Iraqi Journal of Science, 2025, Vol. 66, No. 11, pp: 5204- 5224
DOI: 10.24996/ijs.2025.66.11.40

ISSN: 0067-2904

Detection of Multiple Attacks in Wireless Sensor Networks and Enhancing Security Using Hybrid Self-Organizing Neural Networks

Rafid S. Abdulaziz*
College of Education for Pure Sciences, University of Anbar, Ramadi-City, Iraq

Received: 28/6/2024 Accepted: 16/10/2024 Published: 30/11/2025

Abstract

In many different applications, wireless sensor network (WSN) security enhancement is crucial. The progress in localizing routing attacks is a critical focus of security research. The network performance is compromised by introducing rogue nodes into wireless sensor networks, resulting in various routing attacks. This research introduces a method called hybrid self-organizing neural networks (Hyb_SONN) for detecting and localizing numerous assaults. The proposed approach consists of two main components: localization techniques and a neural network for the detection and localization of WSN DoS attacks. Using the Hyb_SONN method, the dataset is divided into training and testing sets to detect 10 different types of attacks, including denial-of-service (DoS) attacks. The purpose of the simulations is to assess the security-related effectiveness of the proposed technique for locating and detecting malicious nodes. With little localization error, this method provides an accurate assessment of the unknown nodes' position. The proposed system can effectively detect and precisely locate malicious attacks in hierarchically distributed, scalable WSNs. The results indicate that the proposed approach has a high level of accuracy, with a rate of 99.8%. Additionally, the precision is measured at 96.4%, recall at 94.5%, PDR at 98.5%, energy consumption at 43.2%, and localization error at 13.2%.

Keywords: Multiple Attacks, Wireless Sensor Networks, Localization Technique, Neural Network, Denial-of-Service (DoS)

كشف الهجمات المتعددة في شبكات الاستشعار اللاسلكية وتعزيز الامان باستخدام الشبكات العصبية الذاتية التنظيمية الهجينة

رافد صيهود عبد العزيز *
كلية التربية للعلوم الصرفة، جامعة الانبار، الانبار، العراق

الخلاصة

في العديد من التطبيقات المختلفة، يعتبر تعزيز أمان شبكات الاستشعار اللاسلكية (WSN) أمرًا بالغ الاهمية. يعد التقدُّم في تحديد مواقع هجمات التوجيه محورا أساسيًا في أبحاث الأمان. تعرض كفاءة الشبكة للخطر عند إدخال العقد الخبيثة إلى شبكات الاستشعار اللاسلكية، مما يؤدي إلى حدوث هجمات توجيه مختلفة. تقدم هذه الدراسة أسلوبًا يُسمى شبكات العصبية الذاتية التنظيمية الهجينة (Hyb_SONN) للكشف عن الهجمات وتحديد مواقعها بشكل محلي. يتكون النهج المقترح من مكونين رئيسيين: تقنيات التحديد المكاني

* Email: rafid.alhashimy@uoanbar.edu.iq

وشبكة عصبية لاكتشاف وتحديد هجمات قطع الخدمة في WSN. باستخدام طريقة Hyb_SONN، يتم تقسيم مجموعة البيانات إلى مجموعات تدريب واختبار من أجل كشف 10أنواع مختلفة من الهجمات، بما في ذلك هجمات انقطاع الخدمة (DoS) .تهدف المحاكاة إلى تقييم فعالية الأمان المقترحة للتقنية في تحديد واكتشاف العقد الخبيثة. باستخدام هذا الأسلوب، يتم توفير تقدير دقيق لموقع العقد المجهول مع الحد الأدنى من الخطأ في التحديد. يمكن للنظام المقترح اكتشاف الهجمات الخبيثة وتحديد مواقعها بدقة في WSN ذات التوزيع الهرمي والقابلة للتوسعة. تُظهر النتائج أن النهج المقترح يتمتع بمستوى عالٍ من الدقة، بنسبة 99.8٪. بالإضافة إلى ذلك، يتم قياس الدقة بنسبة 96.4٪، والاستدعاء بنسبة 94.5٪، و (PDR) بنسبة 98.5٪، واستهلاك الطاقة بنسبة 43.2٪، وخطأ التحديد بنسبة 13.2٪.

1. Introduction

Distributed denial-of-service (DDoS) attacks are becoming more common on wireless sensor networks. The attacks mentioned, including the sinkhole, black hole, grey hole, wormhole, Sybil, and clone assaults, serve as illustrative instances of such attacks [1]. The continual improvement of wireless communication has contributed to the growing adoption of WSN deployment. WSNs are characterized by their self-organizational nature since they comprise a network of sensor nodes that exhibit self-organization [2]. Sensor nodes are devices that have a cheap cost and low power consumption [3]. The nodes possess the capability to acquire data, whereas the processing-band sensors are responsible for the collection of information. The processing-band sensors also engage in preprocessing as part of their operational procedures [4]. Sensor nodes are used to transmit data across the network. These routers may forward nearby devices' data to the base station (BS). This data is sent to remote servers via the base station [5]. Due to its dynamic design and high-fidelity data transmission, WSNs have several uses. They are used in smart homes, healthcare, and environmental monitoring [6], as well as in military, urban, and industrial applications [7,8].

WSNs possess a versatile architecture and provide a range of features, making them very convenient for deployment. Nevertheless, the susceptibility of sensor nodes to DoS assaults is a consequence of their constrained functionalities and inexpensive, low-power nature [9,10]. The frequent occurrence of these security concerns is increasing regularly, resulting in network disruptions via data manipulation, unauthorized disclosure of private information, facilitation of access for unauthorized users, or enabling unlawful access [11]. The types of DoS assaults include wormholes, black holes, jamming, and clone attacks. The wormhole assault is widely recognized as one of the most detrimental attacks against WSNs [12]. This particular attack presents a significant challenge in terms of countermeasures due to its covert nature, as the attacker selectively targets two specific malicious nodes and establishes a concealed tunnel between them for data forwarding [13]. The perpetrator can transmit data, regulate traffic, alter data, and manipulate information while seeming to be a genuine system [14]. The detection of wormhole assaults is a significant problem due to their inherent difficulty in counteracting [15]. Additional research has been undertaken to identify and mitigate different threats that may exploit WSNs. The descriptions of these attacks are provided as follows:

- A sinkhole attack is a kind of insider attack where an unauthorized individual gains access to a node inside a network and proceeds to initiate an attack [16]. The compromised node attempts to draw traffic from other nodes by using the routing metric used in the routing protocol. After successfully attaining its objective, it will initiate an offensive maneuver. The vulnerability of WSNs to sinkhole attacks arises from the communication topology used, which involves several nodes transmitting data to a central base station.
- A black hole attack is a kind of assault in which the assailant deliberately discards packets in a selected manner, including both control and data packets that go via the attacker

[17]. Consequently, every packet that is routed via this intermediary malicious node will experience either partial or complete loss of data. The black hole attack works by obstructing the flow of network communication originating from a certain host or container. The act of dropping Internet Protocol (IP) packets at the transport layer is achieved using traffic policing functionalities included inside the Linux Kernel and the Windows Filtering Platform for Windows hosts [18].

- A Sybil attack may occur in peer-to-peer networks when a single node takes several fake identities (Sybil identities) [19]. The objective of this kind of assault is to subvert the legitimacy or control inside a respected system by attaining a significant majority of influence within the network.

The use of artificial intelligence has surfaced as a promising approach for the identification of anomalies in WSNs. Deep learning and machine learning algorithms can acquire knowledge from sensor data and identify patterns that may be used to differentiate between normal and abnormal behavior. This process eliminates the need for human rule development. Furthermore, these approaches can adjust to variations in the system and the surrounding conditions, hence facilitating precise and prompt identification of irregularities. Consequently, this enhances the effectiveness and dependability of WSNs [20]. The ensemble learning method [21] is a popular DL/ML-based methodology where many ML models are trained and tested before selecting the best-performing models. This technique demonstrates efficacy not just in WSN but also finds widespread use in the identification of abnormalities and malware in several other contexts. Considering the aforementioned, this study's contributions can be summarized as follows.

