International Journal of Innovative
Computing, Information and Control
Volume **21**, Number **6**, December **2025**

ICIC International ©2025 ISSN 1349-4198
pp. 1503–1522

DEEP Q-LEARNING-BASED PATH LOSS AND ENERGY EFFICIENT TOPOLOGY CONTROL FOR WIRELESS SENSOR NETWORKS

Ho Hai Quan[1,2], Le Huu Binh[1,*] and Nguyen Dinh Hoa Cuong[3]

[1]Faculty of Information Technology
University of Sciences
Hue University
77 Nguyen Hue, Hue 530000, Vietnam
hhquan.dhkh24@hueuni.edu.vn; *Corresponding author: lhbinh@hueuni.edu.vn

[2]Faculty of Information Technology
Ho Chi Minh City University of Industry and Trade
140 Le Trong Tan Street, Tay Thanh Ward, Tan Phu District, Ho Chi Minh City 72000, Vietnam
quanhh@huit.edu.vn

[3]Faculty of Business and Technology
Phu Xuan University
28 Nguyen Tri Phuong, Hue 530000, Vietnam
cuong.nguyen@pxu.edu.vn

Received March 2025; revised July 2025

Abstract. *WSNs are distributed systems that consist of numerous sensor nodes deployed in an environment to collect and transmit data. Although previous studies have focused on graph optimization algorithms and fundamental reinforcement learning methods, these approaches often face limitations in terms of stability and convergence toward predefined objectives. In this study, we propose DQPLET (Deep Q-learning-based Path Loss and Energy-efficient Topology Control), a novel approach that leverages Deep Q-Learning (DQL) in conjunction with the Levenberg-Marquardt (LM) algorithm. This method is employed to adjust the communication range of nodes, thereby optimizing the network topology and enhancing energy efficiency. The simulation results demonstrate that DQPLET outperforms existing methods in terms of the average node degree and transmission quality, making it a promising solution for energy-efficient topology control in WSNs.*
Keywords: Topology control algorithm, DQPLET, Next generation wireless network, Deep Q-learning, Wireless Sensor Network, Levenberg-Marquardt

1. **Introduction.** WSNs are distributed systems that consist of numerous sensor nodes deployed in an environment to collect and transmit data. These sensor nodes communicate wirelessly, forming a network that enables relaying data to a central processing unit. However, WSNs face several research challenges in terms of ensuring operational efficiency and energy conservation. Key issues include energy management for sensor nodes with limited power resources, optimization of communication range and coverage, traffic management between nodes, and maintenance of a stable network topology in scenarios where nodes are mobile or experience failures. Network Topology Optimization is the process of designing, adjusting, or restructuring the connections between nodes in a network to achieve optimal performance according to one or more specific criteria. In this study, topology optimization refers to adjusting the communication range such that the node degree approaches the desired value, while minimizing the Energy Efficiency

DOI: 10.24507/ijicic.21.06.1503

Ratio (EER) and path loss index to enhance network performance and longevity. The existing topology control techniques are generally categorized into coverage-based and connectivity-based approaches. These methods focus on objectives such as full coverage, barrier coverage, sweep coverage, energy management, and power control. These topics have garnered significant interest from the research community, particularly as WSNs have become a foundational technology for IoT applications. In this study, we propose a novel method called DQPLET, which integrates the DQL model with the LM algorithm and "Hello" mechanism. This approach is applied to the communication range adjustment problem in WSNs to achieve the desired node degree, thereby enhancing energy efficiency and optimizing the network topology. Specifically, DQPLET employs DQL to dynamically regulate the communication range of the network nodes, ensuring an optimal trade-off between connectivity and energy consumption. The neural network was trained using the LM algorithm, which optimizes and refines the Q-values for each action based on the observed states and rewards. The neural network is responsible for estimating the Q-value $Q(s, a)$ for each state-action pair, with the training process aimed at minimizing the error between the previous and updated Q-values, thereby improving the learning efficiency of the algorithm. The action selection policy follows an epsilon-greedy strategy, where an action is randomly selected with probability epsilon, whereas the action with the highest Q-value is chosen otherwise. The main contributions of this study are summarized as follows.

- We propose a novel topology control algorithm, namely DQPLET using DQL for WSNs.
- We simulated the implementation of some popular topology control algorithms and the proposed algorithm, and evaluated the performance metrics in the WSNs to demonstrate the effectiveness of the proposed algorithm.

