming packets. The Media Access Control (MAC) sublayer manages the transitions of nodes into their respective operational modes.

For the node u , the residual energy is stated below,

$$E_{res}^{u} = E_{ini}^{u} - E_{con}^{u} + E_{har}^{u} \tag{1}$$

From the above equation, E_{con}^{u} specifies the energy consumption of the u^{th} node, E_{res}^{u} exemplifies residual energy, E_{ini}^{u} signifies initial energy and E_{har}^{u} indicates harvested energy.

3.2 Trust model

The security of the network depends on trust models because it allows secure data transfer between parties without direct contact. When calculating trust, factors such as direct trust, indirect trust [18] and beta distribution [19] are considered. Furthermore, witness trust [20], availability factors and integrity factors [21] were considered.

The ensuing is a calculation of total trust,

$$T_{s,t} = \frac{1}{7}\{D_{st}(\xi) + ID_{st}(\xi) + H_{st}(\xi) + BD_{st} + W_s + AF_{st}(\xi) + IF_{st}(\xi)\} \quad (2)$$

Here, time is denoted as ξ, direct trust is specified by $D_{st}(\xi)$, the beta distribution is indicated as BD_{st}, integrity factor is designated as $IF_{st}(\xi)$, historical trust is exemplified as $H_{st}(\xi)$, witness trust is represented as W_s, total trust is denoted as T_{st}, indirect factor is signified by $ID_{st}(\xi)$, and available factor is denoted as $AF_{st}(\xi)$.

4. Proposed Deep Stacked Neuro-Fuzzy System (DSNFyS) for attack detectionin RPL-based IoT

A novel method for attack detection in the RPL-based IoT was proposed in this study. First, IoT is simulated using the RPL Protocol. Subsequently, attack detection is processed at the BS concerning log data. In this case, the input log data were obtained from a given database to train the model. Further, data normalization is carried out by employing the min-max normalization method [22] to convert the data into a structured format. Next, feature selection is carried out by exploiting SVM-RFE [23] and information gain [24] to find the most relevant features. Subsequently, attack detection is performed by utilizing DSNFyS, which is designed by integrating DSA [25] and ANFIS [26]. Finally, attack mitigation is performed when an attack is detected by employing DSA [25] , which is tuned by exploiting HOA [27]. Fig. 2 shows a schematic view of the DSNFySfor attack detection in the RPL-based IoT.

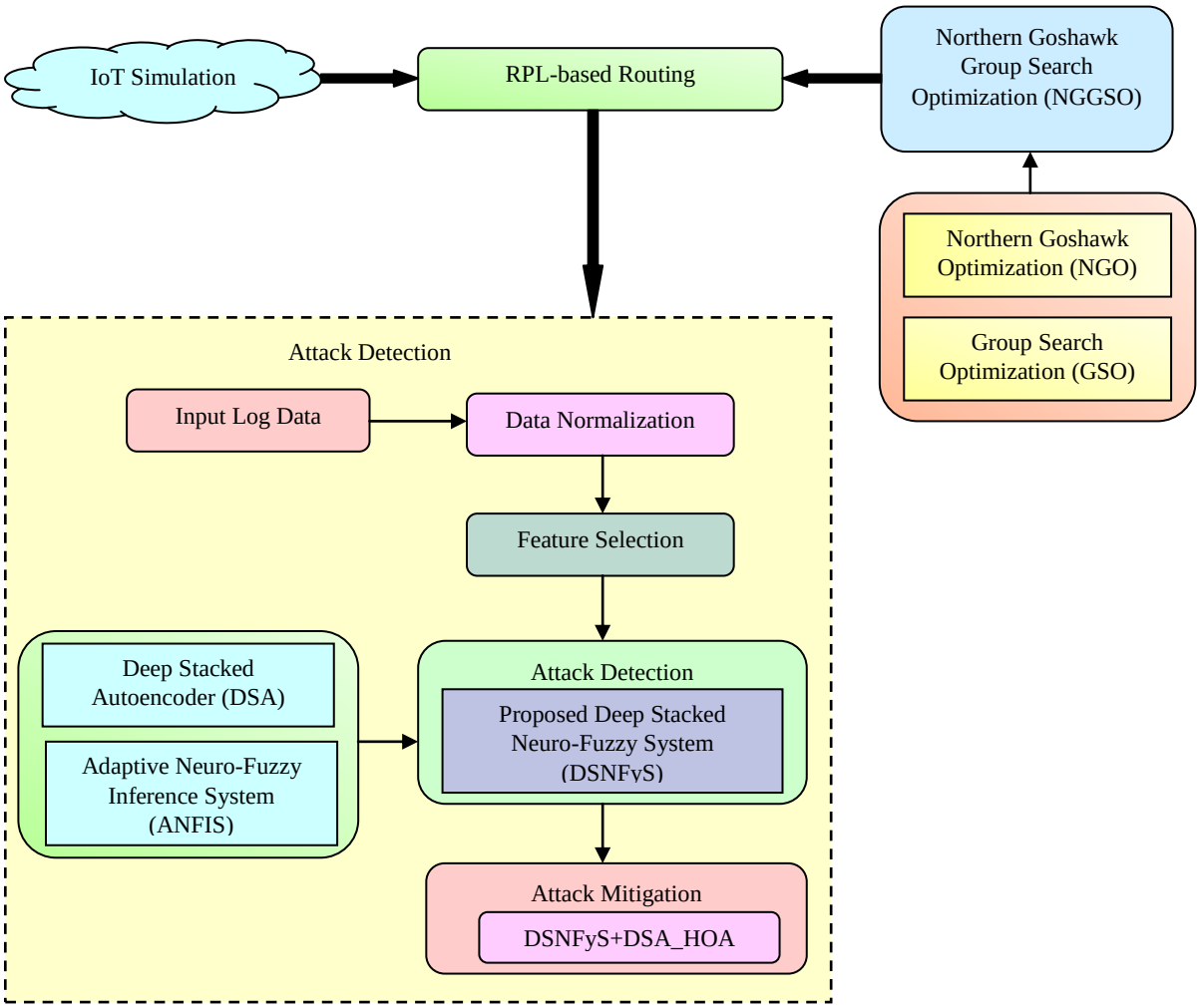


Fig. 2. Systematic representation of DSNFySfor attack detection in RPL-based IoT

4.1 RPL-based routing

In RPL routing protocols, the routing process is effectively managed using DODAG, in which the nodes are organized as a directed acyclic graph. Thus, the reliability of the data communication is ensured. The subsequent procedures are used in the RPL to construct the DODAG structure, and the network topology is considered first. Subsequently, node energy and numerous trust factors, including beta distribution, witness, integrity, availability, historical, recent, recommend, indirect, and direct trust, are considered together with delay and distance to measure the rank of the node in the multi-objective route discovery stage. The nodes are then selected for DODAG using the Northern Goshawk Group Search Optimization (NGGSO), which was created by combining Northern Goshawk Optimization (NGO) [28] and Group Search Optimization (GSO) [29], with the previously stated parameters being taken into account for estimating fitness. The next step is to create an ideal DODAG graph and establish a data-transmission path. Finally, the routing algorithm pattern allows the data to be forwarded.

4.1.1 Topology configuration

Network topology was established during the configuration phase after the IoT was simulated [30]. Every node receives a broadcast of the DODAG Information Object (DIO) packets from the root node. In the routing table, the root node is added as the parent node for each node that obtains a DIO message from the sender. Local node addresses are stored in the routing table and can be used for future usage, either upward or downward.

4.1.2 Multi-objective route discovery stage

The information found in the DIO messages is used to determine the rank of the node during the route-discovery phase. The DIO message's preferred parent is chosen to be the DIO sender if the determined rank is higher than the rank within the message [30]. To join the network, a new node must send a DODAG Information Solicitation (DIS) message requesting the DIO of its neighbors. Every neighborhood node that receives the DIS message is thus required to send a DIO to compute its rank in a DODAG. Trust, distance, delay, and energy were the factors considered when calculating the rank.

4.1.3 Optimal node selection for DODAG graph

The NGGSO technique is used to determine the optimal node that leads to the optimal path from a DODAG source to the destination. The path from the root node to the sink node in the DODAG can be found by storing a small portion of the forwarding information of the parent node and routing table because RPL routing is used in this instance. NGGSO was introduced by incorporating GSO [29] and NGO [28]. The NGO is a swarm-based algorithm formulated concerning northern goshawks and operates in two stages: the tail and chase processes and prey identification. This algorithm simply handles real-world problems. Animal characteristics serve as inspiration for the GSO, especially the way animals search for things. The method operates according to the mathematical formulation of the animal scanning process and exhibits robust convergence. A better exploration ability is achieved when NGO and GSO are combined. The following details the final updated equation of the NGGSO.

$$l_v(Y+1) = \frac{1}{C(2Q-1)}\left[\kappa_v o_v(\mathrm{I}(Y+1))(1+C(2Q-1)\right] \tag{3}$$

Here, Y indicates iteration count, radius is designated as C, random number ranging in [0,1] is demonstrated by Q, distance vector is indicated by $\boldsymbol{K}$, the direction vector is signified by $\boldsymbol{O}$, $l_v(Y+1)$ indicates location of v^{th} northern goshawk at $(Y+1)^{th}$ iteration, and head angle is specified by I.

4.1.4 Construct the optimal DODAG graph

The DODAG graph was used to establish data transmission. It is optimally constructed based on the nodes specified by the NGGSO.

