International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9s
Article Received: *30 April 2023* ***Revised:*** *30 June 2023* ***Accepted:*** *20 July 2023*

Improved Accurate Localization using PSO and the Weighted Dijkstra Algorithm in Software Defined Wireless Sensor Networks

Baraa Abbas Shahal [1]* Mohammed Najm Abdullah [2]

[1,2] *University of Technology-Iraq*
* ce.21.07@grad.uotechnology.edu.iq

Abstract: Wireless Sensor Networks (WSNs) are crucial in various fields, including monitoring the environment, surveillance, and healthcare. They rely on localization services for accurate data transfer and optimal network performance. Traditional WSN techniques struggle to adapt to dynamic environmental changes beyond the intended task scope. A synergy between Software-Defined Networking (SDN) and WSN has been suggested to address this issue. This research paper presents proposed approach for machine learning-based localization in Software Defined Wireless Sensor Networks (SDWSNs) using Particle Swarm Optimization (PSO) technique and the Weighted Dijkstra algorithm. PSO technique is used for clustering, the weighted Dijkstra algorithm (WDA) for finding the shortest path and sending data packets, and machine learning algorithms like AdaBoost and Naïve Bayes for data classification. The effectiveness of the proposed approach is measured using energy consumption, throughput, network lifespan, and packet delivery ratio, outperforming existing models like OEERP, LEACH, DRINA, and BCDCA. The machine learning algorithms' performance is also evaluated, with Naïve Bayes achieving the highest accuracy of 78.24% and AdaBoost 98.90%.

Keywords: Software Defined Wireless Sensor Networks, Localization, Clustering, Particle Swarm Optimization, Weighted Dijkstra Algorithm, Machine Learning Algorithms.

1. Introduction

Wireless sensor networks (WSNs) are crucial in the IoT future for providing connection, control, security, and awareness of assets of varying sizes, locations, and connectivity types. By accurately correlated sensor data with location, WSNs improve network resource utilization and interpretation accuracy, making them essential for various applications [1-2]. WSN is a collection of sensor node deployments for specific applications, allowing for self-organization and task-specific network models. Sensor nodes operate on constrained batteries, affecting energy efficiency. In hazardous areas, battery replacement is challenging [3].

We found the motivation to ensure balanced energy consumption between sensor nodes and Base Stations (BS) using clustering protocols in the research papers that published in this journal [4].

WSNs face challenges like limited energy supply, fault tolerance, scalability, production costs, hardware constraints, topology maintenance, transmission media, and power consumption. To address all these challenges, it was proposed to use Software Defined Networking (SDN) paradigm which simplifies network management and configuration by presenting high-level abstractions for networking concepts. The network is divided into data, control, and application planes, with controllers having a global view and selecting appropriate allocation algorithms for network lifetime, processing power, or data exchange [5-6].

The SDWSN paradigm addresses management complexity in current WSN deployments by adding new functionalities and centralizing network intelligence in an SDWSN controller. This approach is crucial for large WSNs with thousands of sensor nodes, allowing sensor nodes to function as forwarding devices [7].

This paper provides the following contributions for this research work:

1. The study proposes a new approach for localization in SDWSN, combining Particle Swarm optimization (PSO) with the Weighted Dijkstra algorithm, to address limitations in accuracy, scalability, and energy efficiency in finding accurate locations.
2. To apply machine learning algorithms, specifically Naïve Bayes and AdaBoost , for data classification in SDWSN. The research illustrates how accurate classification may be performed using these techniques, which in turn leads to enhanced decision-making and improved network performance.

The remainder of the paper is structured as follows. In the subsequent section, we provide a concise literature review on localization in SDWSN. In Section 3 we discuss existing gaps in the related literature. Section 4 depicts explanation on developed scheme and proposed flowchart. The proposed methodologies and their resulting performance analysis are presented in Section 5. Finally, Section 6 concludes this paper with a discussion of future research.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9s
Article Received: *30 April 2023* ***Revised:*** *30 June 2023* ***Accepted:*** *20 July 2023*

2. Literature survey

The control plane in SDWSN networking architecture offers a thorough perspective of the network, which makes it perfect for localization and resolving location estimation equations. However, because this area is new, there hasn't been much effort done in it, and implementations and uses differ [8].

The traditional distance estimation methods can be used during localization in a SDWSN, but their implementations and use will be varied, like Y. Zhu et al. [9] proposed a localization node selection algorithm using the Cramer-Rao lower bound and SDN controller knowledge, aiming for energy satisfaction and significant improvement in localization performance, The authors found a 45% increase in node accuracy allocation using this method that requires 22 anchor nodes for localization of 50 unknown nodes. In [10] Y. Zhu et al. introduced an anchor scheduling scheme for localization in wireless sensor networks (WSNs). They proposed a centralized scheme based on software-defined networking (SDN) to minimize active anchors and prolong network lifetime. Simulations show this scheme reduces active anchors, ensures desired number of anchors, and reduce energy usage; but the plan can slightly reduce localization accuracy, after that Y. Zhu et al. [11] developed an improved localization algorithm for software-defined sensor networks (SDSNs) using a node-selection strategy under network power constraints. This algorithm uses the Software-Defined Networking (SDN) technique for centralized control and uses the Cramer-Rao lower bound (A-CRLB) value to evaluate each node's contribution to localization accuracy. The algorithm is applied to both cooperative and noncooperative localization scenarios, proving efficient and effective in improving selection convergence speed and localisation accuracy. In [12] Y. Zhu et al. suggested a node scheduling scheme for localization in heterogeneous SD-WSNs. The scheme uses the software-defined networking paradigm and calculates anchor state using a flow table via sensor OpenFlow. Simulations show that the scheme reduces active nodes while ensuring an expected number of anchors and reduces energy consumption with slight positioning accuracy. In [13] Y. Zhu et al. introduced a node selection algorithm and an NLOS mitigation algorithm to improve the cooperative localization algorithm. The node selection algorithm selects informative reference nodes for agents, reducing energy consumption. The NLOS mitigation algorithm penalizes NLOS errors based on link quality, enhancing localization accuracy in severe NLOS environments, however, the suggested node-selection algorithm only takes into account static NLOS scenarios when choosing the reference nodes. In [14] O. P. Cloete et al. compared and implemented three localization algorithms using IT-SDN in a contiki-os environment: Trilateration, Maximum likelihood estimation, and Linear Least Square Localization. The simulation results shows that these algorithms moving localization calculation from the data-plane to the control-plane doesn’t improve performance. Furthermore, the experiment led to unnecessary energy consumption.