The application of DQL in DQPLET has several advantages over traditional Q-learning algorithms because

- **Better generalization in large state spaces:** Traditional Q-learning uses a Q-table, which becomes infeasible when the number of states and actions is too large owing to the high computational and memory costs. Neural networks can approximate the Q-function even for many "unseen" states, enabling the learning of complex models that cannot be handled by Q-tables.
- **Memory efficiency:** There is no need to store Q-values for every state-action pair. Instead, the neural network only stores weights, which is significantly more memory-efficient.
- **Model-free learning from environment interaction:** This does not require prior knowledge of the state transition rules of the environment. The neural network learns directly from the agent's experience, making it suitable for real-world environments where models are often unknown or highly complex.
- **Flexible input handling:** Neural networks can process complex input data such as multi-dimensional vectors (e.g., position, node degree, and transmission range). This allows the algorithm to be applied to practical problems without constraints on input data formats.
- **Enhanced learning in dynamic or multi-agent environments:** With neural networks, agents can better adapt to changing environments or when interacting with multiple agents. This improves the feasibility of policy optimization in distributed or multi-dimensional systems.
- **Expandable with other deep learning techniques:** It is easy to integrate advanced methods, such as *Experience Replay*, which improves stability by learning

from stored experiences, or a *Target Network*, which separates the target network to reduce oscillations.

The remainder of this paper is organized as follows. Section 2 presents work related to the topology control problem in WSNs. In Section 3, the proposed algorithm is presented. Section 4 presents the simulation results. Finally, conclusions and suggestions for further development are presented in Section 5.

2. **Related Work.** In WSNs, network topology control methods are generally classified into two main categories: coverage optimization and connectivity enhancement. Techniques within each category focus on key aspects such as ensuring comprehensive coverage, handling coverage barriers, implementing efficient sweep coverage, and managing the energy consumption of sensor nodes. These issues have gained significant attention from the research community in recent years, as they play a critical role in improving the network performance and extending the lifespan of sensor nodes in dynamic environments.

In recent years, numerous studies have focused on network topology control to optimize resource utilization, enhance performance, and minimize energy consumption. The study in [2] proposed the Topological control by Flexibly Adjusting the Coverage Radius (TF-ACR) algorithm, which optimizes network performance in 5G mobile ad hoc networks by dynamically adjusting the coverage radius to maintain connectivity, balance the node load, and reduce energy consumption. Similarly, the study in [9] proposed the Energy-Efficient Hierarchical Topology Control (EEHTC) algorithm, which integrates clustering and game theory to optimize energy consumption in Software Defined Wireless Sensor Network (SDWSN), thereby significantly extenet work lifetime compared to existing approaches. Additionally, the research in [12] introduced a channel access architecture for large-scale ad hoc networks, designed to optimize channel resource allocation, enhance transmission efficiency, and reduce energy consumption. Singla and Munjal [17] provided a comprehensive review and evaluation of network topology control algorithms in WSNs, highlighting their role in minimizing energy consumption, extending the network lifespan, and optimizing key performance metrics such as latency and energy cost.

In the context of sensor energy management, a method for balancing energy consumption across sensor nodes in different network layers, while optimizing communication range control by considering both transmission energy and control overhead was introduced in [27]. The use of neural networks to construct data fusion models within sensor infrastructure was investigated in [28]. Battery Management Systems (BMSs) play a critical role in energy storage applications and are responsible for system monitoring, state estimation, and fault detection. To enhance the operational safety and reliability, a real-time sensor fault diagnosis technique based on model-driven analysis was proposed in [29].

Beyond resource optimization and energy efficiency, recent studies have focused on enhancing connectivity and fault tolerance in WSNs to ensure network stability, even in the presence of failures or system faults. The study in [10] proposed a heterogeneous sensor node deployment strategy to improve network resilience and lifespan, making the network more robust against failures and attacks. The study in [13] proposed an energy-efficient structural control mechanism for SDWSNs to support IoT applications. Meanwhile, the BPST algorithm, presented in [14], predicts the node mobility parameters and link availability duration, enabling the construction of a highly reliable and efficient network topology. Various optimization techniques have been applied to reducing latency and improving network throughput in the context of wireless networks and ad-hoc network performance optimization. The study in [6] employed the DDPG (Deep Deterministic Policy Gradient) reinforcement learning algorithm to optimize the network topology in a continuous action space, improving both energy efficiency and network throughput while introducing an accelerated learning mechanism for deploying new UAVs into the network. Similarly, [15] presented a UAV-assisted architecture for Connected and Autonomous Vehicles (CAVs), which enhances latency reduction and control optimization in VANETs, thereby improving the reliability and efficiency of Intelligent Transportation Systems (ITS).