- This study aims to examine the practical implementation of WSNs for target localization and tracking. Specifically, the emphasis is on enabling individual sensors to determine their positions by using the average hop size (AHS) of each beacon node.
- The researchers used a Hybrid Self-Organizing Neural Networks (Hyb_SONN) classification approach to effectively categorize incursion types, resulting in notable performance results.
- A clustering strategy has been devised to minimize the computational complexity of the proposed Hyb_SONN structure. The findings from the localization experiment provide evidence that the suggested localization technique, Hyb_SONN, demonstrates satisfactory levels of accuracy in object localization.

The subsequent sections are structured as follows: Section 2 provides an overview of many prior research studies, Section 3 presents the suggested methodology and techniques, Section 4 showcases the experimental results and subsequent discussion. Lastly, Section 5 concludes with a summary and outlines potential avenues for future study.

2. Related Works

Denial-of-service (DoS), Sybil, and distributed denial-of-service (DDoS) attacks are widely recognized as prominent risks to the security of WSNs due to their relatively simple launch mechanisms. Over many years, several researchers have proposed different intrusion detection systems (IDS) for multiple detection. In this section, an examination is conducted on the previous publications, with a specific emphasis on contemporary IDSs that use deep learning and machine learning techniques.

In [22], an intrusion detection system was proposed to identify routing assaults in WSNs. The system utilized fuzzy and feed-forward neural networks (FFNN) as the underlying

detection mechanisms. The experimental findings indicated that the model provided in this study obtained a detection accuracy of 98.8%.

To optimize computing resources and mitigate modal mixing, the typical Hilbert-Huang transform (HHT) involves the elimination of redundant and comparable intrinsic mode functions (IMFs) [23]. The one-dimensional dataflow characteristics are converted into two-dimensional temporal-spectral features using the compressed HHT transformation. These features are then fed into a CNN to detect LDoS assaults. In [24], the CICIDS dataset was obtained from real-world data; the proposed model directly detected domain system flood assaults without label coding, normalization, or feature selection. The performance indicators were assessed by comparing chosen machine learning, shallow neural networks, and deep learning classifiers. Even without normalization, label encoding, feature elimination, and false alarms were few, and the detection accuracy was effective.

The study in [25] introduced a deep neural network-based approach for real-time identification of malicious packets in the context of intrusion detection. The model was trained using recently created benchmark datasets based on NetFlow. A packet collecting and detecting technique was presented for real-time attack detection. Additionally, [26] suggested using an evolving fuzzy neural network to address data stream regression challenges while emphasizing its significant interpretability. The dataset was derived from a predictive model that used wireless sensors to detect and identify unauthorized individuals attempting to access a secured area.

In [27], a deep neural network (DNN) was presented as the foundation for an intrusion detection system. The cross-correlation approach was used to choose the most suitable characteristics from the dataset, and these chosen parameters served as fundamental components for constructing the deep neural network architecture to detect intrusions. The study in [28] developed a robust and efficient Artificial Intelligent Diagnosis System (AIDS) that used fuzzy and neural network (NN)-based methodologies. The suggested system had the potential to be deployed at each node due to its lightweight nature and little resource consumption. Furthermore, it could autonomously observe and assess the conduct of the local nodes and to determine the level of trust, mistrust, or enmity associated with each node.

The study in [29] developed a dynamic recursive feature selection algorithm to identify the best number of features to be selected from a given dataset. Furthermore, this study proposed an intelligent fuzzy temporal decision tree method that expanded upon the existing decision tree technique and incorporated convolutional neural networks for enhanced intruder detection capabilities.

In [30], an array of ensemble-based machine learning algorithms, including AdaBoost, stacking, random forest, bagging, and voting, was deployed. Amongst these, our findings demonstrate that AdaBoost emerges as the preeminent choice, achieving unprecedented levels of performance.

The extensive literature review has revealed several significant limitations in the existing methodologies. These are:

- The performance of several models is seen to be low for example, In WSN security, models like trust-based mechanisms and anomaly detection algorithms often exhibit low accuracy when identifying malicious nodes due to noise in the sensor data or dynamic network conditions.

- The classification process is characterized by the issue of excessive memory usage. For example, WSNs are composed of resource-constrained nodes with limited memory, and many classification models require significant storage for parameters or past data. For instance, deep learning models like CNNs or recurrent neural networks (RNNs) require substantial memory to store weights and intermediate outputs.
- The curse of dimensionality is a significant challenge in this context, resulting in high computational complexity. For example, Security features in WSNs often involve multiple attributes, such as node behavior, communication patterns, signal strength, and trust levels. When applying machine learning or optimization algorithms like trust-based clustering, the dimensionality of the data can explode.
- The inability to effectively handle large-scale datasets is a limitation in the classification task. For example, in large WSN deployments (e.g., in smart cities or industrial applications), the volume of data generated can overwhelm the models used for security, making it difficult to train or update classifiers that detect abnormal patterns or attacks.

3. Network Model

The simulated network includes representations of sink nodes, cluster heads, sensors, beacon nodes, and attacker nodes. The sensor nodes prefer to congregate inside the system. Figure-1 shows that each cluster chooses a cluster head node to coordinate data transmission to the beacon nodes and the base station. The beacon nodes use a fitness function to find the best routing path.

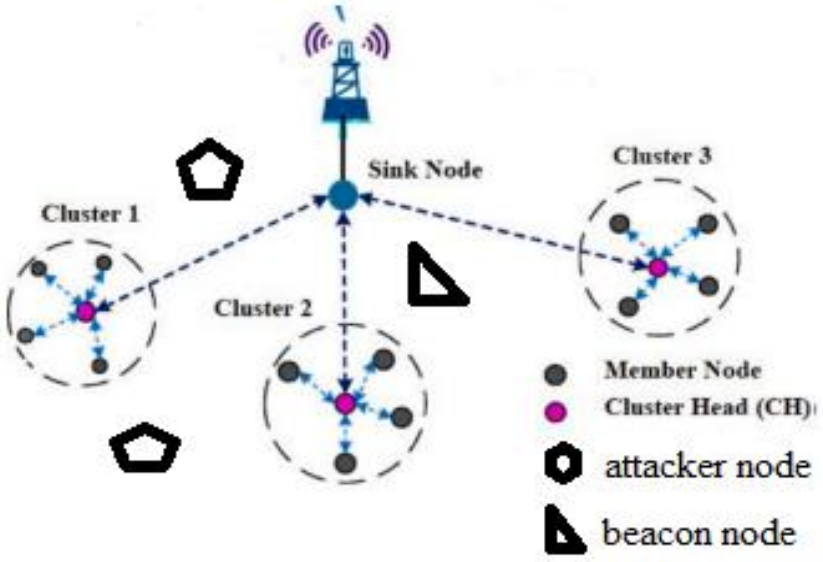

Figure 1: Network Model

The placement of anchor nodes is determined by their connections. Anchor nodes in networks remain stationary indefinitely. Sensor nodes have successfully established their positions. The localization approach involves grouping sensor nodes with a cluster head to accurately determine their positions. Unidentified nodes rely on anchor nodes to determine their placements. The system regularly updates the positions of sensor nodes. Malicious nodes disclose their locations by creating multiple fake identities and presenting themselves as beacon nodes. These malevolent nodes also create tunnels to intercept packets before they reach their intended destinations. The model assumes that valid nodes possess identical computational processing capabilities, storage capacity, communication levels, and activation energy. Malicious nodes obtain base station and cluster security keys faster than permitted nodes. The attacker duplicates the authorized node, disrupting network functionality. The

network is composed of n sensor nodes that are placed randomly throughout an area of size M × M. The network is divided into clusters. The base station or sink is positioned at a considerable distance from the sensing region to carry out data processing. Every cluster consists of a cluster head (CH) that regularly collects data from sensors referred to as cluster members (CM). The CH collects data and transmits it to the BS using multi-hop routing algorithms. The following are the network assumptions.

- Each sensor node placed in the sensing region is given a unique identifier.
- The nodes are uniform and are allocated with the same beginning energy. The nodes and sink remain stationary.
- The calculation of neighbors and distance is performed using the Euclidean formula.