3. Problem statement

SDWSNs are networks of sensors used to gather data from a given region. SDWSN applications which includes " monitoring of environmental, tracking of target, and disaster management" rely on these nodes' precise positioning. Traditional localization approaches, such as triangulation and trilateration, are imprecise, inefficient, and inefficient in terms of scalability and energy efficiency.

The challenge is to develop a machine learning-based localization technique for SDWSNs that can overcome these constraints while still offering accurate and efficient node localization. The objective is to use the power of SDN and ML approaches to increase localization accuracy, minimize energy consumption, and deal with the dynamic nature of SDWSNs. Existing localization approaches based on range measurements or signal intensity frequently degrade in accuracy owing to environmental variables such as multipath fading, signal attenuation, and interference. Overcoming these challenges and achieving high localization accuracy is essential. Data plane in SDWSNs is typically powered by batteries, and energy efficiency is crucial to prolong the network lifetime. Localization techniques should minimize energy consumption by reducing the number of control messages, optimizing sensor node movement, or using energy-aware algorithms. WSNs can contain a large number of sensor nodes, so localization algorithms must be scalable enough to manage networks with hundreds or thousands of nodes. The computational and communication overheads should be minimized to enable efficient localization in large-scale SDWSNs deployments. They are often deployed in dynamic environments where nodes can move or be added/removed over time. Localization algorithms should adapt to changing environments and give real-time node position updates.

4. Existing gaps

Various research gaps in the present work are described in this section. They are as follows:

- Most studies use simulations or limited experimental datasets, and more research is needed to validate proposed algorithms in real-world SDWSNs deployments. Experiments in diverse environments with noise, obstacles, and mobility patterns could provide more realistic insights.
- Labeled training data is crucial for ML model development, but it can be laborious and expensive. Further research is needed to improve localization accuracy and reduce training data quantity.
- Energy consumption is a crucial factor in Wireless Sensor Networks (WSNs) due to limited sensor node resources. Further research is needed to design energy-efficient models, optimize feature extraction and selection, and improve inference methods.
- Most reviewed articles assume sensor nodes are inert or anchor nodes are controlled. However, real-world scenarios may involve dynamic movement. Future research should

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9s
Article Received: *30 April 2023* ***Revised:*** *30 June 2023* ***Accepted:*** *20 July 2023*

focus on developing localization algorithms using machine learning to adapt to these environments.

5. Research methodology

5.1 Proposed system model

SDN technology is used in the suggested approach, as seen in Figure 1, to determine each node's precise location and network routing. In the base station (BS) is the controller that determines the network's overall routing. It is sufficient for sensor nodes to send the BS the topological data. The energy usage and processing overhead in the sensor node are decreased by this technology.

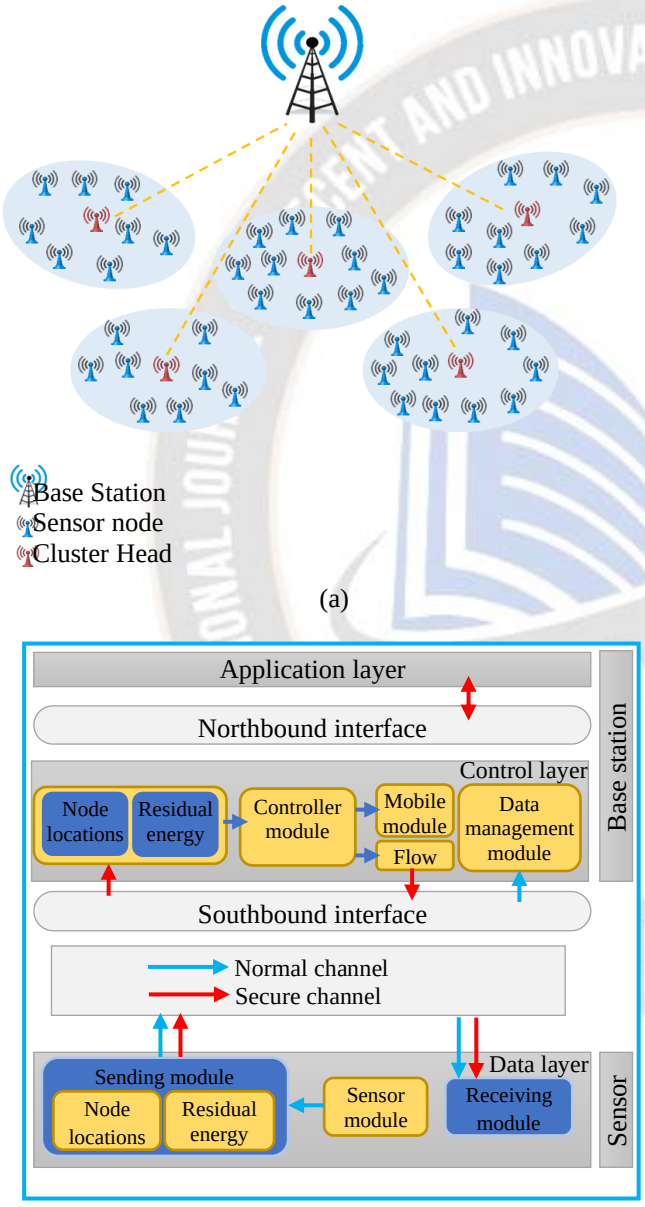

Figure. 1 System model: (a) Overview of an SDWSN framework and (b) architecture details

5.2 Proposed methodology

The proposed study works to find an effective rout to obtain the best location and improve network performance by making location calculations take place in the SDN and finding for the process of proper data transmission. The propose model is designed using the optimization technique based on machine learning. Initially, we have generated the simulation data. In the sensing area, we are generating 50 random nodes in the area of 200*200 square meter. After generating the nodes, PSO (Particle Swarm Optimization) technique is used for clustering. This algorithm assigned the nodes to the clusters and determine the cluster centers depends on cluster radius, also draw the cluster circles. So, we can visualize the clusters and can understand the cluster distribution within the sensing area. Then, perform the cluster head selection in which final cluster heads are determined using gravitational force estimation.

After performing the cluster head selection, Weighted Dijkstra algorithm is applied to find the shortest path for packet transmission and generates data from routing and data transmission to calculate network lifetime, energy consumption, PDR, and throughput. Machine learning-based algorithms like naïve bayes and AdaBoost algorithm are applied for classification, and performance metrics are calculated to assess accuracy, sensitivity, specificity, and Kappa.