4.1.5 Data Transmission

The best DODAG produced was used to transfer the data from the source to the destination. Finally, RPL routing for IoT is used for data forwarding.

4.2 Attack detection

Attack detection is the process of locating anomalies or malicious activity in a network or computer environment and taking the necessary measures. Identifying indications of security breaches, unauthorized access, or other harmful activities involves tracking and evaluating network traffic, system behavior, and other data sources. Here, the input log file data are primarily allowed for data normalization by exploiting min-max normalization [22]. Furthermore, the features are selected from the normalized data by employing information gain [24] and SVM-RFE [23]. Finally, attacks

were detected from the selected features using DSNFyS. Furthermore, the detection procedure performed is detailed as follows,

4.2.1 Log file acquisition

The input log file data is obtained from the BoT-IoT dataset [13], which includes log data and IP address. The log information is labeled as,

$$\omega = \left[\omega_1, \omega_2, ..., \omega_e, ..., \omega_m\right] \tag{4}$$

Here, ω denotes the BoT-IoT database, overall count of log files is represented by m, and e^{th} log file contemplated for attack detection is specified by ω_e.

4.2.2 Data normalization

Min-max normalization [22] is an important technique for improving model performance by standardizing the scales of features. This ensures that all features contribute equally to the learning process, preventing larger values from overshadowing smaller ones. Additionally, min-max normalization helps speed up the convergence of optimization algorithms, such as gradient descent, allowing models to train more quickly and effectively. It also makes models more robust against outliers, reducing their influence on predictions and leading to more stable results.

Data normalization is usually performed to clean up and organize the data in the input data ω_e to assist in effective decision-making. In this work, Min-Max normalization [22] is exploited for normalizing the data. Moreover, min-max normalization is simple to calculate and understand, making it available for several applications. Using a linear transformation to compare the value of input data ω_e before and after the procedure, the Min-Max normalization method balances the actual data. The Min-Max normalization is articulated by,

$$\delta_N = \frac{\omega_e - min(\omega_e)}{max(\omega_e) - min(\omega_e)} \tag{5}$$

In this case, maximum value of the attribute is indicated by $max(\omega_e)$ and minimum value is signified by $min(\omega_e)$. In addition, resultant normalized outcome acquired while normalizing input data ω_e by employing Min-Max normalization is represented by δ_N, which is further passed to feature selection phase.

4.2.3 Feature selection

In order to remove unnecessary data from the log data file and enable accurate detection, feature selection is performed. Here, information gain [24] and SVM-RFE [23] were used to select pertinent features from the normalized log data file δ_N.

-SVM-RFE

The SVM-RFE [23] algorithm determines the signature attributes by generating a ranking coefficient based on the weight vector φ that SVM generates during training. The signature feature with minimum ranking coefficient after every iteration is eliminated. Additionally, the features of the normalized data are eliminated utilizing SVM-RFE algorithm to generate new feature set.

Consider training samples $\{h_n, i_n\}$, $i_n \in \{-1, +1\}$ and μ signifies the output feature ordering set.

a) Set the initial feature set $\varsigma = 1, 2, ..., \mathrm{h}$; feature organized set $\mu = \varnothing$.

b) Repeat these steps until $\varsigma = \varnothing$.

i) Obtain the candidate feature set along with training set

ii) Get φ by training SVM classifier.

iii) Compute ranking norms score:

$$U_N = \varphi_N^2, \qquad N = 1, 2, ..., |\varsigma| \tag{6}$$

iv) Recognize the features with lowest ranking criteria score.

$$\arg\min_N U_N \tag{7}$$

v) Apprise feature set $\mu = \psi \cup \mu$, ψ indicate the features with the lowest ranking.

vi) Eradicate this feature in ς so as $\varsigma = \varsigma / \psi$.

The outcome acquired from SVM-RFE is denoted by f_1.

-Information gain

Information gain is a measurement of the amount of information obtained from normalized log file data based on a particular feature [24]. Information gain was calculated by comparing the entropies of the original set of data and the two child sets. Higher information gain indicates a feature that splits the data more effectively, and the formula of information gain is articulated below,

$$L(P,k) = \chi(P) - \chi(P \mid k) \tag{8}$$

Here, P represents class label, k indicates feature, class label's entropy is designated by $\chi(P)$, information gain is denoted as L, $\chi(P \mid k)$ represents the possibility of class P for a given feature k. Moreover, the features selected with information gain is signified by f_2.

The most relevant features are obtained by combining the outputs f_1 and f_2 selected using SVM-RFE and information gain. The selected feature is articulated as,

$$S = f_1 \cup f_2 \tag{9}$$

wherein, S specifies selected features.

4.2.4 Attack detection using DSNFyS

The protection of networks and information systems from malicious and unauthorized activities requires effective attack detection. Its efficacy improves an organization's overall cybersecurity while having a direct impact on data integrity, system availability, and regulatory compliance. The DSNFyS technique was devised in this study to detect attacks in RPL-based IoT. By combining DSA [25], [31] and ANFIS [26]], DSNFyS was presented.

The three components encompass the DSNFyS model, namely, the DSA method, DSNFyS layer, and ANFIS method. Additionally, the DSNFyS model's detection performance is improved by using the Fractional Calculus (FC) [32] for regression modeling. The input log file data ω_e is fed into DSA technique to obtain the outcome J_1 during attack detection. Following this, the outcome J_1 along with selected feature S is fed to the DSNFyS layer to attain the outcome J_2. Thereafter, the outcome J_2 acquired from DSNFyS layer is fed to ANFIS for attaining attack detection outcome J_3. Moreover, the schematic representation of DSNFyS for attack detection in RPL routing protocol is depicted in Fig. 3.

a) DSA model

The DSA model [25], [31] is a type of Deep Neural Network (DNN) that consists of multiple autoencoder layers, each of which is trained using the output of the layer before it. The relationship between independent and dependent features is also described by autoencoders, which are used to efficiently reduce the dimension of the data. DSA is regarded as an unsupervised learning technique that typically obtains an output value using a backpropagation method. Encoders and decoders are two broad categories in which the autoencoders can be separated. In this case, the encoder is employed to perform dimensionality reduction by executing the encoding process, whereas the decoder reconstructs the original data from the reduced representation through the decoding process. As a result, anomalies with high reconstruction errors in the data can be precisely identified using DSA.

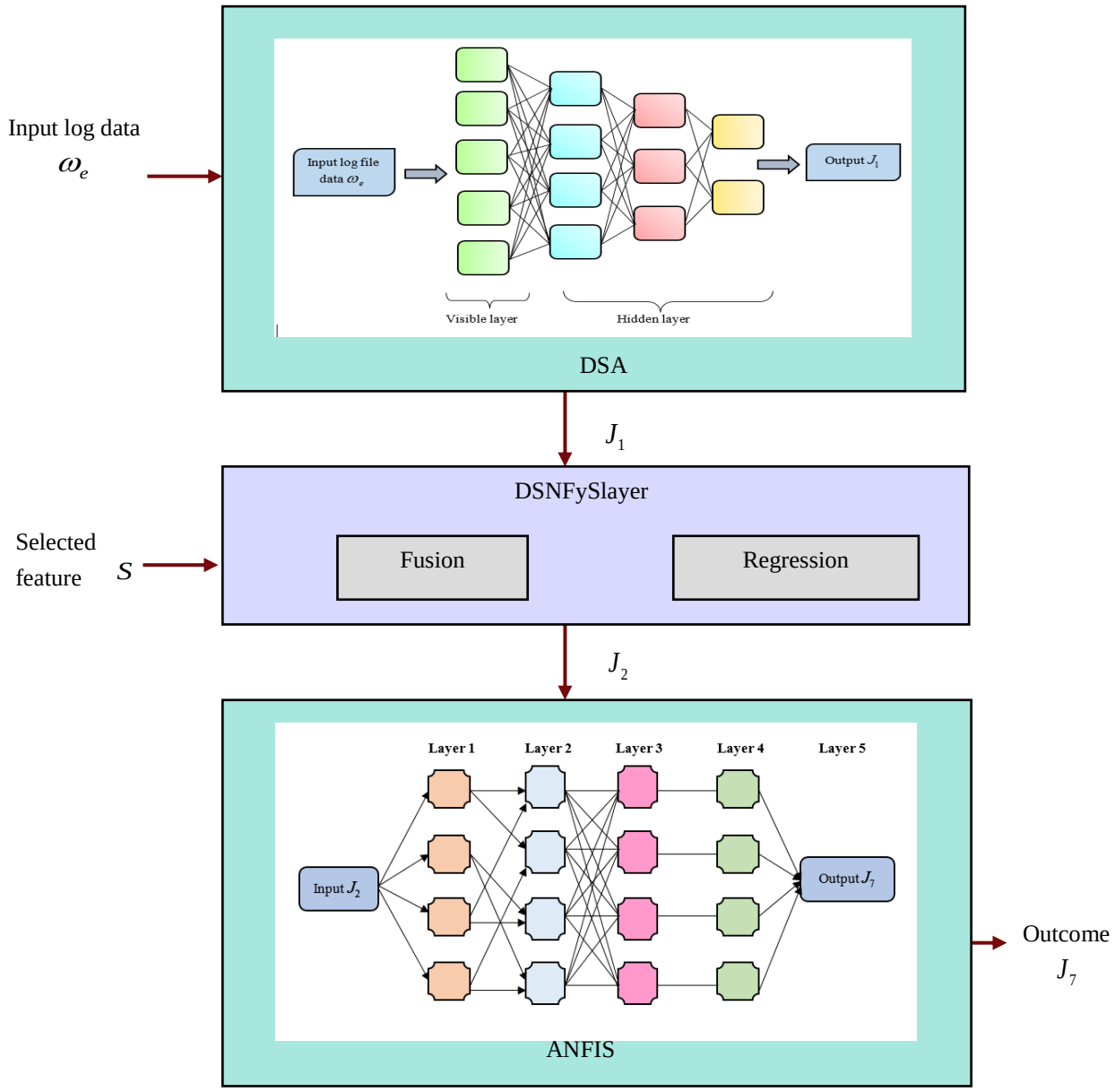


Fig. 3. Pictorial representation of DSNFyS for attack detection

The DSA encoder encompasses a mapping function, a hidden layer, and an input layer. The hidden layer's outcome is articulated below,