$D = \sqrt{(yi - yj)^2 - (xi - xj)^2}$

3.1 Cluster Formation and Data Aggregation

In traditional LEACH, the selection of the CH is random and does not consider energy factors. The selection of CH is governed by the threshold value $thresh_{(ni)}$ and the random number generator function. The nodes ni create random numbers R, where $(0 \leq R \leq 1)$. If the nodes meet the requirement of being less than or equal to the threshold value of $\leq thresh_{(ni)}$. An inherent drawback of standard LEACH is that a node with less energy has the potential to become a cluster head, resulting in a shorter lifespan. To enhance the process of selecting cluster heads, the energy parameters are considered. The cluster head is chosen by calculating a threshold value, denoted as $thresh_{(ni)}$, which can be mathematically represented as.

$$thresh_{(ni)} = \begin{cases} \frac{pb_i}{1 - pb_i * (r * mod\left(\frac{1}{pb_i}\right))} ni \in V \\ 0, otherwise \end{cases} \quad (1)$$

The energy parameters pb_i, where ni represents the node and $i \in [1, N]$, are provided as:

$$pb_i = \frac{\boldsymbol{p} * ni * e_r^i * e_i}{\boldsymbol{e_t} * \boldsymbol{e_a}} \quad (2)$$

The probability of picking the optimum CH is denoted as P. The current residual energy of a node is represented as e_r^i. The node's starting energy is denoted as e_i. The entire network energy is represented as $\boldsymbol{e_t}$, and the average energy of all nodes is denoted as $\boldsymbol{e_a}$.

The suggested cluster head selection threshold balances the energy load among nodes and extends network longevity. The possibility of a node being picked as a CH increases if its residual energy exceeds $\boldsymbol{e_a}$. After r rounds, all nodes' average energy is calculated as.

$$\boldsymbol{e_a} = \frac{\boldsymbol{e_t}\left(\mathbf{1} - \frac{r}{r(max)}\right)}{ni} \quad (3)$$

3.2 Cluster Formation

After the CH selection procedure, the selected CHs will send cluster head advertising messages to non-cluster head nodes to join the cluster. Nodes join clusters depending on RSS and the distance to the sink node. Signal strength is related to node proximity to sink and CH. By determining the distance to the CH, the nodes send a join-request message to the selected CH and alert it that they are cluster members (CM). After calculating energy and distance to the sink, the threshold $thresh_{(ni)}$ is determined for each round. Nodes near the CH generate a

random number and compare it to $thresh_{(ni)}$. If the produced number is below the threshold, the node may become a CH. The distance between the CH and the sink is calculated. Nodes close to sinks are not participating in cluster creation. Instead, the node may communicate data directly with the sink.

3.3 Energy Consumption Model

The radio hardware circuit determines a node's energy utilization while transmitting and receiving packets. The energy model determines node energy consumption depending on sender-receiver distance using the free space channel model (d^2) or multipath channel model (d^2). The formula for calculating energy usage to carry a packet of k bits across d is:

$$e_{tx} = \begin{cases} k * e_{elec} + k * \varepsilon_{fs} * d^2 d \leq d_0 \\ k * e_{elec} + k * \varepsilon_{mp} * d^4 d > d_0 \end{cases} \quad (4)$$

where d_0 is determined by $d_0 = \sqrt{\varepsilon_{fs}/\varepsilon_{mp}}$, and ε_{fs} and ε_{mp} represents the amplification factors of the transmitter circuit. The d_0 serves as a threshold, indicating that when $d \leq d_0$ and $d > d_0 0$, e_{elec}, they represent the amount of electronic energy used for digital coding and modulation.

The amount of energy required to receive k bits is:

$$e_{tx} = k * e_{elec} \quad (5)$$

The energy model used when nodes are closer to the sink is the free space model d^2. The energy dissipation of the CH every round is determined by:

$$e_{CH} = \left(\frac{n}{A} - 1\right) * k * e_{elec} + \frac{n}{A} * k * e_M + k * e_{elec} + k * \varepsilon_{fs} * d^2_{to-sink} \quad (6)$$

Let A be the number of clusters every round, and $\frac{n}{A}$ represents the average number of nodes in each cluster. The energy consumption of a CH while receiving a 1-bit message is represented as e_M, and the distance to the sink is marked as $d^2_{to-sink}$.

Since the sink node is positioned at coordinates (a, b) in a random area and the nodes are scattered throughout the region with coordinate values $\varphi(x, y)$, the distance between the CH and the sink can be calculated as:

$$D[d^2_{to-sink}] = \iint ((a-x)^2 + (b-y)^2)\varphi(x,y)dxdy = \iint \frac{(a-x)^2+b-y^2}{Z} dxdy \quad (7)$$

The distribution area of the nodes is represented by Z. The energy dissipation of cluster members every round is:

$$e_{CM} = k * e_{elec} * \varepsilon_{fs} * d^2_{to-CH} \quad (8)$$

where d^2_{to-CH} denotes the squared distance between the CM to CH.

$$D[d^2_{to-CH}] = \iint (x^2 + y^2)\varphi(x,y)dxdy = \frac{Z^2}{2\pi A} \quad (9)$$

The energy consumption of each cluster is provided as:

$$e_{cluster} = e_{CH} + (\frac{n}{A} - 1)e_{CM} \quad (10)$$

The calculation of energy usage for each round is as follows:

$$e_{round} = CH_{opt}e_{cluster} = k(2ne_{elec} + ne_M + CH_{opt}\varepsilon_{fs}d^2_{to-sink} + n\varepsilon_{fs}d^2_{to-CH}) \quad (11)$$

The ideal CH for each round, denoted as CH_{opt}, is determined based on the associated e_{round}.

$$CH_{opt} = \frac{\sqrt{n}}{\sqrt{2\pi}}\frac{Z}{d^2_{to-sink}} \quad (12)$$

The energy dissipation model used for the multipath channel is provided by d^4 when the sink is located at a significant distance from the nodes.

$$e_{CH} = \left(\frac{n}{A} - 1\right) * k * e_{elec} + \frac{n}{A} * k * e_M + k * e_{elec} + k * \varepsilon_{mp} * d^2_{to-sink} \quad (13)$$

The distance to the sink is provided as:

$$D[d^4_{to-SINK}] = \iint (a-x)^2 + (b-y)^2)\varphi(x,y)dxdy = \iint \frac{(a-x)^2+b-y^2}{Z}dxdy \quad (14)$$

The energy utilization is quantified as :

$$e_{round} = CH_{opt}e_{cluster} = k(2ne_{elec} + ne_M + CH_{opt}\varepsilon_{mp}d^4_{to-SINK} + n\varepsilon_{fs}d^2_{to-CH}) \quad (15)$$

The cluster head CH_{opt} for each round is determined as follows:

$$CH_{opt} = \frac{\sqrt{n}}{\sqrt{2\pi}}\sqrt{\frac{\varepsilon_{fs}}{\varepsilon_{mp}}}\frac{Z}{d^2_{to-SINK}} \quad (16)$$

3.4 Data Aggregation

The data aggregation approach guarantees the transfer of data without replication, hence reducing the computational burden on the CH and minimizing energy usage. The proposed EDAS uses a linear XOR operation for inter-cluster communication between CH across stable networks. When transmitting data, it is important to consider two factors: consistent link utilization and the sequence number. The calculation of steady link use is determined by:

$$L_s = \frac{k_s}{l_d} \quad (17)$$

where k_s represents the size of the forwarded bits in the message, while $\boldsymbol{l_d}$ represents the link delay. The link delay is determined by:

$$l_d = s_d + f_d \quad (18)$$

During inter-cluster communication, when the CH sends data to the next cluster head, the dependability is determined by evaluating the successful transmission of non-replicated messages, denoted as k_s. ϑ uses the range of values between 0 and 1 to provide a conventional normal distribution for monitoring the t outcomes.