Network topology control methods have also been extended to UAV ad hoc networks with the aim of maintaining stable network connectivity in dynamic environments. The study in [16] applied Deep Reinforcement Learning (DRL) for network topology management in UAV ad hoc networks, ensuring continuous connectivity and safe inter-UAV distances, thereby forming stable and efficient network structures. Similarly, [20] proposed a Connected Dominating Set (CDS)-based topology control mechanism for UAV ad hoc networks, which enhances network stability and minimizes frequent topology changes among flying nodes, achieving superior cost-effectiveness and reliability compared to other network control approaches. Genetic Algorithms (GAs) have been applied to solving network optimization problems in wireless network topology optimization. The study in [3] introduced SQETC, an energy-efficient topology control method for WSNs. This method leverages genetic algorithms to minimize the discrepancy the between actual and desired node degrees, thereby improving the data transmission quality and energy efficiency. Graph-based approaches have also been employed for network topology optimization, as demonstrated in [5], which presented a Local Tree-based Reliable Topology (LTRT) algorithm. This method constructs a network topology through four main phases: information exchange, topology formation, link removal, and transmission radius control. LTRT repeatedly utilizes the Minimum Spanning Tree (MST) algorithm to create a low-communication-range, energy-efficient, and stable network topology, thereby ensuring long-term network sustainability.

Topology control algorithms are increasingly being researched and developed in complex wireless networks to address critical challenges such as network connectivity, throughput, and energy efficiency. In practice, RL algorithms and network topology optimization techniques have demonstrated their effectiveness in solving these challenges and enhancing the performance of modern wireless networks. However, most of the existing topology control algorithms face difficulties in efficiently adapting to complex and highly dynamic network environments. Our research focuses on addressing key issues, including communication range optimization, node degree adjustment to achieve the desired degree, network stability maintenance, and energy consumption reduction. To achieve these objectives, we propose DQPLET, which is an efficient and energy-aware algorithm that self-adapts to large-scale and continuously evolving network environments.

3. DQPLET Algorithm.

3.1. Proposal ideas.

The proposed algorithm uses DQL to optimize the transmission range of nodes in WSNs, ensuring that the node degrees remain close to the desired degree. In this approach, the algorithm enhances network coverage, maintains stable connectivity between nodes, optimizes energy consumption, and improves network connectivity efficiency.

The program was designed to autonomously adjust the transmission range of nodes in a WSN, ensure bidirectional communication, achieve the desired number of connections, optimize network performance, and minimize energy consumption. The input parameters of the program include network configuration details, such as the number of nodes, workspace dimensions, initial transmission range, target node degree, number of training episodes, and reinforcement learning parameters (e.g., exploration probability, learning rate, discount factor, and transmission range adjustment rate). In addition, node positions are randomly initialized in a two-dimensional space to ensure a natural distribution. The output of the program consists of the final optimized transmission range for each node, established bidirectional communication links, and key network metrics such as the degree of each node and average network degree, which collectively reflect the system's connectivity and performance.

Program execution begins with environment initialization and node distribution. Key parameters such as the number of nodes, network area size, and initial transmission range were set, followed by random placement of nodes in the 2D space. Next, the program establishes connectivity between nodes by broadcasting "Hello" packets to all nodes within their transmission range. If a node receives a "Hello" message and has a sufficient transmission range to send a response, a bidirectional connection is established, and a neighbor list is generated based on successful message exchanges. The connectivity matrix was then updated, and the degree of each node was computed to assess the current connectivity levels. The program also checks the network connectivity using "Hello" message chains to ensure full interconnectivity.

The training phase is performed using reinforcement learning, in which each node collects information regarding its position, current connections, and transmission range, forming an input vector for the pre-trained neural network model. The model predicts Q-values for two possible actions: increasing or decreasing the transmission range. Using an ε-greedy strategy, nodes either explore by randomly selecting an action or exploit the learned policy by choosing the action with the highest Q-value. The node then adjusts its transmission range accordingly, and the updated range is temporarily applied to reassessing the bidirectional neighbor list.