5.3 Proposed approaches

5.3.1 Particle Swarm Optimization (PSO)

PSO is a "stochastic optimization" method that simulates social activity like fish schooling or bird flocking. Particle swarms (also known as people) in PSO systems move rapidly around the search area. A solution to the optimization problem is represented by each particle. A particle's current position is affected by its best previous experiences and the best past experiences of nearby particles. The gbest PSO is a technique for finding the ideal position within a particle's range for the whole Swarm. This approach is frequently referred to as the best PSO when employing more compact localizations. A fitness function that varies based on the optimization problem evaluates the performance of each particle [15].

The following traits characterize each swarm particle: x_{id} represents the particle's current position, v_{id} its current velocity, and p_{id} its best-ever position. Particle i is greatest location is the finest location it has ever been to. The best vector from each neighbor can be remembered using either lbest or gbest. Each local version particle remembers the optimal vector lbest achieved by its topological companions [16]. The optimal vector gbest for the global version is decided by the collective intelligence of all swarm members. Therefore, the gbest model includes the lbest model. The velocity and position are updated using Eq. (1) and (2)

$$v_{id} = wv_{id} + c_1 rand()(p_{id} - x_{id}) + c_1 Rand()(p_{gd} - x_{id}) \quad (1)$$

$$x_{id} = x_{id} + v_{id} \quad (2)$$

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9s
Article Received: *30 April 2023* ***Revised:*** *30 June 2023* ***Accepted:*** *20 July 2023*

Where c_1 and c_2 represent two cognitive coefficients, rand () and Rand () represent two random real values.there are various communication topologies for particle interaction. The ideal solution is reached by switching to a star topology in which all particles are in constant contact with one another. The star topology outshines any alternative. Using inertia w, we are able to strike a balance between probing new areas and capitalizing on known ones in the search space. The dynamic reduction of inertia during travel promotes a state of balance between the search space's exploration and utilization [17-18].

Algorithm 1: PSO for clustering

Input: Set of n_1, n_2, n_3, n_4…,n_n.
Output: cluster head (CH) for the swarm.
Step 1: Determine the maximum number of iterations.
Step 2: Randomly initialize the position of N particles X_i when.
Step 3: Pick the parameter values: W, ($C_1$and C_2) and (r_1 and r_2)
Step 4: For Iteration in range (max_iter):
For each particle in range (N):
a. Compute new velocity of particle using Eq. (1)
b. Compute new position of the particle using its new velocity as shown in Eq. (2)
c. If position is not in range [min, max] then clip it:
if x_i < min:
x_i = minx
else if x_i > max:
x_i = max
d. Update new best of this particle and new best of Swarm
$P_{best,i}=x_i$ if f_i better f_{pbest}
$g_{best,i}=p_{best,i}$ if $f_{pbest,i}$ better f_{gbest}
Step 5: determine the cluster head (CH) for the swarm, when $CH=g_{best,i}$.

5.2.2 Weighted Dijkstra Algorithm

The Weighted Dijkstra Algorithm (WDA) is an extension of Dijkstra's algorithm that takes into account the weights or costs associated with the edges of a graphite is used to determine the shortest path among a source nodes and every other node in a weighted graph [19]. Here's an explanation of the Weighted Dijkstra Algorithm:

Algorithm 2: WDA for shortest path

Input: clustered nodes
Output: shortest path connecting each source node to all other nodes.
Step 1: Install the distance values of all nodes as infinity, except for the source node, which is set to 0.
Step 2: Create an empty set of visited nodes.
Step 3: While there are unvisited nodes:

- Choose the nodes with the minimum distance from the source among the unvisited nodes. This node becomes the current node.
- Mark the current node as visited.
- For each neighbour of the current node:
  - Add the source's distance to the current node's edge weight to estimate the neighbour's distance.
  - Update the neighbour's distance if the tentative distance is less.

Step 4: Once all nodes are visited, or the destination node is reached, the algorithm terminates.

5.2.3 Machine learning Algorithms

In this research, there are two machine learning algorithms used for classification, which are Naïve Bayes and AdaBoost.

5.2.3.2 Naïve Bayes Classifier

It's an approach to classification predicated on the Bayes Theorem and the idea that predictors are unrelated to one another. Naive Bayes classifiers function on the premise that each feature included in a class is independent of the other. The field of text categorization is where Naïve Bayes dominates. The conditional probability of an occurrence is used for aggregation and classification. The Naïve Bayes classifier is based on the Bayes theorem as shown in the following equation:

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \ldots \times P(x_n|c) \times P(c) \quad (3)$$

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

Where $P(c|x)$ represent Posterior Probability, $P(x|c)$ is Likelihood, $P(c)$ is Class Prior Probability, $P(x)$ *is* Predictor Prior Probability. Finally, c and x are hypothesis and data vectors respectively. To make a classification decision, the Naive Bayes algorithm find P(c/x) for each c ∈ C. The class with the highest probability is the final classification, which forms the output of the classifier [20].

Algorithm 3: Naïve Bayes classifier

Input: Training dataset.
Output: Predicted class for the test document.
Step 1: First, Read the training dataset T.
Step 2: Compute each class's predictor variable mean and standard deviation.
Step 3: Repeat the Gauss density equation to determine the probability of class fi given each predictor variable f_1, f_2, f_3..., f_n.
Step 4: Determine the probabilities associated with each group.
Step 5: Get the greatest likelihood.

5.2.3.2 AdaBoost classifier

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9s
***Article Received:** 30 April 2023 **Revised:** 30 June 2023 **Accepted:** 20 July 2023*

AdaBoost is an ensemble learning technique that uses an iterative procedure to improve weak classifiers by learning from their errors. It is also known as "Meta-learning" and uses sequential ensemble, unlike random forest's parallel ensemble. AdaBoost combines multiple weak classifiers into one, generating a powerful classifier with superior accuracy. It is an adaptive classifier that enhances performance, but may lead to overfitting in some cases. It is most effective when used to improve decision trees and base estimators, addressing binary classification challenges due to noise and outliers. AdaBoost's central concept is to accurately predict out-of-the-ordinary observations by adjusting classifier weights and training each iteration's data sample [21-22].

Algorithm 4: AdaBoost classifier

Input:
- Training dataset with labeled examples
- Weak classifier algorithm

Output: Return ensemble of weak classifiers and their weights.

Step 1: AdaBoost randomly selects a training subset.

Step 2: Selects a new AdaBoost ML training model set based on the performance of its predictions during the preceding training iteration.

Step 3: It allocates more weight to incorrectly categorized data so that there is a high probability of classification in the subsequent iteration.

Step 4: Each iteration's weight for the trained classifier is determined by the classifier's performance in the previous iteration. We will priorities the best-performing classifier.