$$z(\omega_e) = \vartheta(\eta\omega_e + I) \tag{10}$$

Here, activation function is specified by $\vartheta(.)$, η represents encoding weight matrix, bias vector is exemplified by I and input log file data is signified as ω_e. Likewise, the decoder alters the hidden layer, mapping function and output layer. The decoder uses a mapping function to transform hidden layer outcome into input layer. The outcome that is produced as a result of the reconstructed data $\hat{\omega}$ is stated as,

$$\hat{\omega}_e(w) = r(\eta^* w + I^*) \tag{11}$$

In the above equation, I^* signifies decoded bias vector, decoding weight matrix is represented by the term η^*, reconstructed input data is demonstrated as $\hat{\omega}_e$, $r(.)$ exemplifies decoder activation function. In addition, Mean Square Error (MSE) is utilized to evaluate reconstruction effect of autoencoder, and is articulated below,

$$MSE = \frac{1}{2}\left\|\omega_e - \hat{\omega}_e\right\|^2 \tag{12}$$

Typically, the DSA model is trained through two stages such as unsupervised layer-by-layer pre-training and supervised reverse fine-tuning. The learning accuracy and rate of the model were improved by these stages. The initial operation is dropout, which involves averaging predictions from multiple model instances to ensure that the weight updates are no longer influenced by fixed interactions among neurons. The second strategy for addressing overfitting involves the incorporation of a weight decay term into the loss function.

A subnetwork of the original network is created in a dropout operation by random discarding the neurons in a hidden layer with probability G_b. The Bernoulli distribution is confirmed by the discarded matrix $p = \left[p_1, p_2, p_3, ..., p_\beta\right]$. The function that is produced as a consequence of probability distribution is articulated below,

$$\varpi\left(d, G_b\right) = \begin{cases} G_b, & d = 1 \\ 1 - G_b, & d = 0 \end{cases} \tag{13}$$

wherein, possible outcome is represented by d. Following the updating of the mapping function at the original neural network, the dropout operation is then executed in hidden layers, as follows:

$$J_1 = p_y \vartheta\left(\sum_{\iota=1}^{q} \eta\omega_e + I\right) \tag{14}$$

$$J_1 = \begin{cases} \vartheta\left(\sum_{\iota=1}^{q} \eta\omega_e + I\right), & p_y = 1 \\ 0, & p_y = 0 \end{cases} \tag{15}$$

In the above equation, p_y signifies discarded matrix and y denotes amount of elements. Further, the overfitting problems are also addressed by including weight decay coefficient in the loss function. Furthermore, Categorical Cross Entropy (CCE) function is the existing loss function for the classification of multilabel. The network structure of DSA method is illustrated in Fig. 4. Also, resultant outcome J_1 is achieved while subjecting input log file data ω_e into DSA framework.

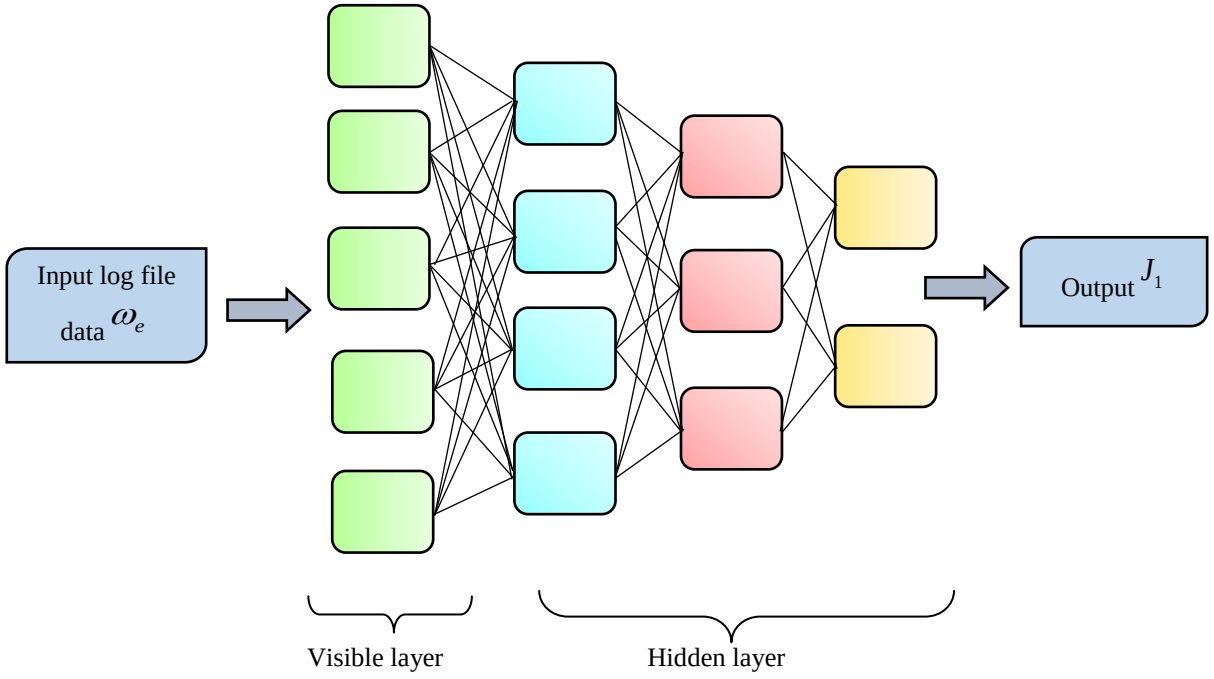


Fig. 4. Pictorial view of DSA

b) DSNFyS layer

The selected feature S and the DSA model's outcome J_1 are fed into the DSNFyS layer to identify the relationship among the inputs by involving regression modeling to the system and the regression is carried out by employing FC [32]. FC provides solutions for a range of integrals and derivatives using Laplace transform. Moreover, inverse Laplace transform is used to get the desired solutions. The output of the DSNFyS layer is produced by contemplating weighted features at several time intervals. Consequently, at F^{th} interval the output of DSNFyS layer is demonstrated below,

$$B_1 = \sum_{c=1}^{V} (S)_c * a_c \tag{16}$$

Here, V denotes feature size and a represents weight coefficient. Furthermore, at $(F-1)^{th}$ time interval, DSA model outcome J_1 is considered to produce DSNFyS layer outcome.

Lastly, regression modeling is done for the fusion of the inputs acquired at different intervals using FC [32]. According to FC,

$$\varepsilon(F+1) = \gamma \cdot \varepsilon(F) + \frac{1}{2}\gamma \cdot \varepsilon(F-1) \tag{17}$$

Here, γ indicates order of derivatives. Consider, $\varepsilon(F+1) = J_2$, $\varepsilon(F) = B_1$ and $\varepsilon(F-1)$. Furthermore, the final outcome from DSNFyS layer is articulated beneath,

$$J_2 = \gamma \cdot B_1 + \frac{1}{2}\gamma \cdot J_1 \tag{18}$$

The outcome J_1 from DSA and the value of B_1 from equation (16) are substituted in equation (18). Thus, above equation can be rewritten as,

$$J_2 = \gamma \cdot \left[\sum_{c=1}^{V} (S)_c * a_c \right] + \frac{1}{2}\gamma \cdot p_y \vartheta \left(\sum_{\iota=1}^{q} \eta \omega_e + I \right) \tag{19}$$

Afterwards, the resulting DSNFyS layer outcome J_2 is fed into the ANFIS framework for attack detection.

c) ANFIS model

The output J_2 from the DSNFyS layer is included with the ANFIS [26] to detect attacks.The ANFIS technique combines the concepts of Artificial Neural Network (ANN) and Fuzzy Interference System (FIS). Designing a non-linear system was made feasible by the ANFIS's capacity to adapt and learn. Because of its fast-learning capabilities, ANFIS improves parameter optimization and can solve complicated problems more effectively. Here, output from DSNFyS layer J_2 is fed as input, and ANFIS functions based on two fuzzy rules, which are articulated as beneath,

Rule-a): If υ is λ_1 and A is X_1, then $\Omega_1 = [\sigma_1 \upsilon + \tau_1 A + O_1]$

Rule-b): If υ is λ_2 and A is X_2, then $\Omega_2 = [\sigma_2 \upsilon + \tau_2 A + O_2]$

Here, $\sigma_1, \sigma_2, \tau_1, \tau_2, O_1,$ and O_2 signifies linear parameters, fuzzy rule is indicated as Ω. Furthermore, λ_1, X_1, λ_2, and X_2 exemplifies membership function of inputs υ and A. Furthermore, the ANFIS architecture consists of five layers, which are described below.