The reward function evaluates the action based on how well the new node degree approximates the target degree, while also considering range adjustments. The calculated reward is then used to update the neural network model and improve future action selection. If a node's adjusted degree closely matches the target degree and the local network remains connected, a new transmission range is officially applied. The nodes then rebroadcast the "Hello" messages to confirm the bidirectional communication. This process was iteratively repeated over multiple training episodes, with each episode focusing on the adjustment of a specific node. As training progresses, the probability of random action selection (exploration) gradually decreases, allowing the model to transition from exploration to exploitation of the learned knowledge.

Through continuous learning and adaptive adjustments, the algorithm dynamically optimizes the transmission range of each node, ensuring network stability and achieving optimal performance in large-scale and highly dynamic WSNs environments.

Upon completing the training episodes, the program calculates the key network parameters, including the average node degree, average inter-node distance, and path loss value for each link. These metrics, along with detailed network data, were exported to Excel for further analysis. Finally, the program visualizes the trained network topology, where nodes are displayed along with their transmission ranges and the corresponding node degrees. The execution terminates upon reaching a predefined number of training episodes, ensuring that the network is optimized for connectivity maintenance and maximized transmission efficiency. These steps are executed continuously to optimize the configuration of the WSNs by combining DQL using the LM method with the "Hello" mechanism.

3.2. **DQL fundamentals.** DQL is an RL approach designed to solve control problems in environments with large or continuous states and action spaces. This method extends Q-learning, a classical algorithm based on the Bellman equation, by leveraging deep neural networks to approximate the Q-function (action-value function). This allows DQL to be applied to complex tasks such as game playing, robotic control, and automation.

In reinforcement learning, the environment is typically modeled as a Markov Decision Process (MDP) that consists of the following components: state space (S), action space (A), transition function $P(s'|s,a)$, reward function $R(s,a)$, and discount factor γ. The objective was to maximize the cumulative discounted rewards over time. Q-learning is based on the Bellman equation, where the optimal Q-function $Q^*(s,a)$ is defined as the desired current reward plus the maximum desired reward in the next state, and is formulated as follows:

$$Q^*(s,a) = R(s,a) + \gamma \max_{a'} Q^*(s',a') \tag{1}$$

In this algorithm, the Q-function is updated using the following equation:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right] \tag{2}$$

where α is the learning rate, s is the current state (representing the position and transmission range of the nodes), a is the action taken, r is the reward received after the action, s' is the new state, γ is the discount factor for future rewards, and $\max_{a'} Q(s',a')$ represents the highest predicted Q-value in state s'.

When the state space is extremely large, storing the Q-table becomes impractical. Therefore, DQL employs a deep neural network with parameter θ to approximate the Q-function, denoted by $Q(s,a;\theta) \approx Q^*(s,a)$. The objective is to determine the optimal parameter set θ such that the predicted values closely match the actual values based on the Bellman equation. To train the network, the loss function is defined as [30]

$$L(\theta) = E_{(s,a,r,s')} \sim D\left[(y - Q(s,a;\theta))^2\right] \tag{3}$$

with the target value:

$$y = r + \gamma \max_{a'} Q\left(s', a'; \theta^-\right) \tag{4}$$

where D is the experience replay dataset and θ^- represents the parameters of the target network, which is periodically updated from the main network. The training process uses gradient descent (or its variants such as Adam) with gradient [30]

$$\nabla_\theta L(\theta) = E_{(s,a,r,s')} \sim D\left[2(Q(s,a;\theta) - y) \cdot \nabla_\theta Q(s,a;\theta)\right] \tag{5}$$

and updates the parameters using the rule:

$$\theta \leftarrow \theta - \eta \nabla_\theta L(\theta) \tag{6}$$

where η denotes the learning rate.

3.3. **Modeling DQL for topology control.** To apply DQL to the transmission range adjustment problem in WSNs, the problem components were modeled as follows.

3.3.1. *Environment.* The environment represents the space in which the agent interacts. It is defined by states, and provides feedback through rewards. In the context of WSNs, the environment consists of the entire sensor network, including sensor nodes and their attributes (e.g., position, energy level, and transmission range) as well as communication conditions (e.g., interference, packet loss, and link quality). The environment provides feedback by updating the next state and reward after a node takes action. Specifically, in this problem, the environment is a WSNs space, represented as a 2D plane of size $H \times W$ [m^2]. The network consists of sensor nodes with fixed positions; however, their transmission ranges (R_i) are dynamically adjustable.