Step 5: This procedure repeats itself until either all training data is fitted error-free, or the maximum number of estimators is achieved.

Step 6: To classify, "vote" on all the learning

6. Results and analysis

The performance of the proposed localization techniques was evaluated here. We start by describing a simulated system model in two dimensions, and then we talk about how to scale this model up to deal with localization issues in higher dimensions. We conclude with a review of the performance findings. The 50 sensor nodes in our baseline simulated network were spread out over a 200 by 200 meter area. All simulations used a completely random distribution of sensor nodes. Below, we have provided the simulation parameters in tabular form, which is as follows:

Table 1: simulation settings/parameters

Parameters	Value
Cluster Radius	30 meters
Sensing Range	36 meters
Size of Packet	512 Bytes
Number of Nodes	50
Transmission Power	0.02 Joule
Receiving Power	0.01 Joule
Initial Energy	200 Joule

6.1 Experimental results

This section presents the experimental findings after employing various methods such as the PSO technique, Weighted Dijkstra algorithm, and ML algorithms, i.e., AdaBoost and Naïve Bayes.

6.1.1 Results of the Optimization Routes

This section provides the results visualization results of the optimization routes. To find the routes, we have used PSO (Particle Swarm Optimization), and the Weighted Dijkstra algorithm. After applying these algorithms, some transmission results are found, which are represented below.

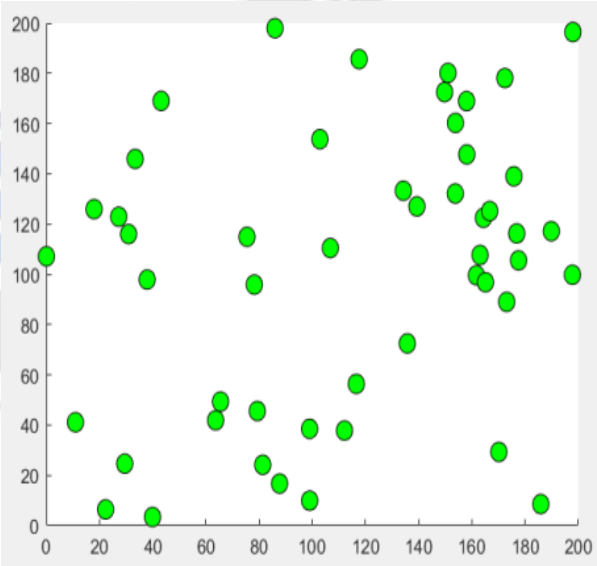

Figure 2:Nodes Spread in 200*200 Area.

The above figure of nodes dispersed throughout a 200x200 square area illustrates the distribution or placement of nodes in a two-dimensional environment. We can see in this figure there are a total of 50 nodes given, which is represented by green color. The 200x200 area is a square region measuring 200 units by 200 units. The spread or distribution of nodes within a 200x200 area reveals information about the network or system's density and coverage. It enables us to determine how densely or sparsely the nodes are dispersed throughout the space. Overall, the figure of nodes dispersed in a 200x200 area gives a visual depiction of the spatial distribution of nodes and is a useful tool for analyzing and understanding the deployment of a network or system in a specific region.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9s
Article Received: *30 April 2023* ***Revised:*** *30 June 2023* ***Accepted:*** *20 July 2023*

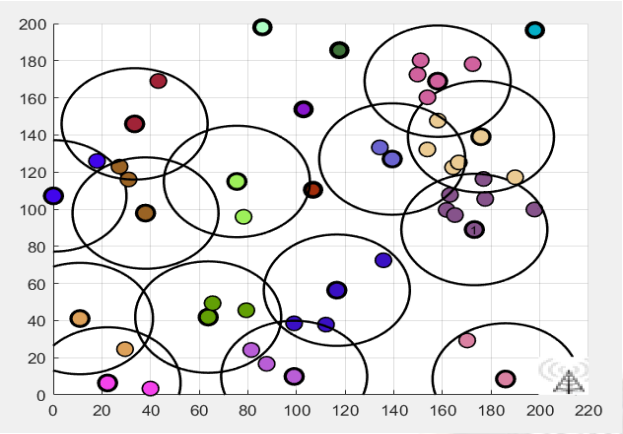

Figure 3: Nodes in localization range

Fig. 3 illustrates the nodes in the localization range in the context of WSNs and represents the spatial distribution of nodes within the localization range, considering specific parameters such as the cluster radius, sensing range, number of nodes, packet size, initial energy, transmission power, and receiving power. As shown in the previous figure, these nodes are represented by the same color, but in this figure, we can see that every node is represented by different colors. In this figure, it is clearly shown that the different clusters' circles are marked. In a clustered WSN arrangement, the cluster radius is the shortest distance between any two nodes in a cluster. Each cluster's size determines its node count. This particular cluster has a radius of 30 meters. The maximum distance over which a node can detect and receive signals from other nodes in the network is known as its sensing range. The result specifies the radius within which a node may send and receive data and communications. Here, a sensing range of 36 meters is specified. The number of nodes is already discussed in the previous figure description. In this scenario, there are 50 nodes. And the packet size of this is specified as 512 bytes. The size of a packet is the amount of data that may be sent in a single transmission, measured in bytes. Larger packet sizes necessitate more energy for transmission and may hinder network performance. In addition, the term "initial energy" refers to the amount of energy that is available to each node in the network at the start of a simulation or deployment. It reveals the node's energy reserves for running processes. The initial energy is set to 200 joules. In this scenario, 0.02 joules are shown as the transmission power. Data transmission power is the amount of energy needed to convey information over a certain distance between two nodes. The node's signal strength and its ability to communicate with other devices are diminished. Moreover, A node's power usage while receiving data from other nodes is represented by the term "receiving power." It normally uses less power than transmission. Here, 0.01 joules of receiving power is specified. The figure is useful for evaluating the efficiency of the provided WSN network in terms of coverage, energy consumption, and communication.