Layer-1: The initial layer encompassed of square nodes η' in layer 1, also known as the e^{th} fuzzification layer. The following describes the layer-1 ANFIS technique operation,

$$R_{\lambda_{\eta'}}(\upsilon) = \exp\left[-\left(\frac{\upsilon - e_{\eta'}}{2g'_{\eta'}} \right)^2 \right] \tag{20}$$

where, $R_{\lambda_{\eta'}}(Z)$ indicates degree of membership function for fuzzy $\lambda_{\eta'}$, which is articulated beneath,

$$R_{\lambda_{\eta'}}(\upsilon)=\frac{1}{1+\left|\frac{\upsilon-e_{\eta'}}{2g'_{\eta'}}\right|^{2\phi}} \tag{21}$$

The following describes the outcome J, which is the representation of the input's membership grade:

$$J_{3,\eta'}=R_{\lambda_{\eta'}}(A'),\quad \eta'=1,2,3 \tag{22}$$

$$J_{3,\eta'}=R_{X_{\eta'}}(B'),\quad \eta'=3,4 \tag{23}$$

Let's consider, $A'=J_2$ and $B'=\delta_N$, the equation becomes,

$$J_{3,\eta'}=R_{\lambda_{\eta'}}\left(\gamma\cdot\left[\sum_{c=1}^{V}(S)_c*a_c\right]+\frac{1}{2}\gamma\cdot p_y\vartheta\left(\sum_{\iota=1}^{q}\eta\omega_e+I\right)\right),\quad \eta'=1,2,3 \tag{24}$$

$$J_{3,\eta'}=R_{X_{\eta'-2}}(\delta_N),\quad \eta'=3,4 \tag{25}$$

wherein, $\{e_{\eta'},g'_{\eta'},\phi_{\eta'}\}$ specifies set of parameters.

Layer-2:Each rigid node in this layer works by multiplying incoming signals and then sends the resulting product. The procedure executed within layer 2 can be articulated as follows,

$$J_{4,\eta'}=R_{\lambda_{\eta'}}\left(\gamma\cdot\left[\sum_{c=1}^{V}(S)_c*a_c\right]+\frac{1}{2}\gamma\cdot p_y\vartheta\left(\sum_{\iota=1}^{q}\eta\omega_e+I\right)\right)*R_{X_{\eta'-2}}(\delta_N),\quad \eta'=1,2 \tag{26}$$

Layer-3: The η'^{th} node computes the ratio of η'^{th} rule to the total firing strength. Additionally, an expression for layer 3 is shown below,

$$J_{5,\eta'}=\overline{\beta'_{\eta'}}=\frac{\beta'_{\eta'}}{\sum_{\eta'}\beta'_{\eta'}}\Omega_{\eta'} \tag{27}$$

In this case, $\beta'_{\eta'}$ specifies firing rule strength and thenormalized firing strength is represented by $\overline{\beta'_{\eta'}}$.

Layer-4: The η'^{th} node in layer-4 of ANFIS is a squared node that performs node function, and the formula is articulated beneath,

$$J_{6,\eta'}=\overline{\beta'_{\eta'}}\Omega_{\eta'}=\overline{\beta'_{\eta'}}\left[\sigma_{\eta'}\left(\gamma\cdot\left[\sum_{c=1}^{V}(S)_c*a_c\right]+\frac{1}{2}\gamma\cdot p_y\vartheta\left(\sum_{\iota=1}^{q}\eta\omega_e+I\right)\right)+w'_{\eta'}(\delta_N)+O_{\eta'}\right] \tag{28}$$

The parameter set is denoted by $\{\sigma_{\eta'},w'_{\eta'},O_{\eta'}\}$, which signifies the consequent parameter and variables in the node are exemplified by $\left[\sigma_{\eta'}\upsilon+w'_{\eta'}A+O_{\eta'}\right]$.

Layer-5: In this layer, the node processes all incoming signals and produces the resultant value as expressed as,

$$J_7=\sum_{\eta}\overline{\beta'_{\eta'}}\Omega_{\eta'}=\frac{\sum_{\eta'}\beta'_{\eta'}\Omega_{\eta'}}{\sum_{\eta'}\beta'_{\eta'}} \tag{29}$$

wherein, J_7 specifies ANFIS outcome. Fig. 5 provides the pictorial representation of the ANFIS technique.

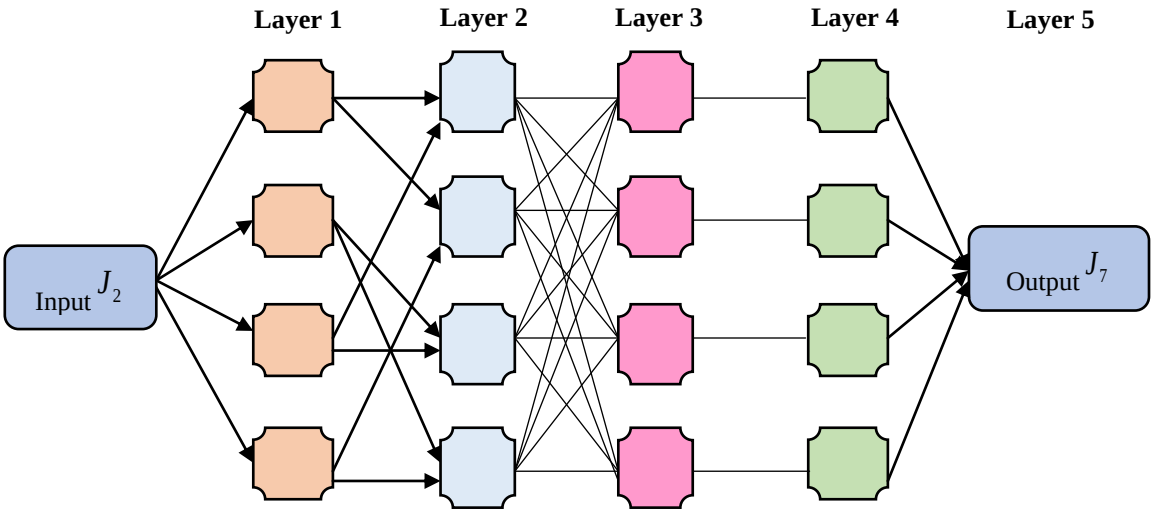


Fig. 5. Schematic view of ANFIS

4.2.5 Attack mitigation

Attack mitigation is the procedure of applying strategies and techniques to stop, lessen, or handle attacks on data, networks, and systems. Attack mitigation is primarily used to minimize damage in the event of an attack and to reduce the risk and impact of security incidents by defending against possible attacks. In this study, the impact of the detected attacks was minimized by employing the DSA_HOA model. In this case, the DSA [25] model was ideally tuned by exploiting HOA [27] . Moreover, the detected output J_2 is fed as an input to the attack mitigation process.

i) Architecture of DSA

The main applications of a DSA [25] neural network architecture are dimensionality reduction, feature extraction, and unsupervised learning. It expands the idea of autoencoders, which are neural networks trained to decode input data back into its original form after encoding it into a lower-dimensional representation. DSAs have the ability to decrease data complexity by compressing high dimensional data into a lower dimensional latent space. Finding patterns and anomalies that might be indications of an attack may become simpler as a result. Furthermore, the developed DSA model is elucidated in detail in section 4.2.4. The outcome acquired from DSA model is represented as D_m, which is the attack mitigation result.

ii) Training of DSA using HOA

An optimization method called HOA [27] is used to carry out the DSA training process. In this case, HOA is used to generate the classifier weights in order to produce the best consequence. HOA improve the performance of DSA by ideally training with HOA. Hiking is a popular recreational activity, and HOA takes inspiration from the mountainous terrain hikers traverse in their search landscapes for optimization problems. Tobler's Hiking Function is an exponential function used to model a hiker's velocity by accounting for the gradient or incline of the terrain. HOA maintains a good balance among exploration and exploitation of the solution space, which assists in finding optimal solutions effectually. The updated formula of HOA is stated beneath:

$$\alpha_{\zeta,x+1} = \alpha_{\zeta,x} + K_{\zeta,x} \tag{30}$$

where, hiker is denoted as ζ , hikers' location is signified by $\alpha_{\zeta,x}$, iteration is specified as x and present velocity of ζ^{th} hiker is represented as K .

-Fitness function

Fitness measure is defined as the difference among the DSA's projected and achieved outcomes, and it can be quantified using the subsequent expression:

$$fitness = \frac{1}{v}\sum_{m=1}^{v}\left[\mathrm{D}_m - \mathrm{D}_m^{*}\right]^2 \tag{31}$$

From the above equation, number of training samples is signified by V , expected outcome of DSA is represented as D_m^* and attained output is specified by D_m .

5. Results and Discussion

The experimental consequences measured by DSNFyS employed for attack detection and DSA_HOA for attack mitigation in in RPL-based IoT is deliberated in this section. Furthermore, the corresponding discussions of detection techniques are briefed beneath.

5.1 Experimental setup

The DSNFyS model devised for the attack detection in RPL routing protocol is executed by exploiting python tool using BoT-IoT dataset [13].