The standard distribution rule states that the mean is supplied by multiplying t with μ, and the variance is given by multiplying t with σ^2. The normalized distribution of data dependability, abbreviated as ϑ', is represented as:

$$\vartheta' = \frac{\vartheta - t - \mu}{\sqrt{t \times \sigma^2}} = \frac{\tau - t \times \mu}{\sqrt{t} \times \sigma} \quad (19)$$

If $\vartheta' \geq b_\vartheta$, it reflects transmission dependability. The result of non-replicated transmission t'_{nr} transmitted from CH can be calculated as follows:

$$t'_{nr} = k_s \times \frac{t_{nr}}{k_r} \quad (20)$$

where the variable k_r indicates the rate at which data is created. The transmission time T_X, is represented as $[T_1, T_2, T_3, \ldots T_n] \in T$, which is associated with each transmission. Each transmission also creates a unique sequence number $[S_1, S_2, S_3, \ldots S_n]$, where each number is mapped to a specific T value. For every value of T, the condition ϑ' is satisfied and calculated as $\vartheta' \geq b_\vartheta$. The expression for the last flow message $k_s(i,j)$ is as follows:

$$k_s(i,j) = k_r * h_{i,j} \quad (21)$$

where $h_{i,j}$ represents the transmission between two CH nodes, and j represents the hop count identified between them. The recipient node checks the sequence number against the transmission number T_X. The reciprocal of the transmission sequence number with time slots for $I_d \neq 0$ should be one, as shown in:

$$[S_1, \ldots . S_n] \times [T_1, \ldots T_n] = 1 \quad (22)$$

The non-redundant data is calculated by eliminating sequence numbers that do not meet the criteria defined in Equation (22).

$$k_{s-nr} = k_s * \frac{t_{nr}}{k_r * h_{i,j}} \quad (23)$$

To solve equation (20) and replace t_{nr} into equation (23),

$$k_{s-nr} = k_s * \frac{t_{nr} * k_r}{k_r * h_{i,j} * k_s} \quad (24)$$

$$k_{s-nr} = \frac{t_{nr}}{h_{i,j}} \quad (25)$$

During data aggregation, some of the nodes may be unknown, and it is important to localize them. This can be done by the following localization technique.

3.5 Localization Technique

In the process of placing unknown nodes, if the average hop size (AHS) of each beacon node is not precise, it will amplify the inaccuracy in the succeeding stages and affect the accuracy of the findings. Hence, it is important to use a precise method for calculating the AHS. Due to the random deployment of sensor nodes within the detection region, the nodes must exert varying effects on each other in terms of placement. The proposed system first uses the MMSE criteria to calculate the initial AHS of beacons. Subsequently, it gives varying weights to each beacon, taking into account the per-hop error, to equalize the impact of beacons with significant per-hop errors. The computation of the AHS and allocation of weights will continue until the specified termination condition is satisfied. Finally, the AHS with the lowest error may be designated as the ultimate AHS of beacons. The intricate procedure of our proposed method for AHS is delineated as follows.

Step 1: Obtain the initial AHS and its corresponding error. To begin with, if we assume that all beacons have an equal impact on each other, we can calculate the initial AHS of each beacon using Equation (26) based on the MMSE criteria. Simultaneously, the predetermined distance between beacon nodes is already known, and the approximated distance can be substituted with the multiplication of AHS and hop count. Hence, the inaccuracy in the initial AHS can be determined using Equation (27).

$$AHS_i^{init} = \frac{\sum_{f\neq i}^{N_a} Dis_{ij}.h_{ij}}{\sum_{f\neq i}^{N_a} h_{ij}^2} \tag{26}$$

$$\beta_t^{init} = \frac{\sum_{j\neq 1}^{N_a} |Dis_{ij} - AHS_t^{init}.h_{ij}|}{N_a - 1} \tag{27}$$

where AHS_i^{init} indicates the initial AHS of the beacon node i, and Dis_{ij} represents the precise distance between beacons i and j. β_t^{init} represents the error of the initial AHS of beacon node i.

Step 2: Determine the per-hop discrepancy between beacons. The inter-beacon error can be calculated using the following equation:

$$EH_{(i)j} = \frac{|dis_{ij} - AHS_i^*.h_{ij}|}{h_{ij}} \tag{28}$$

where $EH_{(i)j}$ represents the per-hop error from beacon i to j, and AHS_i^* is the AHS acquired by the iteration of beacon node i.

Step 3: Calculate the AHS weighted average. An example of beacon dependency is the per-hop error. To lessen its influence on the beacon node i's AHS, beacon node j will be assigned a lower weight value if its hop inaccuracy is considerable. Alternatively, beacon node j will be weighted higher. Using weighted MMSE, the AHS can be enhanced. The weights and weighted AHS formula are shown below. Due to the inability to minimize disparities between planned and actual distance, Equation (29)'s denominator will not be equal to zero.

$$\propto_{(i)j} = \left(\frac{1}{EH_{(i)j}}\right)^2 \tag{29}$$

$$AHS_i^{(m)} = \frac{\sum_{j\neq i}^{N_a} \propto_{(i)j}.Dis_{ij}.h_{ij}}{\sum_{j\neq i}^{N_a} \propto_{(i)j}.h_{ij}^2} \tag{30}$$

where the weight value for beacon j while computing the AHS of beacon i is denoted as $\propto_{(i)j}$, and $AHS_i^{(m)}$ represents the average hop count of beacon i, which is collected during the m-th iteration.

Step 4: Calculate the error of the weighted AHS. The assessment criteria for weighted AHS are identical to Equation (27).

$$\beta_i^{(m)} = \frac{\sum_{j\neq i}^{N_a} \left|Dis_{ij} - AHS_i^{(m)}.h_{ij}\right|}{N_a - 1} \tag{31}$$

where $\beta_i^{(m)}$ represents the error of AHS acquired in the m-th iteration of the beacon node i.

Step 5: Determine the most effective AHS. If the value of $\beta_i^{(m)}$ is less than $\beta_i^{(m-1)}$, return to step 2 and recompute the per-hop error using $AHS_i^{(m)}$.

Therefore, a new AHS of beacon node i can be calculated using Equation (30) again. The aforementioned procedure involves conducting the weighted iterative computation many times on the AHS of the beacon node.

If, during a certain iteration, the value of $\beta_i^{(m)}$ is greater than $\beta_i^{(m-1)}$, the iteration terminates, and the AHS acquired in the $(m-1)$th iteration is selected as the optimum AHS for the beacon i.
Every beacon node can calculate its optimal AHS using a weighted iterative approach, reducing cumulative error while recognizing unknown nodes. In certain cases, the beacon node's initial AHS is accurate enough to avoid the weighted iterative technique. The iterative procedure in most beacon nodes will continue until it reaches the ideal solution. The issue of localizing approaches is addressed by using hybrid self-organizing neural networks. This is achieved by calculating the distance and location of each kind of node, which has a distinct identity, to identify and locate the malicious nodes.

In conventional hopping, unidentified nodes use the closest beacon node's AHS. Their estimated distance is calculated by multiplying their AHS by the hops needed to reach them. However, unrecognized nodes in real networks may have a much different AHS from the nearest beacon node. This disparity will increase calculated distance inaccuracy and affect unidentified node location. AHS computation for unknown nodes may be removed using beacon node AHS. It is possible to calculate the distance using the minimum number of hops to unknown nodes and the hop set (AHS) of beacon nodes. Equation (32) is used to get the approximate distance between the nodes. This approach simplifies the computation procedure of the adapted hop algorithm and, also, reduces the distance divergence between the nodes.

$$d_{ui} = AHS_i^{best} \times h_{ui} \tag{32}$$

where AHS_i^{best} refers to the ideal AHS of the beacon node i, which is determined using a weighted iteration technique.

3.6 Proposed Hybrid Self-Organizing Neural Networks for Attack Detection

The self-organizing map (SOM) was first conceptualized by Kohonen(in Figure-2). The method classifies and reorders the data in the map based on the similarity of the input information. Formally, the input data vectors X_i are established, where i ranges from 1 to n, representing the number of Kohonen neurons R_j, with j ranging from 1 to m. Additionally, the number of iterations is determined as t, where t ranges from 1 to z.

The training phase involves a certain number of repetitions known as epochs. Each epoch represents a full cycle of the training algorithm, during which the network learns from all the input data vectors.