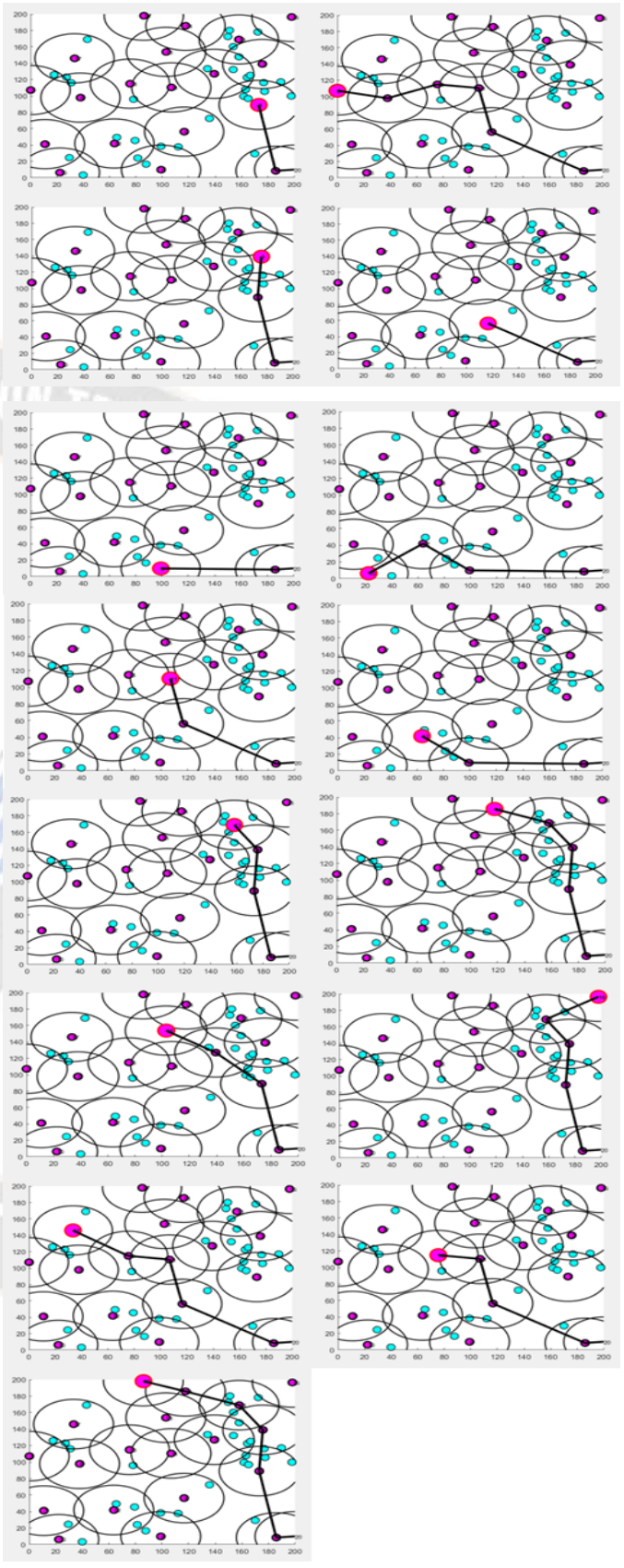

Figure 4: Different Data Transmission Plots

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9s
Article Received: *30 April 2023* ***Revised:*** *30 June 2023* ***Accepted:*** *20 July 2023*

Figure 4 shows the different data transmission plots to find the shortest routes. The shortest path is found using the Weighted Dijkstra algorithm in these plots. We can see in these plots; the shortest path is marked by a line. More detail about these graphs is already described in the previous graph description, such as cluster radius, sensing range, number of nodes, etc.

6	0	7	0	0	4	2	
7	8	5	0	0	3	3	
8	1	7	0	0	4	3	
9	12	25	0	6	13	7	1(
10	12	13	0	9	7	7	
11	3	19	0	19	10	6	
12	8	19	0	1	10	9	
13	22	37	0	0	19	15	
14	2	5	2	0	3	3	
15	49	11	0	6	6	5	
16	1	1	0	0	1	1	
17	3	8	0	0	5	3	
18	2	5	0	0	3	2	
19	0	4	0	0	3	2	
20	1	3	2	0	2	2	
21	2	1	0	0	1	1	
22	16	27	2	12	14	13	
23	9	9	0	2	5	2	
24	2	5	0	1	3	3	
25	1	3	0	0	2	2	

Figure 5: Collected data.

The above figure shows the collected data in the tabular form. This data is generated after applying the above given process in which PSO and Weighted Dijkstra algorithm is included.

6.1.2 Results of the Machine Learning Algorithms

This section presents the simulation results after applying the ML approaches such as AdaBoost and Naïve Bayes. There are some graphs and tables presented in this section, and obtain the classification results in terms of the performance metrics, which are classification accuracy, sensitivity, and specificity.

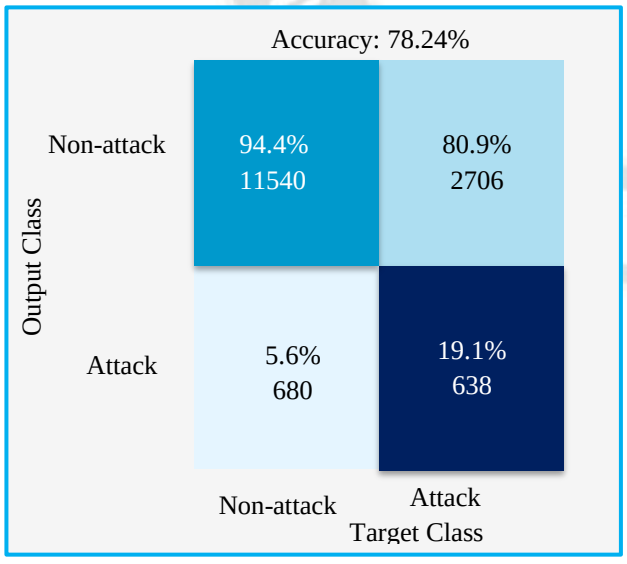

Figure 6: Confusion Matrix of the Naïve Bayes Classifier

The confusion matrix of the naive Bayes classifier, which illustrates the binary classification, is depicted in Fig.6. The x-axis of this matrix represents the target class, while the y-axis represents the output class. Here, we provide two data classes: attack and non-attack. This confusion matrix provides values for 11540 true negatives (TN), 680 false negatives (FN), 638 true positives (TP), and 2706 false positives (FP). 78.24% is the total accuracy of the Naïve Bayes classifier.

Table 2: Performance results of the Naïve Bayes classifier

Model	Accuracy	Sensitivity	Specificity	Kappa
Naïve Bayes	78.24	94.43	19.07	0.173

Table 2, provides the naive Bayes classifier's performance findings. From this table, the naïve Bayes classifier obtained the highest 78.24% of accuracy, 94.43% of sensitivity, 19.07% of specificity, and 0.173 Kappa, respectively.