5.2 Dataset description

The BoT-IoT dataset [13] was created in the Cyber Range Lab of UNSW Canberra to simulate a realistic network environment with normal and botnet traffic. The tshark tool was utilized to create the raw network packets (Pcap files) of the BoT-IoT dataset. The packets consist of both normal and abnormal traffic. Ostinato tool and Node-red were used to generate simulated network traffic. The source files for the database are available in a variability of formats, contain original PCAP files, the generated Argus files, and the CSV format. It includes various attack categories such as DDoS, DoS, OS and Service Scan, Keylogging, and Data exfiltration, with over 72 million records in the original dataset (69.3 GB pcap files and 16.7 GB csv files). A 5% subset (1.07 GB, three million records) was extracted for easier handling [13].

5.3 Evaluation metrics

The efficiency of the DSNFyS technique in identifying attacks in the RPL based IoT is carried out using various assessment parameters, which are further described below,

a) Accuracy

It is the relationship among real and predicted output acquired by DSNFyS is referred to as accuracy, which is stated beneath,

$$Accuracy = \frac{j_{tp} + j_{tn}}{j_{tp} + j_{tn} + j_{fp} + j_{fn}} \tag{32}$$

In the above equation, j_{tn} signifies true negative, j_{fp} specifies false positive, true positive is exemplified as j_{tp} and j_{fn} represents false negative.

b) TNR

The percentage of real negative cases that the model precisely detects is measured by the TNR. Additionally, TNR is determined by applying the specified expression,

$$TNR = \frac{j_{tn}}{j_{tn} + j_{fp}} \tag{33}$$

c) TPR

The amount of true positives divided by the overall amount of cases that are actually positive yields the TPR. Furthermore, TPR is determined with the below expression,

$$TPR = \frac{j_{tp}}{j_{tp} + j_{fn}} \tag{34}$$

5.4 Comparative methods

This study devises a new model called DSNFyS for attack detection in RPL-based IoT, and the DSNFyS+DSA_HOA model for attack mitigation in RPL-based IoT. To assess the performance of DSNFyS and DSNFyS+DSA_HOA, these methods were compared with conventional methods, including MMTM-RPL [7], CDRPL [6] , THC-RPL [15] , and MFO-RPL [16].

5.5 Comparative analysis

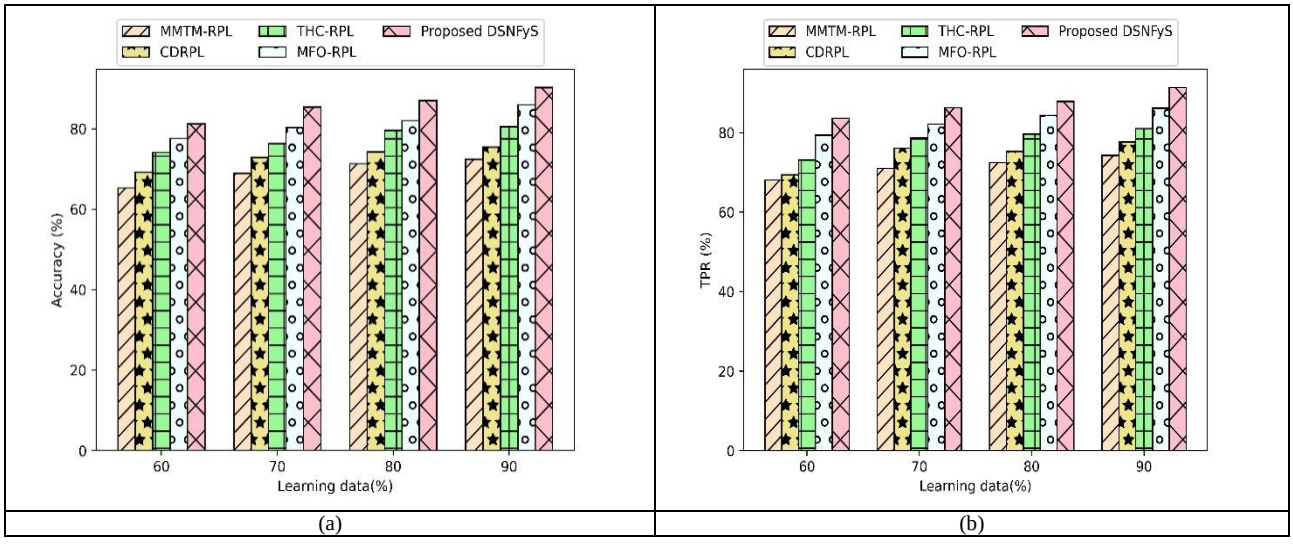
The evaluation is carried out by contemplating parameters including accuracy, TPR, and TNR by considering "with mitigation" and "without mitigation" scenarios.

5.5.1 Without mitigation

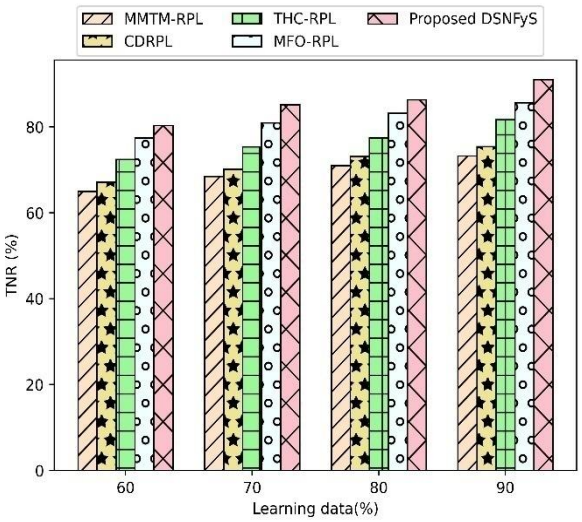
This evaluation concentrated on contrasting the performance of several techniques for attack detection in RPL-based IoT with the proposed DSNFyS concerning k-fold [33] and learning data without mitigation. The choice of k-value in k-fold cross-validation for analyzing deep learning models is crucial for reliable performance estimates. K-fold cross-validation involves splitting the dataset into k subsets, or folds, and systematically training the model on k-1 folds while validating it on the remaining fold. This process is repeated k times, ensuring each data point is used for both training and validation. In this research work k value is selected as 8, providing a good balance between computational efficiency and robust evaluation. A smaller k may lead to less reliable estimates due to larger training sets, while a larger k can yield more granular insights but increase computational time. The chosen k-value helps mitigate overfitting by ensuring diverse training and testing scenarios, leading to a more comprehensive understanding of the model's generalization capabilities.

-Based on learning data

The examination of DSNFyS approach considering learning data is depicted in Fig. 6. In Fig. 6a), the examination of DSNFyS method concerning accuracy is portrayed. The DSNFyS achieved an accuracy of 90.244%, while conventional schemes, like MMTM-RPL, CDRPL, THC-RPL and MFO-RPL measured accuracy of 72.456%, 75.490%, 80.557%, and 85.967%, with a learning data of 90%. This shows that DSNFyS computed an improved performance by 19.71%, 16.35%, 10.73%, and 4.74%. In Fig. 6b), the investigation of DSNFyS with respect to TPR is signified. The DSNFyS calculated TPR value of 91.345%, with 90% learning data, whereas TPR computed by MMTM-RPL is 74.316%, CDRPL is 77.648%, THC-RPL is 81.042%, and MFO-RPL is 86.135%. This shows that DSNFyS acquired a higher performance by 18.64%, 15.00%, 11.28%, and 5.70%. Fig. 6c) displays the analysis of DSNFyS contemplating the TNR metric. With 90% learning data, the classical techniques, namely MMTM-RPL, CDRPL, THC-RPL, MFO-RPL, and DSNFyS computed TNR of 73.240%, 75.436%, 81.679%, 85.642%, and 91.042%, which shows that DSNFyS generated a superior TNR by 19.55%, 17.14%, 10.28%, and 5.93%.



(a) (b)

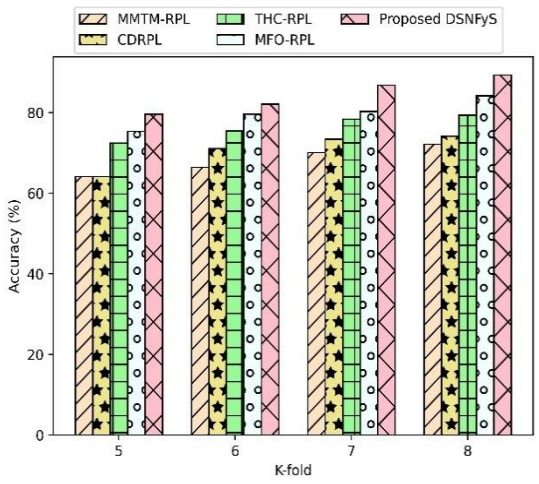


(c)

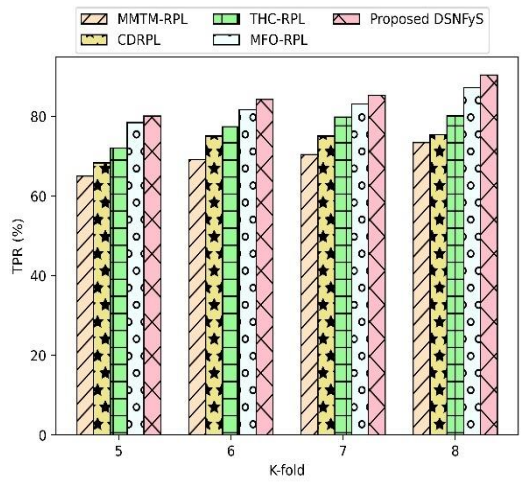
Fig. 6. Evaluation of DSNFyS altering learning data based on, a) accuracy, b) TPR, and c) TNR by assuming without mitigation