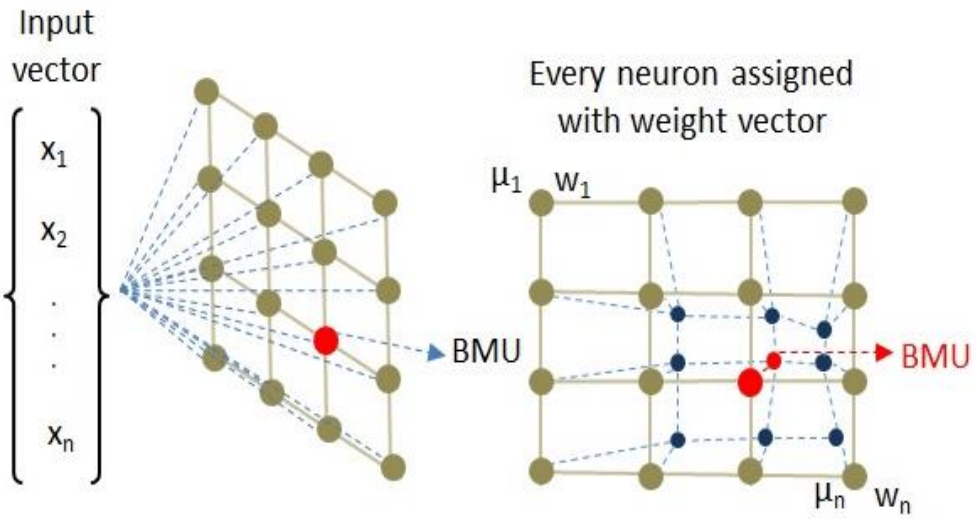

Figure 2: Architecture of the Hybrid Self-Organizing Map [31]

The input data vectors X_i and the neurons' codebook vectors R_j can be compared using Manhattan, Tanimoto, Bray Curtis, Canberra, and Chebyshev distances. Nevertheless, the Euclidean distance often yields somewhat superior categorization outcomes. Let $R_c(t)$ denotes as the value at iteration t, where

$$c = \min_j\{\|x_j(t) - R_j(t)\| \tag{33}$$

The neuron that best matches the grid input vector $x_j(t)$ at iteration t is $R_c(t)$. After selecting the winning neuron, it and its neighbors are modified as.

$$R_j(t+1) = R_j(t) + h_{jc}(t)[x_j(t) - R_j(t)] \tag{34}$$

where the neighborhood function is $h_{jc}(t)$. The mathematical structure of the neighborhood function determines the rate of change at multiple neurons surrounding the winning neuron. This function is vital to SOM networks since it preserves input topology. The Gaussian neighborhood function is the most often utilized among many neighborhood functions in SOM neural networks.

$$h_{jc}(t) = \alpha(t).\exp\left(\frac{\|r_j - r_c\|^2}{2\sigma(t)^2}\right) \tag{35}$$

where the learning rate $\alpha(t)$ is a monotonically decreasing function in every iteration t, whereas the neighborhood function $\alpha(t)$ represents the width, which likewise decreases over the training process. The variable r_j represents the location of neuron j.

It is necessary to select the activation function based on the specific prediction task. The activation function in the proposed framework is efficiently calculated using the Adam method. The probability of class j, where $j\epsilon\{1,2,\dots c\}$, given the evidence y, is calculated as $prob(cls_j|y)$.

The SoftMax function meets the posterior probability function's conditions, as shown below:

$$prob(cls_j|y) = softmax(v^L) = \frac{e^{v_j^L}}{\sum_{k=1}^{c} e^{v_k^L}} \tag{36}$$

where the element in the activation of the vector v^L is represented by v_j^L.

The following equation shows how the optimized cost function is used for training:

$$Q(W^1, bias^1) = \frac{1}{S}\sum_{a=1}^{S} Q_{CE}\left(W^1, bias^1, Y_a, l_f\right) + \beta|W^1|_F^2 \quad (37)$$

Ultimately, the anticipated categorized label can be generated, as seen in the following format:

$$Q_{CE}(W^1, bias^1, Y_a) = -\sum_{k=1}^{C} y_a[F_m^1] \quad (38)$$

$$cls_f = Q_{CE}\left(W^1, bias^1, l_f\right) \quad (39)$$

The proposed system accurately classifies normal and attacking events based on this prediction procedure.

Table 1: Network Configuration

Layers	configuration
2D grid	10x10
Weight Initialization	-1 and 1
Learning Rate	0.1
Hidden layer	64
Dropout rate	0.5
Activation Function	softmax
Learning Rate	0.001
Batch Size	32

4. Experimental Setup

The proposed hybrid self-organizing neural network (Hyb_SONN) was simulated in NS-2. The hardware used for optimal data processing and model training had the following details: A CPU from the Intel Core i7 8th Generation or a more advanced version, operating at a minimum clock speed of 3.5 GHz. The NVIDIA GeForce RTX 2080 Ti GPU with 11 GB of GDDR6 VRAM was chosen for its deep-learning capabilities. The machine had 16 GB of 2400 MHz DDR4 RAM and a 500 GB solid-state drive. The database held datasets, code, and model checkpoints.

The following software supported the hardware tools: The PC ran Windows 10 64-bit. The simulation environment of the proposed Hyb_SONN scheme comprised 500 sensor nodes with a dimension of 100*100 square meters. The base station of the system was reputed to be positioned at the corner of the network and with beacon nodes. The simulation factors employed for the accomplishment of the recommended Hyb_SONN and the benchmarked FFNN [22], CNN [23], and DNN [27] approaches are analyzed below.

In this work, the suggested Hyb_SONN model was validated by calculating parameters such as accuracy, precision, and recall. The performance of the security system was determined by these parameters, which were evaluated using the subsequent models.

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (40)$$

$$precision = \frac{TP}{TP+FP} \quad (41)$$

$$recall = \frac{TN}{TP+FP} \quad (42)$$

Dataset description

- *Sybil Attack*- a dataset with 50,000 records simulating a Sybil attack scenario. Here's the plan for each record in the dataset:

1. ID: A unique sequential number.
2. Timestamp: Randomized date and time.
3. User_ID: Randomly assigned, with a pattern to simulate normal and Sybil identities.
4. Action_Type: Randomly chosen from a predefined list of actions like login, transaction, comment, etc.
5. Is_Sybil: Randomly assigned with a bias towards normal interactions but with a significant portion flagged as Sybil actions to simulate an attack.
6. IP_Address: Randomly generated, with Sybil actions often sharing IPs to reflect the nature of the attack.

- *Dos/DDos attack:*

CICDDoS2019 contains benign and the most up-to-date common DDoS attacks, which resembles the true real-world data (PCAPs). It also includes the results of the network traffic analysis using CICFlowMeter-V3 with labeled flows based on the time stamp, source, and destination IPs, source and destination ports, protocols and attack (CSV files).

We employed a [specific split, e.g., 70:30] ratio, with [70%] of the data used for training and [30%] for testing. To ensure robustness and avoid overfitting, we implemented [k-fold cross-validation (e.g., 5-fold or 10-fold)], where the model was trained and tested across multiple subsets of the data. This approach ensures that the results are consistent and not influenced by the random distribution of data.

- **Cross-Validation Methods**: We employed [k-fold cross-validation (e.g., 5-fold or 10-fold)], which splits the dataset into k subsets. The model is trained on k-1 subsets and tested on the remaining subset, rotating through all subsets. This approach helps ensure the model is evaluated on different portions of the dataset, reducing the risk of overfitting to specific subsets.
- **Validation on Unseen Data**: To further prevent overfitting, we reserved a separate portion of the dataset as a validation set, distinct from both the training and testing sets. This unseen validation set was used to evaluate the model's generalization capabilities and fine-tune hyperparameters without influencing the final test results.
- **Regularization Techniques**: We incorporated regularization methods, such as L2 regularization (ridge), L1 regularization (lasso), or dropout, to penalize complex models and prevent overfitting. For example, dropout layers were applied with a dropout rate of [mention percentage] to randomly deactivate a portion of neurons during training, encouraging the model to learn more robust patterns rather than memorizing the training data.