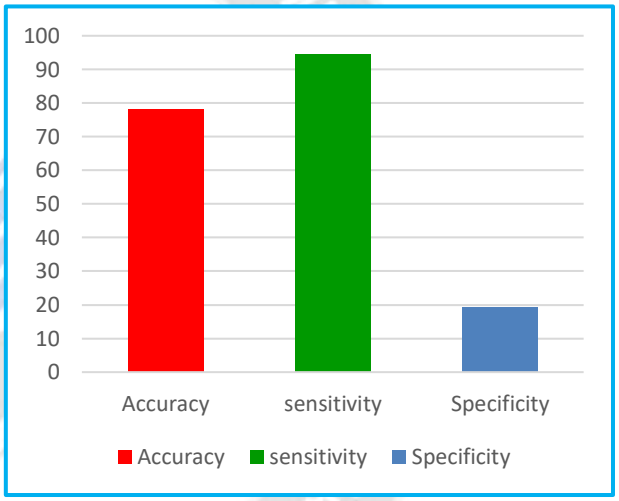

Figure 7: Bar plot of the performance results for Naïve Bayes classifier

Fig. 7 demonstrates a bar graph of the performance results for the Naïve Bayes classifier. In this graph, the values of three performance metrics are shown with different colors. Performance evaluation criteria are shown along the x-axis, while the proportion of values is shown along the y-axis. The red color bar is shown the accuracy parameter, which is 78.24%; the green color bar is shown the sensitivity parameter, which is 94.43%; and the blue color bar is shown the specificity parameter, which is 19.07%, respectively. It is clearly shown that the sensitivity has the highest value than the other parameter.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9s
Article Received: *30 April 2023* ***Revised:*** *30 June 2023* ***Accepted:*** *20 July 2023*

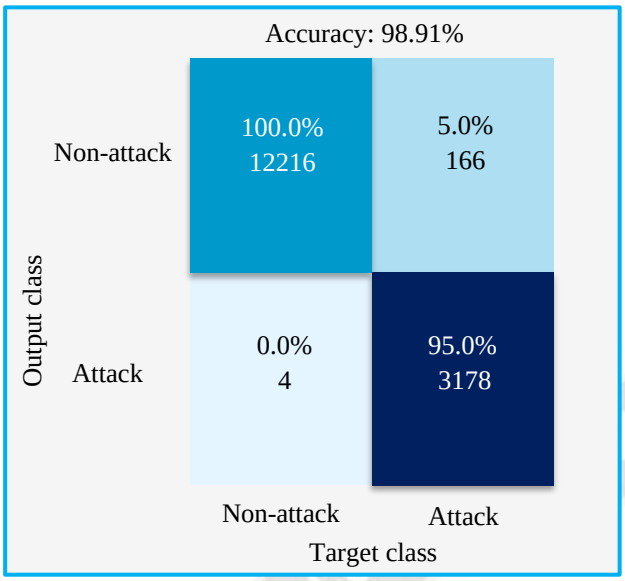

Figure 8: Confusion Matrix of the AdaBoost Classifier

Fig.8 shows a confusion matrix of the AdaBoost classifier, which shows the binary classification. The target class is indicated by the x-axis in this matrix, while the output class is represented by the y-axis. Here two data classes are given, which are attack and non-attack. From this confusion matrix, 12216 true negatives (TN), 166 false positives (FP), 3178 true positives (TP), and four false negatives (FN) values are given. The overall accuracy of the AdaBoost classifier is 98.91%.

Table 3: Performance results of the AdaBoost classifier

Model	Accuracy	Sensitivity	Specificity	Kappa
Adaptive Boost	98.90	99.96	95.03	0.967

The performance results of the AdaBoost classifier are given in Table 3. From this table, the AdaBoost classifier obtained the highest 98.90% of accuracy, 99.96% of sensitivity, 95.03% of specificity, and 0.967 Kappa, respectively.

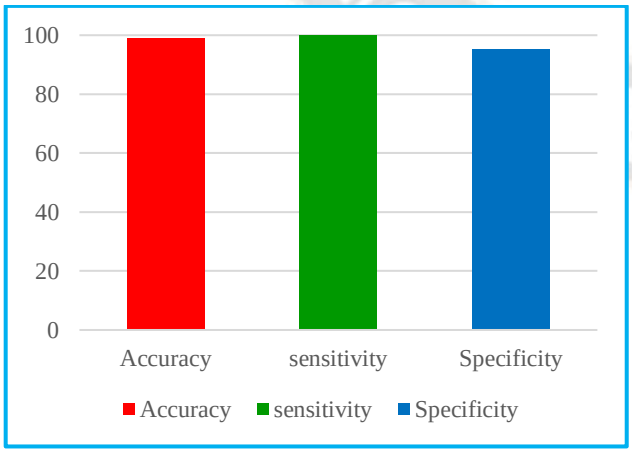

Figure 9: Bar plot of the performance results for AdaBoost classifier

Fig.9 demonstrates a bar graph of the performance results for AdaBoost classifier. In this graph, the values of three performance metrics are shown with different colours. The x-axis shows the performance evaluation parameters, and y-axis shows the number of values in percentage. The red colour bar shows the accuracy parameter which is 98.90%, the green colour bar is shown the sensitivity parameter which is 99.96%, and the blue colour bar is shown the specificity parameter which is 95.03%, respectively. It is clearly shown that sensitivity has a higher value than the other parameter.

6.2 Comparative analysis

This section compares the outcomes, and the comparison graphs and tables are shown below.

Table 4: Comparative Results of the Total Energy Consumption at Different Time Slots

Models	50	100	150	200	250	300
Proposed	8.8	17.6	26.4	35.2	44.0	52.9
OEERP	10	19	30	37	47	55
LEACH	20	42	65	90	110	135
DRINA	5	10	27	47	85	120
BCDCP	12	17	28	30	40	50

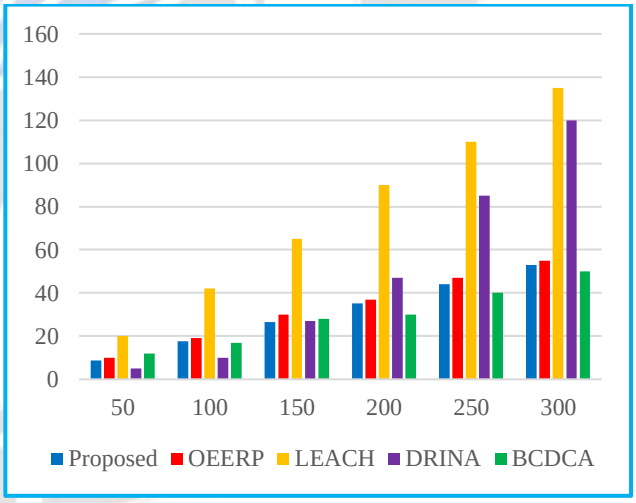

Figure 10: Total Energy Consumption at Different Time Slots

Fig.10 demonstrates the total energy consumption at different time slots of proposed and other existing routing protocols such as OEERP, LEACH, DRINA, and BCDCA. Time, in milliseconds, is plotted along the x-axis, and total energy consumption, in joules, along the y-axis. All models are represented by different colors. Our proposed model is represented by the blue color. At different time slots, the proposed approach consumed energy faster than the other methods.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9s
***Article Received:** 30 April 2023 **Revised:** 30 June 2023 **Accepted:** 20 July 2023*