-Based on k-fold

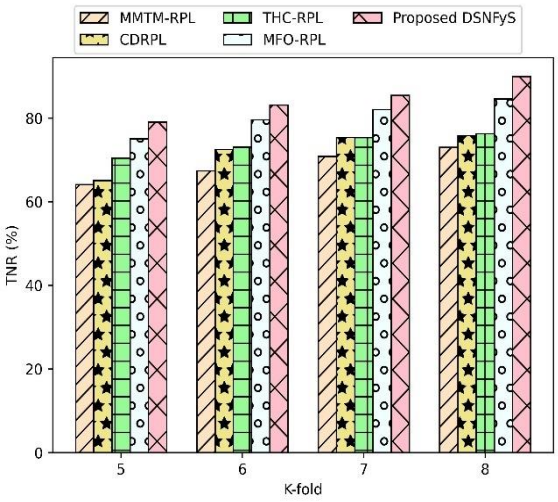
The DSNFyS approach is examined in contrast with classical attack detection systems concerning k-fold and this is signified in Fig. 7. The valuation of DSNFyS model concerning accuracy is depicted in Fig. 7a). The DSNFyS obtained a maximum accuracy of 89.345%, with k-fold of 8 whereas accuracy achieved by MMTM-RPL at 72.146%, CDRPL at 74.166%, THC-RPL at 79.350%, and MFO-RPL at 84.345%. This infers that a performance improvement of 19.25%, 16.99%, 11.19%, and 5.80% is computed by DSNFyS technique. Fig. 7b) denotes the evaluation of the DSNFyS with respect to TPR. The TPR achieved by the several attack detections approaches, including MMTM-RPL is73.455%, CDRPL is 75.488%, THC-RPL is 80.167%, MFO-RPL is 87.246% and DSNFyS is 90.424%, with k-fold 8. This shows that DSNFyS acquired a higher performance by 18.77%, 16.52%, 11.34%, and 3.51%. The examination of DSNFyS based on TNR is portrayed in Fig. 7c). The DSNFyS calculated TNR of 90.021%, which is superior by 18.89%, 15.75%, 15.21%, and 6.05% than TNR recorded by MMTM-RPL at 73.015%, CDRPL at 75.843%, THC-RPL at 76.325%, and MFO-RPL at 84.577%, with k-fold 8.



(a)



(b)



(c)

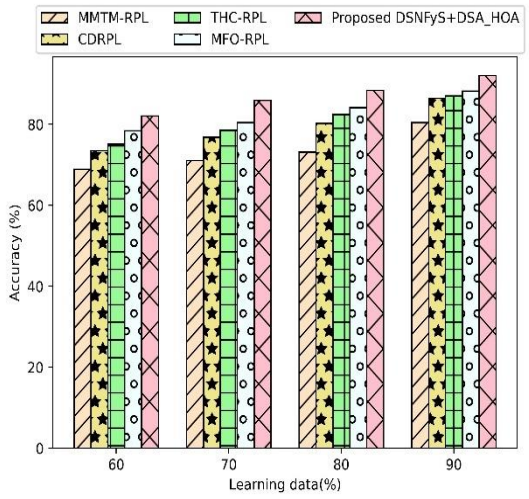
Fig. 7. Valuation of DSNFyS altering k-fold with a) accuracy, b) TPR, and c) TNR by taking into account without mitigation

5.5.2 With mitigation

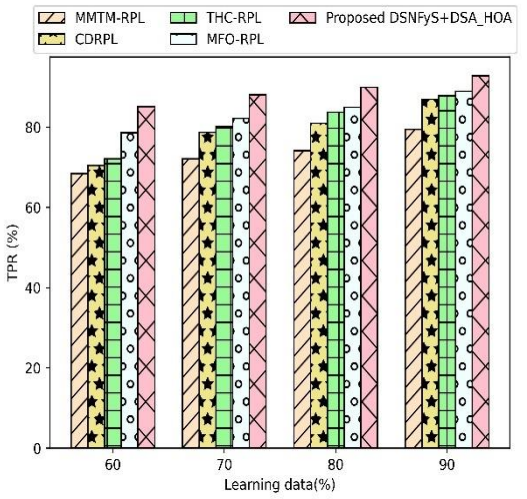
The valuation of the DSNFyS+DSA_HOA method in "with mitigation" scenario is effectuated using several learning data and k value.

-Based on learning data

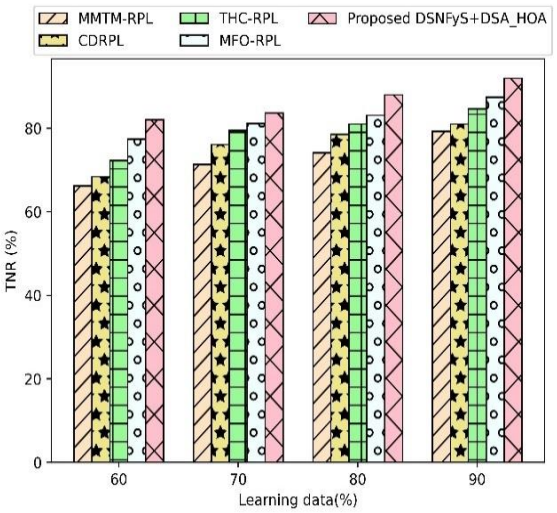
Fig. 8 shows valuation of DSNFyS+DSA_HOA technique with varying learning data. The accuracy-based evaluation of DSNFyS+DSA_HOA is portrayed in Fig. 8a). The DSNFyS+DSA_HOA model achieved an accuracy of 92.043%, while prevailing schemes, like MMTM-RPL, CDRPL, THC-RPL and MFO-RPL attained accuracy of 80.425%, 86.459%, 87.013%, and 88.216%, with a learning data of 90%. This displays that DSNFyS+DSA_HOA acquired an enhanced performance by 12.62%, 6.07%, 5.47%, and 4.16%. In Fig. 8b), the examination of the DSNFyS+DSA_HOA regarding TPR is illustrated. The DSNFyS+DSA_HOA calculated TPR value of 92.847%, with 90% learning data, while TPR recorded by MMTM-RPL is 79.466%, CDRPL is 86.958%, THC-RPL is 87.855%, and MFO-RPL is 88.976%. This shows that the DSNFyS+DSA_HOA gained a higher performance by 14.41%, 6.34%, 5.38%, and 4.17%. Fig. 8c) depicts the investigation of DSNFyS+DSA_HOA regarding TNR measure. With 90% learning data, the traditional techniques, such as MMTM-RPL, CDRPL, THC-RPL, MFO-RPL, and DSNFyS+DSA_HOA quantified TNR of 79.246%, 81.043%, 84.676%, 87.400%, and 92.014%, which shows that DSNFyS+DSA_HOA generated a superior TNR by 13.88%, 11.92%, 7.97%, and 5.01%.



(a)



(b)

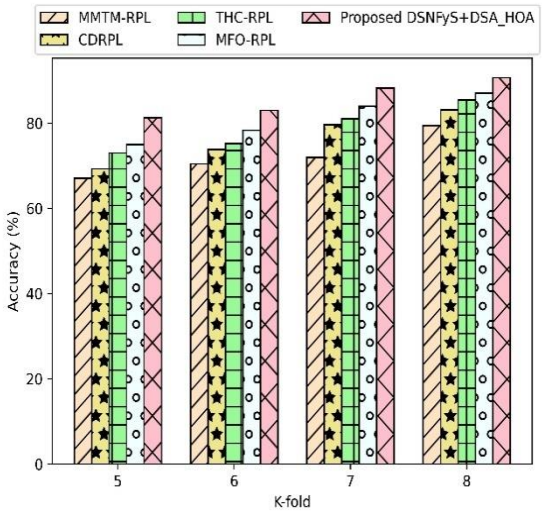


(c)

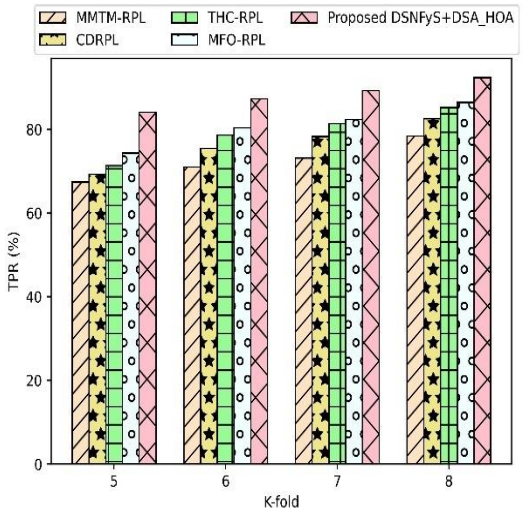
Fig. 8. Evaluation of DSNFyS+DSA_HOA changing learning data with a) accuracy, b) TPR, and c) TNR by contemplating with mitigation