Table 2 : Comparison of Accuracy

Attack Type	FFNN	CNN	DNN	Hyb_SONN
Sybil Attack	97.5	95.3	91.3	98.9
DoS	98	94	92	99
DDoS	97.3	94.2	92.4	99.4
Botnet Attack	96.3	95	94.3	98.4
Brute Force Attack	98	93.5	93	98.8

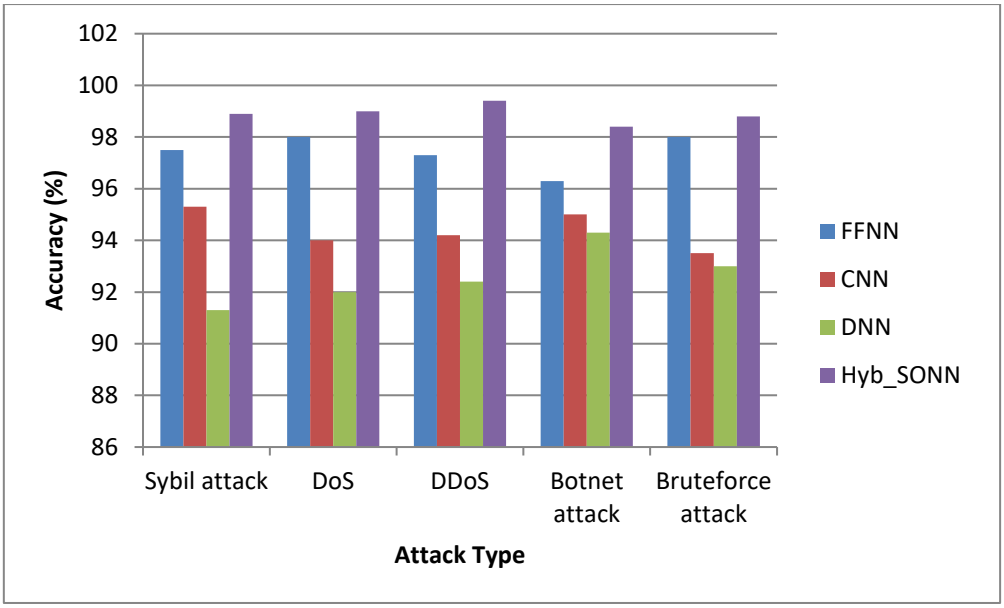

Figure 3: Comparison of Accuracy

Figure-3 illustrates the accuracy of the existing FFNN, CNN, and DNN models with the new Hyb_SONN model. The x-axis is the assault type, while the y-axis is the corresponding numbers expressed in percentages. When compared, existing methods achieve 98.8%, 95.3%, and 92.5% accuracy, while the proposed method achieves 99.8% of accuracy, which is 10%,4.3%, and 7.3% better than the aforementioned methods.

Table 3 : Comparison of Precision

Attack Type	FFNN	CNN	DNN	Hyb_SONN
Sybil Attack	85	70.4	79	97.5
DoS	84.6	71	78.5	96
DDoS	83	72.5	76.4	97.3
Botnet Attack	85.6	73	79.3	97.2
Brute Force Attack	84	73.6	76	96.8

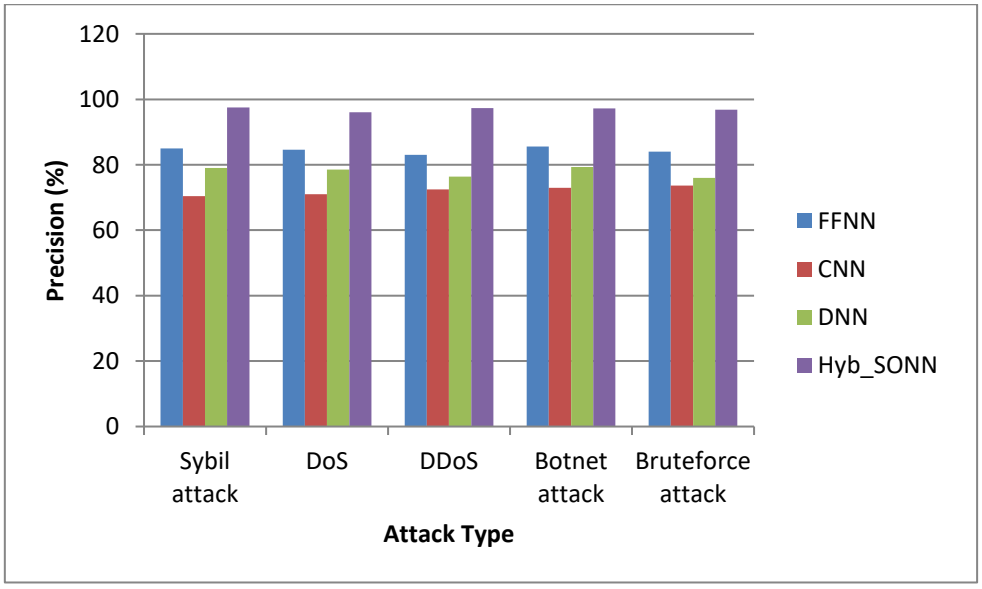

Figure 4: Comparison of Precision

Figure-4 illustrates the precision of the existing FFNN, CNN, and DNN methods with the proposed Hyb_SONN. The x-axis is the assault type, while the y-axis is the corresponding numbers expressed in percentages. When compared, the existing methods achieve 85.3%, 72.6%, and 79.3% of precision, while the proposed method achieves 96.4% of precision, which is 11.1%, 24.2%, and 18.1% better than the aforementioned methods.

Table 4 : Comparison of Recall

Attack Type	FFNN	CNN	DNN	Hyb_SONN
Sybil Attack	82	84.3	80.9	94
DoS	81.3	83	81	95.4
DDoS	84	83.8	82.5	93.5
Botnet Attack	83.4	85	83	94.2
Brute Force Attack	82	84.8	83.4	94.6

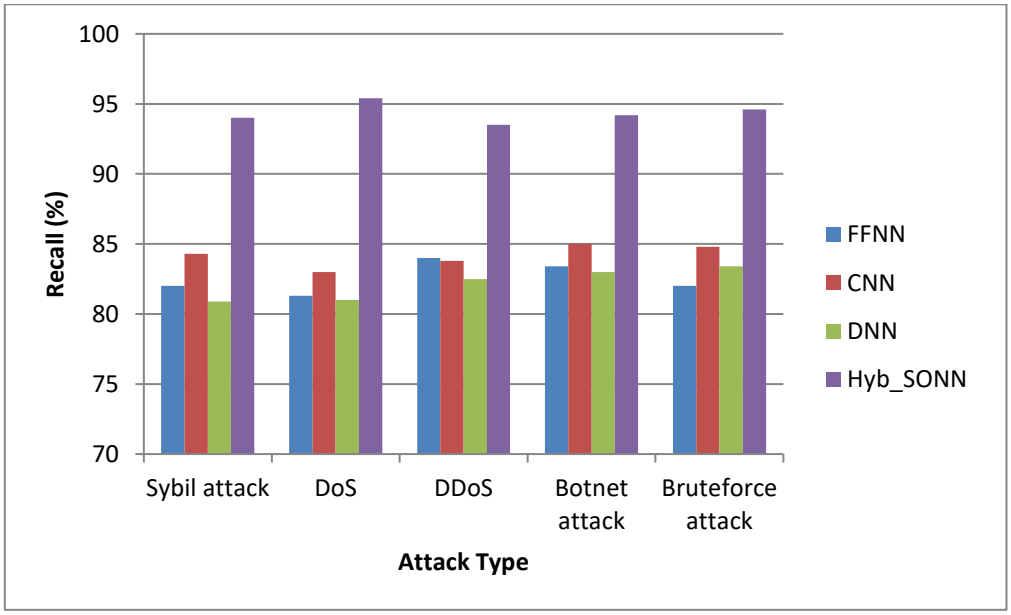

Figure 5: Comparison of Recall

Figure-5 illustrates the recall comparison between the existing FFNN, CNN, and DNN models with the proposed Hyb_SONN model. The x-axis denotes the assault type, while the corresponding y-axis shows the values expressed in percentages. When compared, existing methods achieve 81.4%, 84.5%, and 81.3% of recall, while the proposed method achieves 94.5% , which is 13.1%, 10%, and 10.2% better than the aforementioned methods.