Models	50	100	150	200	250	300
Proposed	76754.3	76086.9	75593.9	75431.0	75349.83	74388.9
OEERP	55000	55000	53000	52000	55000	54000
LEACH	57000	50000	52000	51000	55000	55000
DRINA	75000	74500	76000	70000	62000	56000
BCDCP	17000	26000	27000	28000	29000	30000

Table 5: Comparative Results of the Throughput at Different Time Slots

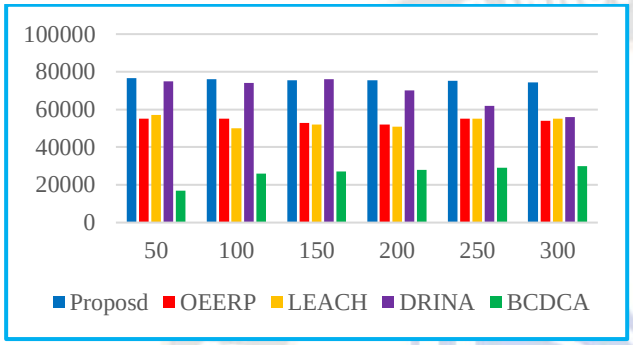

Figure 11: Throughput at different time slots

Table (5) and Fig.11 demonstrate the throughput at different time slots of proposed and other existing routing protocols such as OEERP, LEACH, DRINA, and BCDCA. In this figure, x-axis represents the time in milliseconds, and y-axis depicts the throughput value. The all models are represented by different colours. Our proposed model is represented by the blue colour. It is clearly shown that the proposed model has the highest throughput than the others. But obtained minimum time to consume the energy compared to the other protocols at different time slots. DRINA protocol is also performed well like the proposed protocol.

Table 6: Comparative results of the PDR at different time slots

Models	50	100	150	200	250	300
Proposed	100	99.285	99.285	99.285	99.285	99
OEERP	60	62	61	61	60	61
LEACH	63	63	63	63	62	63
DRINA	90	97	98	85	75	65
BCDCP	20	35	36	37	37	38

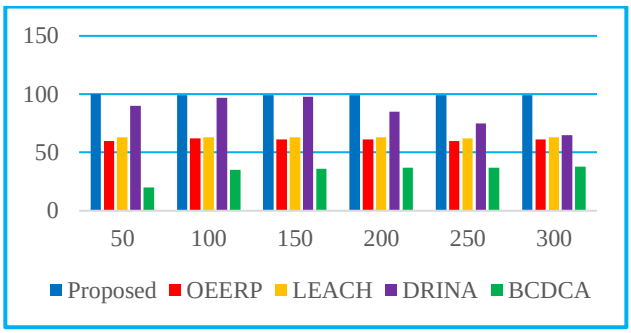

Figure 12: Packet Delivery Ratio at different time slots

Fig.12 demonstrates the packet delivery ratio (PDR) at different time slots of proposed and other existing routing protocols such as OEERP, LEACH, DRINA, and BCDCA. The x-axis in this graph depicts time in milliseconds, and the y-axis shows the PDR. The all models are represented by different colours. The proposed model is represented by the blue colour. According to this figure, the proposed protocol has the highest PDR value compared to the other protocols.

Table 7: Comparative results of the network lifetime at different time slots

Models	50	100	150	200	250	300
Proposed	10964.91	5482.45	3916.04	3132.83	2741.22	2284.3
OEERP	7000	3000	2000	1800	1200	1000
LEACH	2800	1500	1000	1000	800	500
DRINA	8500	6500	3300	1500	1000	600
BCDCP	4800	3000	2100	1800	1300	1000

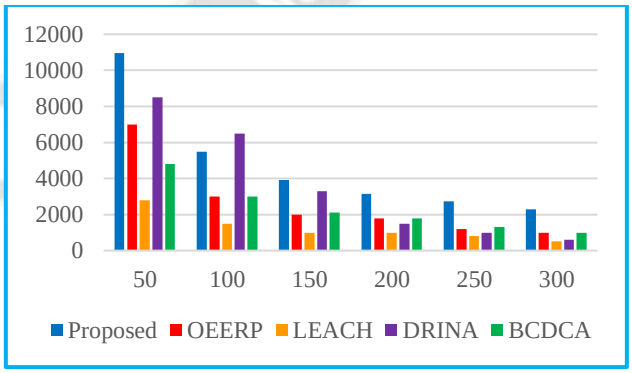

Figure 13: Overall network Lifetime at different time slots

At various time intervals, Fig. 13 shows the overall network lifetime of proposed and other current routing protocols such as OEERP, LEACH, DRINA, and BCDCA. In this graph, the x-axis shows time in milliseconds and the y-axis depicts network

lifetime in milliseconds. The all models are represented by different colours. The proposed protocol is represented by the blue colour. According to this figure, the proposed protocol has maximum network lifetime than the other protocols.

7. Conclusion and future work

This research concludes by discussing the critical need for accurate location in Wireless Sensor Networks (WSNs) based on SDN and proposed an approach that makes use of machine learning algorithms. The study acknowledges the dynamic nature of the environments in which WSNs operate and emphasizes the importance of adapting to changing conditions without requiring a substantial redesign. Using the Particle Swarm Optimization (PSO) technique for clustering, the Weighted Dijkstra algorithm for determining the shortest path, and machine learning algorithms like Naive Bayes and AdaBoost for data classification, the proposed methodology offers a comprehensive solution for localization in SDWSNs. A performance evaluation demonstrates that the proposed approach outperforms the state-of-the-art rivals. "throughput, energy consumption, packet delivery ratio, and network lifetime" are only few of the performance parameters that have been demonstrated to improve using the suggested strategy. Also, the study's machine learning algorithms performed well, with Naive Bayes attaining a 78.24% accuracy rate with a sensitivity and specificity of 94.43% and 19.07%, and AdaBoost achieving a remarkable 98.90% accuracy rate with a sensitivity and specificity of 99.96% and 95.03%, respectively.. These results demonstrate the efficiency of incorporating ML techniques into the localization procedure in SDN. In conclusion, our study aids the development of localization methods based on machine learning for SDWSNs. The proposed method provides useful insights and implementable solutions for optimizing resource utilization, prolonging network lifetime, and enhancing overall WSN performance by tackling the issues of accurate localization and using machine learning methods.