-Based on k-fold

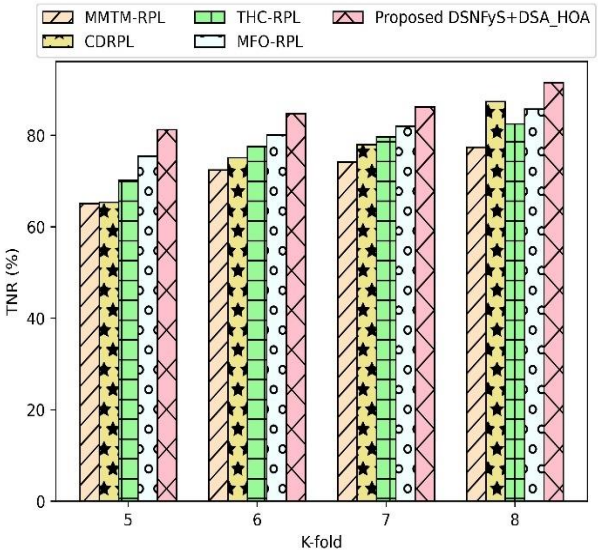
Fig. 9 shows the valuation of DSNFyS+DSA_HOA model in relation to conventional attack detection schemes concerning the k-fold. The accuracy-based evaluation of the DSNFyS+DSA_HOA technique is presented in Fig. 9a). The DSNFyS+DSA_HOA obtained a maximum accuracy of 90.710%, with a k-fold of 8 in comparison to accuracy quantified by MMTM-RPL at 79.453%, CDRPL at 83.125%, THC-RPL at 85.469%, and MFO-RPL at 87.045%. This displays that a performance improvement of 12.41%, 8.36%, 5.78%, and 4.04% is attained by the DSNFyS+DSA_HOA technique. Fig. 9b) portrays the examination of the DSNFyS+DSA_HOA concerning TPR. The TPR acquired by several attack detection models, like MMTM-RPL is78.425%, CDRPL is 82.578%, THC-RPL is 85.246%, MFO-RPL is 86.431% and DSNFyS+DSA_HOA is 92.387%, with k-fold 8. This demonstrates that DSNFyS gained a maximum performance by 15.11%, 10.62%, 7.73%, and 6.45%. The examination of DSNFyS+DSA_HOA based on TNR is demonstrated in Fig. 9c). The DSNFyS+DSA_HOA acquired TNR of 91.570%, which is superior by 18.89%, 15.75%, 15.21%, and 6.05% than the TNR measured by MMTM-RPL at 77.425%, CDRPL at 87.401%, THC-RPL at 82.496%, and MFO-RPL at 85.734%, with k-fold 8.



(a)



(b)



(c)

Fig. 9. Examination of DSNFyS+DSA_HOA changing k-fold with a) accuracy, b) TPR, and c) TNR by considering with mitigation

5.6 Comparative discussion

The effectiveness of the DSNFyS technique proposed in this study for attack detection in RPL-based IoT is assessed by comparing it with the other methods used, as indicated in Table 1. The table presents multiple metrics that are quantified by considering a 90% learning dataset and a k-value of 8. From Table 1, it can be seen that DSNFyS is effective in attaining superior accuracy, TPR, and TNR values of 90.244%, 91.345%, and 91.042%, respectively. Moreover, classical techniques, namely, MMTM-RPL, CDRPL, THC-RPL, and MFO-RPL, quantified accuracies of 72.456%, 75.490%, 80.557%, and 85.967%, respectively. The TPR measured by the MMTM-RPL was 74.316%, CDRPL was 77.648%, THC-RPL was 81.042%, and MFO-RPL was 86.135%. Likewise, traditional models, such as MMTM-RPL, CDRPL, THC-RPL, and MFO-RPL, obtained TNR of 73.240%, 75.436%, 81.679%, and 85.642%, respectively. Thus, the DSNFyS developed by the amalgamation of DSA and ANFIS models achieved more effective results than conventional techniques. DSA enhances computational efficacy, while ANFIS improves the robustness and stability of the framework with limited data. Therefore, unification of these two frameworks resulted in the best outcomes.

Table 1. Comparative discussion of DSNFyS

Variations	Metrics	MMTM-RPL	CDRPL	THC-RPL	MFO-RPL	Proposed DSNFyS
Without mitigation						
Learning data	**Accuracy (%)**	72.456	75.490	80.557	85.967	**90.244**
	TPR (%)	74.316	77.648	81.042	86.135	**91.345**
	TNR (%)	73.240	75.436	81.679	85.642	**91.042**
K-fold	**Accuracy (%)**	72.146	74.166	79.350	84.163	89.345
	TPR (%)	73.455	75.488	80.167	87.246	90.424
	TNR (%)	73.015	75.843	76.325	84.577	90.021

The efficacy of the DSNFyS+DSA_HOA methodology developed in the paper for attack mitigation in RPL-based IoT is assessed by correlating it with other methods used in this section, as indicated in Table 2. The table shows several types of measures that have been quantified using a k-value of 8 and learning data of 90%. Table 2 demonstrates that the DSNFyS+DSA_HOA method achieved optimal performance, with maximum values for accuracy, TPR, and TNR recorded at 92.043%, 92.847%, and 92.014%, respectively. Additionally, traditional models such as MMTM-RPL, CDRPL, THC-RPL, and MFO-RPL achieved accuracy of 80.425%, 86.459%, 87.013%, and 88.216%, respectively. TPR is measured as 79.466% for MMTM-RPL, 86.958% for CDRPL, 87.855% for THC-RPL, and 88.976% for MFO-RPL. Furthermore, conventional schemes including MMTM-RPL, CDRPL, THC-RPL, and MFO-RPL achieved TNR of 79.246%, 81.043%, 84.676%, and 87.400%, respectively. It can be seen from the experimental results that the DSNFyS+DSA_HOA model outperforms other existing schemes when different evaluation parameters are used, and it converges quickly to achieve superior experimental results. The DSNFyS+DSA_HOA precisely mitigates attacks without any manual intervention. Furthermore, DSA converges faster than prevailing networks in attack mitigation more correctly by maintaining accuracy. Also, HOA handled complex search spaces efficiently.

Table 2. Comparative discussion of DSNFyS+DSA_HOA

Variations	Metrics	MMTM-RPL	CDRPL	THC-RPL	MFO-RPL	Proposed DSNFyS+DSA_HOA
With mitigation						
Learning data	**Accuracy (%)**	80.425	86.459	87.013	88.216	**92.043**
	TPR (%)	79.466	86.958	87.855	88.976	**92.847**
	TNR (%)	79.246	81.043	84.676	87.400	**92.014**
K-fold	**Accuracy (%)**	79.453	83.125	85.469	87.045	90.710
	TPR (%)	78.425	82.578	85.246	86.431	92.387
	TNR (%)	77.425	87.401	82.496	85.734	91.570

6. Conclusion

Attack detection is a fundamental element of cybersecurity, and is dedicated to identifying and responding to unauthorized or malicious activities within an information system or network infrastructure. This paper proposes an innovative technique for attack detection in RPL-based IoT. Primarily, the IoT is simulated, and attack detection is handled at the Base Station based on log data. In this case, the input log data are obtained from the Bot-IoT dataset, e. After that, data normalization is executed by utilizing min-max normalization method to convert the data into a structured format. Then, feature selection is done by employing SVM-RFE and information gain to determine the most relevant features. Later, attack detection is performed by exploiting DSNFyS, which is devised by the incorporation of DSA and ANFIS. Lastly, attack mitigation is performed in case an attack is detected utilizing DSA, which is tuned using HOA. The DSNFyS acquired maximum values of accuracy, TPR and TNR at 90.244%, 91.345%, and 91.042% respectively without mitigation. DSNFyS+DSA_HOA is effective in attaining superior accuracy, TPR, and TNR values of 92.043%, 92.847%, and 92.014%, respectively. Future work will aim to contemplate large datasets to improve the early detection and response capabilities.

References

[1] M. Rouissat, M. Belkehir, A. Mokaddem, M. Bouziani, and I. S. Alsukayti, "Exploring and mitigating hybrid rank attack in RPL-based IoT networks," *J. Electr. Eng.*, vol. 75, no. 3, pp. 204–213, Jun. 2024, doi: 10.2478/jee-2024-0025.

[2] D. Ray, P. Bhale, S. Biswas, P. Mitra, and S. Nandi, "A Novel Energy-Efficient Scheme for RPL Attacker Identification in IoT Networks Using Discrete Event Modeling," *IEEE Access*, vol. 11, pp. 77267–77291, 2023, doi: 10.1109/ACCESS.2023.3296558.

[3] B. Alabsi, M. Anbar, and S. Rihan, "CNN-CNN: Dual Convolutional Neural Network Approach for Feature Selection and Attack Detection on Internet of Things Networks," *Sensors*, vol. 23, no. 14, p. 6507, Jul. 2023, doi: 10.3390/s23146507.

[4] K. Kowsalyadevi and N. V. Balaji, "IoBTSec-RPL: A Novel RPL Attack Detecting Mechanism Using Hybrid Deep Learning Over Battlefield IoT Environment," *Int. J. Comput. Netw. Appl.*, vol. 10, no. 4, p. 637, Aug. 2023, doi: 10.22247/ijcna/2023/223317.

[5] L. C. Sejaphala, V. Malele, and F. Lugayizi, "High-Level Defence Model against Routing Attacks on the Internet-of-Things," *Indones. J. Comput. Sci.*, vol. 13, no. 1, Mar. 2024, doi: 10.33022/ijcs.v13i1.3744.

[6] I. S. Alsukayti and A. Singh, "A Lightweight Scheme for Mitigating RPL Version Number Attacks in IoT Networks," *IEEE Access*, vol. 10, pp. 111115–111133, 2022, doi: 10.1109/ACCESS.2022.3215460.