4.1 Packet Delivery Ratio (PDR)

PDR is the rate at which data packets are successfully sent from their originating network node to their final destination.

$$PDR = \frac{numberofpacketreceivedsuccesfully}{Totalnumberofpacketsforwarded} \quad (43)$$

Table 5: Comparison of Packet Delivery Ratio (PDR)

Number of Packets	FFNN	CNN	DNN	Hyb_SONN
20	77	82	64.5	98.5
40	78.4	81.4	66	97
60	77.3	83.4	65.3	97.4
80	78.4	82.6	63.8	96.4
100	79	82	64	98

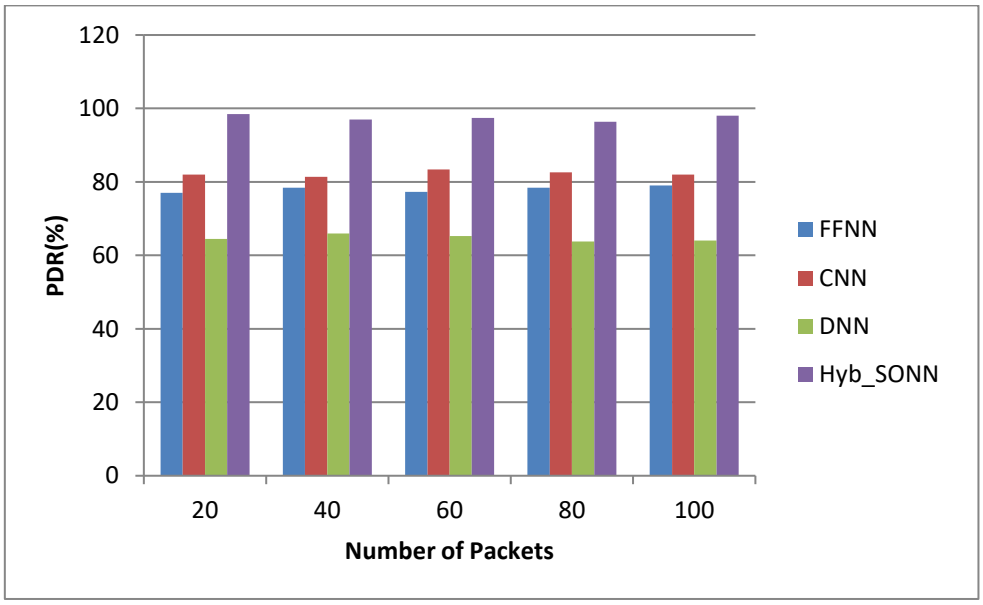

Figure 6: Performance of Packet Delivery Ratio (PDR)

Figure-6 illustrates the evaluation of the Packet Delivery Ratio (PDR) between the existing FFNN, CNN, and DNN methods with the proposed Hyb_SONN. The x-axis is the number of packets, whereas the y-axis is the results acquired in percentages. When compared, the existing methods achieve 78.3%, 81.4%, and 65.4% of PDR, while the proposed method achieves 98.5% of PDR, which is 20.2%, 18.1%, and 37.1% better than the aforementioned methods.

4.2 Energy Consumption

This is quantified as the cumulative energy of all hops and is calculated as

$$Energy = \frac{1}{p}\sum_{n}^{p} E_n \quad (44)$$

In the multi-hop routing, p denotes the number of hops, and E_n denotes the energy of the n^{th} hop.

Table 6 : Comparison of Energy Consumption

Number of Packets	FFNN	CNN	DNN	Hyb_SONN
20	67	87	78	43
40	68.4	86.4	77.5	42.4
60	66	85.8	76	46
80	66.9	87	76.9	46.7
100	68	87.9	74	44

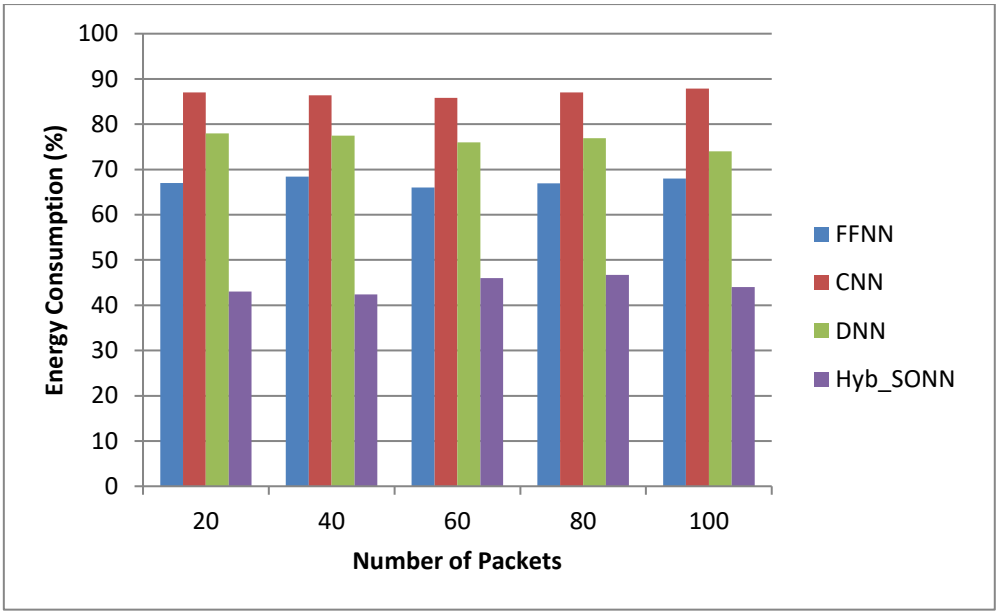

Figure7: Comparison of Energy Consumption

Figure-7 displays the energy consumption between the existing FFNN, CNN, and DNN methods with the proposed Hyb_SONN. The x-axis is the quantity of packets, whereas the y-axis displays the results in percentages. When compared, the existing methods achieve 68.2%, 87.3%, and 78.3% of energy consumption, while the proposed method achieves 43.2%, which is 25%, 43.1%, and 35.1% less than the aforementioned methods.

4.3 Localization Error

Table 7 : Comparison of Localization Error

Number of Packets	FFNN	CNN	DNN	Hyb_SONN
20	53	64.4	45	13
40	54	65	46.3	14.4
60	54.6	65.7	44	13.8
80	52	66	44.8	12.6
100	55	67.3	45	13.1

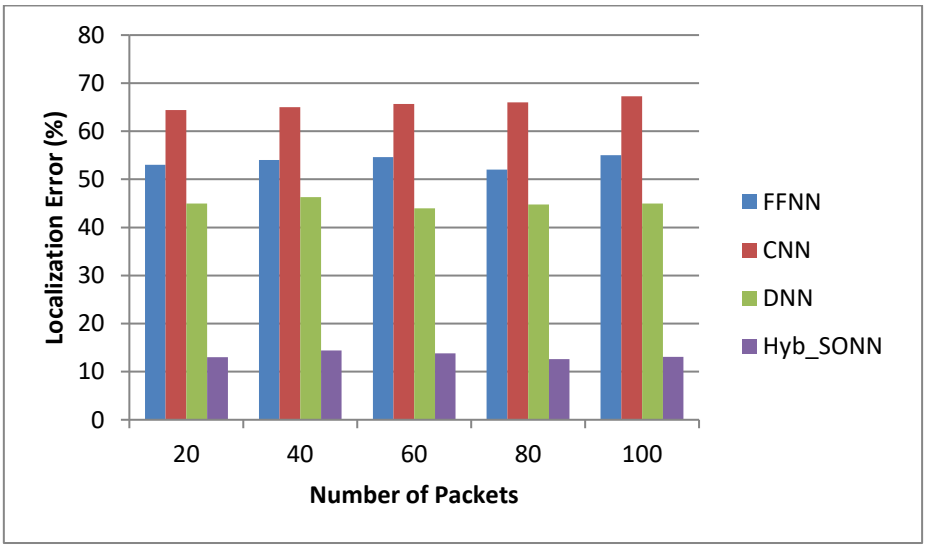

Figure 8: Comparison of Localization Error

Figure-8 depicts the localization error comparison between the existing FFNN, CNN, and DNN methods with the proposed Hyb_SONN. The x-axis is the number of packets, whereas the y-axis displays the results in percentages. When compared, the existing methods achieve 54.9%, 64.8%, and 45.2% of localization error, while the proposed method achieves 13.2% , which is 41.7%, 51.6%, and 33% less than the aforementioned methods.

Table 8: Overall Comparative Analysis

Parameters	FFNN	CNN	DNN	Hyb_SONN
Accuracy (%)	98.8	95.3	92.5	99.8
Precision (%)	85.3	72.6	79.3	96.4
Recall (%)	81.4	84.5	81.3	94.5
Packet Delivery Ratio (%)	78.3	81.4	65.4	98.5
Energy Consumption (%)	68.2	87.3	78.3	43.2
Localization Error (%)	54.9	64.8	45.2	13.2

5. Conclusion

This research introduces a method for identifying numerous assaults on WSNs using self-organizing maps. The simulation findings demonstrate that increasing the number of methods used enhanced both the accuracy and stability of target location estimation. However, this also led to a longer duration for target location estimation. A remarkable detection accuracy rate of 99.8% was attained. The usefulness and resilience of the proposed technique in identifying various threats in WSN networks were emphasized by this finding. To improve the effectiveness and practicality of the proposed technique, various avenues might be further investigated in future research. Hence, enhancing the suggested model by integrating supplementary layers or architectures that may facilitate the capture of more intricate patterns and enhance detection capabilities.