Further research might look at how well the proposed approach scales in large-scale SDWSN deployments. To further improve localization's accuracy and efficiency, researchers should investigate further machine learning methods and optimization strategies. Also, future study might benefit from taking energy efficiency into account by focusing on methods to reduce power use without sacrificing localization accuracy.

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Authors Contributions

Baraa Abbas Shahal and Mohamed Najm Abdullah suggested using machine learning-based localization in Software Defined Wireless Sensor Networks (SDWSNs), Both authors discussed the numerical simulation results contributed to the final work.

References

[1] S. I. Saheb, K. U. Khan, and C. S Bindu. "A Multi-Objective Sensor Deployment using BAT Algorithm to Improve Coverage, Lifetime and Energy Consumption in Wireless Sensor Networks." *International Journal of Intelligent Engineering & Systems* 15, no. 3 (2022).

[2] H. J. Abdullah, and M. N. Abdullah. "Routing enhancement in wireless sensor networks based on capsule networks: A survey." *International Journal of Nonlinear Analysis and Applications* 13, no. 2 (2022): 1229-1238.

[3] H. J. Abdullah and M.N. Abdullah. "ENHANCED SELECTING OF CLUSTER HEAD IN WIRELESS SENSOR NETWORK USING CAPSULE NETWORK."

[4] V. Bahl, A. Bhola, and V. Vyas. "Localization Based Distributed Selective Cluster Scheduling for Sustainable Energy Consumption and Efficient Data Transmission." *International Journal of Intelligent Engineering & Systems* 15, no. 4 (2022).

[5] F. Zijie, M. A. Al-Shareeda, M. A. Saare, S. Manickam, and S. Karuppayah. "Wireless sensor networks in the internet of things: review, techniques, challenges, and future directions." *Indonesian Journal of Electrical Engineering and Computer Science* 31, no. 2 (2023): 1190-1200.

[6] A. Melis, A. Al Sadi, D. Berardi, F. Callegati, and M. Prandini. "A Systematic Literature Review of Offensive and Defensive Security Solutions with Software Defined Network." *IEEE Access* (2023).

[7] F. F. Jurado-Lasso, L. Marchegiani, J. F. Jurado, A. M. Abu-Mahfouz, and X. Fafoutis. "A survey on machine learning software-defined wireless sensor networks (ml-SDWSNS): Current status and major challenges." *IEEE Access* 10 (2022): 23560-23592.

[8] O. P. Cloete, A. M. Abu-Mahfouz, and G. P. Hancke. "A review of wireless sensor network localisation based on software defined networking." In *2019 IEEE international conference on industrial technology (ICIT)*, pp. 1731-1736. IEEE, 2019.

[9] Y. Zhu, Y. Zhang, W. Xia, and L. Shen. "A software-defined network based node selection algorithm in WSN localization." In *2016 IEEE 83rd Vehicular Technology Conference (VTC Spring)*, pp. 1-5. IEEE, 2016.

[10] Y. Zhu, F. Yan, Y. Zhang, R. Zhang, and L. Shen. "SDN-based anchor scheduling scheme for localization in heterogeneous WSNs." *IEEE Communications Letters* 21, no. 5 (2017): 1127-1130.

[11] Y. Zhu, S. Xing, Y. Zhang, F. Yan, and L. Shen. "Localisation algorithm with node selection under power constraint in software-defined sensor networks." *IET Communications* 11, no. 13 (2017): 2035-2041.

[12] Y. Zhu, F. Yan, W. Xia, F. Shen, S. Xing, Y. Wu, and L. Shen. "Node Scheduling for Localization in Heterogeneous Software-Defined Wireless Sensor Networks." In *International Conference on Ad Hoc Networks*, pp. 154-164. Cham: Springer International Publishing, 2018.

[13] Y. Zhu, F. Yan, S. Zhao, S. Xing, and L. Shen. "On improving the cooperative localization performance for IoT WSNs." *Ad Hoc Networks* 118 (2021): 102504.

[14] O. P. Cloete, A. M. Abu-Mahfouz, and G. P. Hancke. "Comparison of localisation estimation algorithms in software defined wireless sensor networks." In *2019 IEEE 28th International Symposium on Industrial Electronics (ISIE)*, pp. 1556-1561. IEEE, 2019.

[15] D. Wang, D. Tan, and L. Liu. "Particle swarm optimization algorithm: an overview." *Soft computing* 22 (2018): 387-408.

[16] A. Halkai, and S. Terdal. "QALPA-An Efficient QoS Aware Hybrid Localization Model Using Particle Swarm Optimization and Ant Colony Optimization for Cognitive Wireless Sensor Networks." *International Journal of Intelligent Engineering & Systems* 14, no. 1 (2021).

[17] A. Pradhan, S. K. Bisoy, and A. Das. "A survey on PSO based meta-heuristic scheduling mechanism in cloud computing environment." *Journal of King Saud University-Computer and Information Sciences* 34, no. 8 (2022): 4888-4901.

[18] M. N. Abdullah, and K. E. Dagher. "Airborne Computer System Path-Tracking Based Multi-PID-PSO Controller Design." *International Journal of Intelligent Engineering & Systems* 14, no. 3 (2021).

[19] T. K. Hua, and N. Abdullah. "Weighted sum-Dijkstra's algorithm in best path identification based on multiple criteria." *J. Comput. Sci. Comput. Math* 8, no. 3 (2018): 2-8.

[20] Balasubramanian, S., Raparthi, M., Dodda, S. B., Maruthi, S., Kumar, N., & Dongari, S. (2023). AI-Enabled Mental Health Assessment and Intervention: Bridging Gaps in Access and Quality of Care. PowerTech Journal, 47(3), 81. https://doi.org/10.52783/pst.159

[21] B. Mahesh. "Machine learning algorithms-a review." *International Journal of Science and Research (IJSR).[Internet]* 9, no. 1 (2020): 381-386.

[22] W. Wang, and D. Sun. "The improved AdaBoost algorithms for imbalanced data classification." *Information Sciences* 563 (2021): 358-374.

[23] H. M. Fadhil, M. N. Abdullah, and M. I. Younis. "A framework for predicting airfare prices using machine learning." *Iraqi J. Comput. Commun. Control Syst. Eng* 22 (2022): 81-96.