[7] U. Farooq, M. Asim, N. Tariq, T. Baker, and A. I. Awad, "Multi-Mobile Agent Trust Framework for Mitigating Internal Attacks and Augmenting RPL Security," *Sensors*, vol. 22, no. 12, p. 4539, Jun. 2022, doi: 10.3390/s22124539.

[8] S. M. Muzammal, R. K. Murugesan, N. Z. Jhanjhi, M. Humayun, A. O. Ibrahim, and A. Abdelmaboud, "A Trust-Based Model for Secure Routing against RPL Attacks in Internet of Things," *Sensors*, vol. 22, no. 18, p. 7052, Sep. 2022, doi: 10.3390/s22187052.

[9] E. Garcia Ribera, B. Martinez Alvarez, C. Samuel, P. P. Ioulianou, and V. G. Vassilakis, "An Intrusion Detection System for RPL-Based IoT Networks," *Electronics*, vol. 11, no. 23, p. 4041, Dec. 2022, doi: 10.3390/electronics11234041.

[10] R. Bokka and T. Sadasivam, "Securing IoT Networks: RPL Attack Detection with Deep Learning GRU Networks," *Int. J. Recent Eng. Sci.*, vol. 10, no. 2, pp. 13–21, Apr. 2023, doi: 10.14445/23497157/IJRES-V10I2P103.

[11] W. Choukri, H. Lamaazi, and N. Benamar, "RPL rank attack detection using Deep Learning," in *2020 International Conference on Innovation and Intelligence for Informatics, Computing and Technologies (3ICT)*, Sakheer, Bahrain: IEEE, Dec. 2020, pp. 1–6. doi: 10.1109/3ICT51146.2020.9311983.

[12] R. Vatambeti and G. Mamidisetti, "Routing Attack Detection Using Ensemble Deep Learning Model for IIoT," *Inf. Dyn. Appl.*, vol. 2, no. 1, pp. 31–41, Mar. 2023, doi: 10.56578/ida020104.

[13] N. Moustafa, "The Bot-IoT dataset." IEEE DataPort, Oct. 16, 2019. doi: 10.21227/R7V2-X988.

[14] M. Albishari, M. Li, R. Zhang, and E. Almosharea, "Deep learning-based early stage detection (DL-ESD) for routing attacks in Internet of Things networks," *J. Supercomput.*, vol. 79, no. 3, pp. 2626–2653, Feb. 2023, doi: 10.1007/s11227-022-04753-4.

[15] D. Arshad, M. Asim, N. Tariq, T. Baker, H. Tawfik, and D. Al-Jumeily Obe, "THC-RPL: A lightweight Trust-enabled routing in RPL-based IoT networks against Sybil attack," *PLOS ONE*, vol. 17, no. 7, p. e0271277, Jul. 2022, doi: 10.1371/journal.pone.0271277.

[16] A. Seyfollahi, M. Moodi, and A. Ghaffari, "MFO-RPL: A secure RPL-based routing protocol utilizing moth-flame optimizer for the IoT applications," *Comput. Stand. Interfaces*, vol. 82, p. 103622, Aug. 2022, doi: 10.1016/j.csi.2022.103622.

[17] T. D. Nguyen, J. Y. Khan, and D. T. Ngo, "An effective energy-harvesting-aware routing algorithm for WSN-based IoT applications," in *2017 IEEE International Conference on Communications (ICC)*, Paris, France: IEEE, May 2017, pp. 1–6. doi: 10.1109/ICC.2017.7996888.

[18] Z. Ye, T. Wen, Z. Liu, X. Song, and C. Fu, "An Efficient Dynamic Trust Evaluation Model for Wireless Sensor Networks," *J. Sens.*, vol. 2017, pp. 1–16, 2017, doi: 10.1155/2017/7864671.

[19] M. Salehi, A. Boukerche, A. Darehshoorzadeh, and A. Mammeri, "Towards a novel trust-based opportunistic routing protocol for wireless networks," *Wirel. Netw.*, vol. 22, no. 3, pp. 927–943, Apr. 2016, doi: 10.1007/s11276-015-1010-4.

[20] N. A. Khalid, "Distributed Trust-based Routing Decision Making for WSN," 2019.

[21] J. Zhu, "Wireless Sensor Network Technology Based on Security Trust Evaluation Model," *Int. J. Online Biomed. Eng. IJOE*, vol. 14, no. 04, pp. 211–226, Apr. 2018, doi: 10.3991/ijoe.v14i04.8590.

[22] H. Henderi, "Comparison of Min-Max normalization and Z-Score Normalization in the K-nearest neighbor (kNN) Algorithm to Test the Accuracy of Types of Breast Cancer," *IJIIS Int. J. Inform. Inf. Syst.*, vol. 4, no. 1, pp. 13–20, Mar. 2021, doi: 10.47738/ijiis.v4i1.73.

[23] Y. Zhang, Q. Deng, W. Liang, and X. Zou, "An Efficient Feature Selection Strategy Based on Multiple Support Vector Machine Technology with Gene Expression Data," *BioMed Res. Int.*, vol. 2018, pp. 1–11, Aug. 2018, doi: 10.1155/2018/7538204.

[24] B. Prasetiyo, Alamsyah, M. A. Muslim, and N. Baroroh, "Evaluation of feature selection using information gain and gain ratio on bank marketing classification using naïve bayes," *J. Phys. Conf. Ser.*, vol. 1918, no. 4, p. 042153, Jun. 2021, doi: 10.1088/1742-6596/1918/4/042153.

[25] A. Dairi, F. Harrou, Y. Sun, and M. Senouci, "Obstacle Detection for Intelligent Transportation Systems Using Deep Stacked Autoencoder and k -Nearest Neighbor Scheme," *IEEE Sens. J.*, vol. 18, no. 12, pp. 5122–5132, Jun. 2018, doi: 10.1109/JSEN.2018.2831082.

[26] M. Ragab *et al.*, "A novel metaheuristics with adaptive neuro-fuzzy inference system for decision making on autonomous unmanned aerial vehicle systems," *ISA Trans.*, vol. 132, pp. 16–23, Jan. 2023, doi: 10.1016/j.isatra.2022.04.006.

[27] S. O. Oladejo, S. O. Ekwe, and S. Mirjalili, "The Hiking Optimization Algorithm: A novel human-based metaheuristic approach," *Knowl.-Based Syst.*, vol. 296, p. 111880, Jul. 2024, doi: 10.1016/j.knosys.2024.111880.

[28] H. T. Sadeeq and A. M. Abdulazeez, "Improved Northern Goshawk Optimization Algorithm for Global Optimization," in *2022 4th International Conference on Advanced Science and Engineering (ICOASE)*, Zakho, Iraq: IEEE, Sep. 2022, pp. 89–94. doi: 10.1109/ICOASE56293.2022.10075576.

[29] S. He, Q. H. Wu, and J. R. Saunders, "Group Search Optimizer: An Optimization Algorithm Inspired by Animal Searching Behavior," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 973–990, Oct. 2009, doi: 10.1109/TEVC.2009.2011992.

[30] Natanael Sousa, José V.V. Sobral, Joel J.P.C. Rodrigues, Ricardo A.L. Rabêlo, and Petar Solic, "ERAOF: A new RPL protocol objective function for Internet of Things applications," presented at the 2nd International Multidisciplinary Conference on Computer and Energy Science, SpliTech 2017, in 2017 2nd International Multidisciplinary Conference on Computer and Energy Science, SpliTech 2017. Split, Croatia: Institute of Electrical and Electronics Engineers Inc., Jul. 2017.

[31] Y. Yu, J. Li, J. Li, Y. Xia, Z. Ding, and B. Samali, "Automated damage diagnosis of concrete jack arch beam using optimized deep stacked autoencoders and multi-sensor fusion," *Dev. Built Environ.*, vol. 14, p. 100128, Apr. 2023, doi: 10.1016/j.dibe.2023.100128.

[32] P. R. Bhaladhare and D. C. Jinwala, "A Clustering Approach for the l -Diversity Model in Privacy Preserving Data Mining Using Fractional Calculus-Bacterial Foraging Optimization Algorithm," *Adv. Comput. Eng.*, vol. 2014, pp. 1–12, Sep. 2014, doi: 10.1155/2014/396529.

[33] O. Chamorro-Atalaya *et al.*, "K-Fold Cross-Validation through Identification of the Opinion Classification Algorithm for the Satisfaction of University Students," *Int. J. Online Biomed. Eng. IJOE*, vol. 19, no. 11, Aug. 2023, doi: 10.3991/ijoe.v19i11.39887.

Authors' Profiles

Prashant Maurya received his M.Tech degree in 2014. He is a research scholar at department of Computer Science, Institute of Science, Banaras Hindu University, Varanasi, India. His research interest includes Wireless Sensor Networks, Network Security and Internet of Things.

Vandana Kushwaha is working as an Assistant Professor in Department of Computer Science, Institute of Science, Banaras Hindu University, Varanasi (India). She has 17 years of teaching experience. She has done her Ph.D. in Computer Science at the Department of Computer Science, Institute of Science, Banaras Hindu University, Varanasi (India). Her research interests include High Speed Network, Wireless Sensor Network, Internet of Things.

How to cite this paper: Prashant Maurya, Vandana Kushwaha, "DSNFyS: Deep Stacked Neuro Fuzzy System for Attack Detection and Mitigation in RPL based IoT", International Journal of Information Engineering and Electronic Business(IJIEEB), Vol.17, No.3, pp. 62-83, 2025. DOI:10.5815/ijieeb.2025.03.05