3.3.2. *State.* The state describes the current situation of the environment as observed by the agent. In the WSNs topology control problem, state represents the current status of a sensor node and includes information such as node positions, the current number of links for each node, and the average network degree. A state can be represented as a vector or encoded into discrete values. Each state describes the node positions, current transmission range of each node, and degree of network connectivity. For example, state S can be expressed as

$$S = \{(x_1, y_1, R_1, d_1), (x_2, y_2, R_2, d_2), \ldots, (x_N, y_N, R_N, d_N)\} \tag{7}$$

where (x_i, y_i) represent the fixed positions of node i, R_i is the current transmission range of node i, and d_i denotes the connectivity degree (number of links) of node i.

3.3.3. *Agent.* It is a decision-making entity that seeks to optimize actions to achieve its goals. In network topology control, each sensor node acts as an independent agent. The objective of the agent is to learn how to adjust its transmission range to optimize the network performance, including achieving the desired node degree, conserving energy, and maintaining connectivity. Sensor nodes operate independently and interact with the environment by performing various actions.

3.3.4. *Action.* Actions represent the operations that an agent can perform to modify the state of an environment. In this problem, actions involve adjusting the transmission range, specifically increasing the transmission power to expand the communication range, decreasing the transmission power to conserve energy, or maintaining the current transmission power to keep the existing range unchanged. Action space is typically defined in a discrete manner.

3.3.5. *Reward.* This is a feedback signal from the environment that evaluates the quality of the action taken. This is a numerical value received by the agent after performing an action, reflecting its effectiveness. In this problem, the reward is primarily designed based on the deviation between the average degree of the network and the desired degree k, as well as the maintenance of links within the maximum transmission range $R_{\max}$.

3.4. **Neural network and LM training method [26].** The neural network enables the model to learn how to adjust the transmission range of nodes in the network to achieve the desired degree, thereby optimizing the network topology while maintaining full graph connectivity. The neural network was trained using the LM method to optimize and update the Q-values for actions based on state and reward information. It is responsible for estimating $Q(s, a)$ for each state-action pair, with training aimed at minimizing the error between the previous and updated Q-values.

The neural network is initialized with n hidden layers and takes four input parameters of size numNodes $\times$ 4, including the node position (x, y) in 2D space, node degree, and transmission range. The network has two output nodes that correspond to two possible actions: increasing or decreasing transmission range.

Feedforward Network Configuration: The neural network is structured as a feedforward network with the following parameters: lr = 0.7: Learning rate; max_fail = 5: Maximum consecutive errors before stopping training; epochs = 5: Maximum number of training iterations; LM training method, which ensures precise optimization.

The LM method is an optimization technique that is primarily used for training artificial neural networks. It combines the strengths of gradient descent and Newton's method to enhance the convergence speed and accuracy.

Update Formula: The LM method optimizes the loss function by updating the parameters using the following formula [26,31]:

$$w \leftarrow w - \left[J^T J + \lambda I\right]^{-1} J^T e \tag{8}$$

where w represents the vector of the model (neural network weights), J is the Jacobian matrix of the loss function with respect to the parameters, e is the error vector between the model output and actual value, λ represents the damping factor for adjustment and I represents the identity matrix.

Exploration Decay Formula: During training, the exploration parameter ϵ is gradually reduced to minimize random actions, following the equation:

$$\epsilon = \epsilon \times \text{Dec} \tag{9}$$

Algorithm 1 presents the pseudocode of the DQPLET algorithm. It begins by initializing fundamental parameters such as the number of nodes, network space size, desired node degree, number of training episodes, and learning coefficients. The positions of the nodes are then randomly generated within a given space, along with their initial transmission ranges. A neural network with n hidden layers was configured to compute the Q-values during the training process, enabling adaptive topology optimization for energy-efficient and stable connectivity in WSNs. Each node in the wireless sensor network is considered an independent agent capable of learning and making decisions to adjust its own transmission range to optimize the overall network topology in terms of connectivity and energy efficiency (Step 1). Each node broadcasts a "Hello" packet to identify neighboring nodes within its transmission range, thereby collecting information such as coordinates, transmission range, and the number of neighboring nodes that can connect directly (i.e., the current node degree) (Step 2). The collected data are then processed and normalized into a feature vector, which is fed into a neural network designed with multiple hidden layers to learn the mapping between the current state and Q-value of each action (Step 3). The neural network predicts the Q-values corresponding to feasible actions that the node can take, typically by increasing or decreasing the transmission range, representing the expected future reward if the action was chosen (Step 4). Based on the predicted Q-values, a specific action is selected using the epsilon-greedy policy: with probability epsilon, a random action is chosen for exploration, and with probability 1-epsilon, the action with the highest Q-value is chosen to exploit the learned knowledge (Step 5). According