References

[1] M. Adil, M. A. Almaiah, A. Omar Alsayed, and O. J. S. Almomani, "An anonymous channel categorization scheme of edge nodes to detect jamming attacks in wireless sensor networks," *Sensors*, vol. 20, no. 8, p. 2311, April 2020.

[2] W. Elsayed, M. Elhoseny, S. Sabbeh, and A. J. C. Riad, "Self-maintenance model for wireless sensor networks," *Computer & Engineering*, vol. 70, pp. 799-812, September 2018.

[3] D. K. Sah and T. J. I. F. Amgoth, "Renewable energy harvesting schemes in wireless sensor networks: A survey," *Information Fusion*, vol. 63, pp. 223-247, July 2020.

[4] K. Sampoornam, S. Saranya, G. Mohanapriya, P. S. Devi, and S. Dhaarani, "Analysis of LEACH routing protocol in wireless sensor network with wormhole attack," *in 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, pp. 147-152: IEEE, March 2021.

[5] A. Shahraki, A. Taherkordi, Ø. Haugen, and F. J. C. N. Eliassen, "Clustering objectives in wireless sensor networks: A survey and research direction analysis," *Computer Networks*, vol. 180, p. 107376, June 2020.

[6] M. Numan, F. Subhan, W. Z. Khan, S. Hakak, S. Haider, G. T. Reddy, A. Jolfaei, and M. Alazab, "A systematic review on clone node detection in static wireless sensor networks," *IEEE Access*, vol. 8, pp. 65450-65461, March 2020.

[7] A. Ali, Y. Ming, S. Chakraborty, and S. J. F. i. Iram, "A comprehensive survey on real-time applications of WSN," *Future Internet*, vol. 9, no. 4, p. 77, November 2017.

[8] D. Kandris, C. Nakas, D. Vomvas, and G. Koulouras, "Applications of wireless sensor networks: an up-to-date survey," *Applied System Innovation*, vol. 3, no. 1, p. 14, February 2020.

[9] Y. Liu, M. Ma, X. Liu, N. N. Xiong, A. Liu, and Y. Zhu, "Design and analysis of probing route to defense sink-hole attacks for Internet of Things security," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 1, pp. 356-372, November 2018.

[10] M. Premkumar, and T. J. M. Sundararajan, "DLDM: Deep learning-based defense mechanism for denial-of-service attacks in wireless sensor networks," *Microprocessor and Microsystems*, vol. 79, p. 103278, September 2020.

[11] M. S. Yousefpoor, E. Yousefpoor, H. Barati, A. Barati, A. Movaghar, and M. Hosseinzadah, "Secure data aggregation methods and countermeasures against various attacks in wireless sensor networks: A comprehensive review," *Journal of Networks and Computer Applications*, vol. 190, p. 103118, June 2021.

[12] O. R. Ahutu and H. J. I. A. El-Ocla, "Centralized routing protocol for detecting wormhole attacks in wireless sensor networks," *IEEE Access*, vol. 8, pp. 63270-63282, March 2020.

[13] S. S. Narayanan and G. J. C. Murugaboopathi, "Modified secure AODV protocol to prevent wormhole attack in MANET," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 4, p. e5017, February 2020.

[14] F. A. Alenezi, S. Song, and B.-Y. Choi, "WAND: wormhole attack analysis using the neighbor discovery for software-defined heterogeneous internet of things," *in 2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1-6: IEEE, July 2021.

[15] M. N. Siddiqui, K. R. Malik, and T. S. Malik, "Performance analysis of blackhole and wormhole attack in MANET based IoT," *in 2021 International Conference on Digital Futures and Transformative Technologies (ICoDT2)*, pp. 1-8: IEEE, June 2021.

[16] A.-u. Rehman, S. U. Rehman, and H. J. W. P. C. Raheem, "Sinkhole attacks in wireless sensor networks: A survey," *Wireless Personal Communications*, vol. 106, pp. 2291-2313, 2019.

[17] D. M. Khan, T. Aslam, N. Akhtar, S. Qadri, I. M. Rabbani, and M. Aslam, "Black hole attack prevention in mobile ad-hoc network (MANET) using ant colony optimization technique," *Information Technology and Control*, vol. 49, no. 3, pp. 308-319, September 2020.

[18] S. Thakur, and S. J. I. J. o. A. C. Dalwal, "Mitigating gray hole attack in mobile ad hoc network using artificial intelligence mechanism," *International Journal of Advanced Computronics and Management Studies (IJACMS)*, vol. 3, no. 5, pp. 1-13, 2019.

[19] M. Mounica, R. Vijayasaraswathi, and R. Vasavi, "RETRACTED: Detecting Sybil Attack in Wireless Sensor Networks Using Machine Learning Algorithms," *in IOP Conference Series: Materials Science and Engineering*, vol. 1042, no. 1, p. 012029: IOP Publishing, 2021.

[20] H. H. Bosman, G. Iacca, A. Tejada, H. J. Wörtche, and A. J. I. F. Liotta, "Spatial anomaly detection in sensor networks using neighborhood information," *Information Fusion*, vol. 33, pp. 41-56, January 2017.

[21] Q. E. Alahy, M. N.-U.-R. Chowdhury, H. Soliman, M. S. Chaity, and A. Haque, "Android malware detection in large dataset: smart approach," *in Advances in Information and Communication: Proceedings of the 2020 Future of Information and Communication Conference (FICC)*, Volume 1, pp. 800-814: Springer, February 2020.

[22] M. Ezhilarasi, L. Gnanaprasanambikai, A. Kousalya, and M. J. S. C. Shanmugapriya, "A novel implementation of routing attack detection scheme by using fuzzy and feed-forward neural networks," *Soft Computing*, vol. 27, no. 7, pp. 4157-4168, 2023.

[23] Y. Liu, D. Sun, R. Zhang, and W. J. S. Li, "A method for detecting LDoS attacks in SDWSN based on compressed Hilbert–Huang transform and convolutional neural networks," *Sensors*, vol. 23, no. 10, p. 4745, May 2023.

[24] Ö. J. C. Kasim, "A Robust DNS flood attack detection with a hybrid deeper learning model," *Computers and Electrical Engineering*, vol. 100, p. 107883, March 2022.

[25] M. Vishwakarma and N. J. D. A. J. Kesswani, "DIDS: A Deep Neural Network based real-time Intrusion detection system for IoT," *Decision Analytics Journal*, vol. 5, p. 100142, November 2022.

[26] P. V. de Campos Souza, E. Lughofer, and H. J. S. Rodrigues Batista, "An explainable evolving fuzzy neural network to predict the k barriers for intrusion detection using a wireless sensor network," *Sensors*, vol. 22, no. 14, p. 5446, July 2022.

[27] V. Gowdhaman and R. J. S. C. Dhanapal, "An intrusion detection system for wireless sensor networks using deep neural network," *Soft Computing*, vol. 26, no. 23, pp. 13059-13067, 2022.

[28] S. Sinha and A. J. W. p. c. Paul, "Neuro-fuzzy based intrusion detection system for wireless sensor network," *Wireless Personal Communications*, vol. 114, no. 1, pp. 835-851, April 2020.

[29] P. Nancy, S. Muthurajkumar, S. Ganapathy, S. Santhosh Kumar, M. Selvi, and K. J. I. C. Arputharaj, "Intrusion detection using dynamic feature selection and fuzzy temporal decision tree classification for wireless sensor networks," *IET Communications*, vol. 14, no. 5, pp. 888-895, March 2020.

[30] Hossain, M. A., Haque, M. A., Ahmad, S., Abdeljaber, H. A., Eljialy, A. E. M., Alanazi, A., ... & Nazeer, J. "AI-enabled approach for enhancing obfuscated malware detection: a hybrid ensemble learning with combined feature selection techniques,". *International Journal of System Assurance Engineering and Management*, 1-19, 2024.

[31] B. Senzio-Savino, M. R. Alsharif, C. E. Gutierrez, and K. Setarehdan, "An Online Synchronous Brain Wave Signal Pattern Classifier with Parallel Processing Optimization for Embedded System Implementation," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 1, 2017.