Algorithm 1 DQPLET Algorithm

Input:
- A WSN with its technical parameters;
- Desired node degree;

Output: A topology with average node degree approaching the desired degree;

Method:

1: **for** (*each node I in the network*) **do**
2: Collect information of coordinates, communication range and degree of neighbor nodes based on "Hello" packets;
3: Build a neural network with the input parameters collected from Step 2;
4: Predict the Q-values of actions in the set of feasible actions (A_i) that node I can choose;
5: Choose action $a \in A_i$ according to ε-greedy policy;
6: Adjust the communication range of node I based on action a selected in Step 5;
7: Broadcast "Hello" packets to neighboring nodes to update new node degree;
8: Update Q-value using Equation (2);
9: Retrain the neural network to predict the Q-values for the next actions;
10: **end for**

to the selected action, the node adjusts its transmission range: if the action is to increase, the range is extended to the allowed maximum; if the action is to decrease, the range is reduced but not below a predefined minimum threshold (Step 6). After adjustment, the node sends another "Hello" packet to update its neighbor list and recalculates its node degree based on the newly adjusted transmission range (Step 7). The Q-value is then updated based on the received reward, which reflects the deviation between the achieved node degree and desired node degree, contributing to improved learning policies in future iterations (Step 8). With the updated information, the neural network was retrained to enhance its ability to predict Q-values in subsequent iterations, thereby improving the accuracy and adaptability to changes in the network environment (Step 9). This process is repeated until all the nodes reach a stable state, where the average node degree approaches the target value (Step 10).

4. **Performance Evaluation.** In this section, the proposed DQPLET algorithm is evaluated and compared with existing methods using MATLAB R2024 on a Windows 10 64-bit operating system and computer configuration: Intel(R) Core(TM) i5-6300U CPU @ 2.40GHz 2.50GHz, 24GB RAM. The proposed algorithm was compared with the traditional RL-CRC [4] and LTRT [5] algorithms, as well as the baseline case without topology control. The RL-CRC [4] algorithm utilizes Q-learning to adjust the transmission range while adapting the node degrees to match the desired values. The LTRT [5] algorithm constructs multiple spanning trees to achieve low-degree network structures with reduced transmission ranges.

4.1. **Simulation environment.** In our simulations, nodes are randomly deployed in a 1000 m $\times$ 1000 m area, and the maximum transmission range was 250 m. Each node can dynamically adjust its transmission range within the range $[R_{\min}, R_{\max}]$, and a WSN is considered fully connected when the desired node degree is 4. Therefore, we set the desired node degrees to $k_{\max} = 3$, 4, 5 to simulate different node degree scenarios. The number of deployed nodes varied from 50 to 90, and each simulation scenario was repeated 10 times. The number of iterations for the DQPLET was set to 100 to achieve the desired node degree. The simulation parameters are presented in the following table.

Table 1. Simulation parameters

Parameter	**Value**
Number of Nodes (N)	$50 \sim 90$
Communication Range ($R_{\min}, R_{\max}$)	50 m, 250 m
Network Area	1000×1000 m^2
Desired Node Degree (D_{desired})	3, 4, 5
Discount Factor (γ)	0.9
Learning Constant (α)	0.001
Number of Episodes	100
Random Action Probability (ϵ)	0.1

4.2. **Simulation results.**

4.2.1. *Average node degree.* To achieve the desired node degree, analyzing the efficiency, stability, and variability of different algorithms is crucial for evaluating their convergence ability and practical applicability in network systems. Data collected from 50 to 90 nodes were aggregated using statistical indicators such as Q1, Q2 (median), Q3, IQR, mean, and standard deviation. The three algorithms under evaluation, DQPLET, RL-CRC, and

LTRT, were analyzed under the desired node degree levels ($k = 3$, 4, and 5), providing a comprehensive assessment of their ability to maintain the desired node degree.

The following section presents a quartile analysis for the case where the desired node degree is $k = 3$, 4, 5.

First, we analyze the node degree, with the results presented in Table 2, Figure 1, and Figure 2(a), showing the outcomes for $k = 3$ as follows: the DQPLET algorithm achieves an average node degree of approximately 3.33, with a narrow value range from 3.32 to 3.35. Specifically, the Q1, median, and Q3 values were around 3.33, 3.33, and 3.34, respectively, with an IQR of only 0.01 and a Std Dev of approximately 0.01. This indicates that the DQPLET maintains convergence and stability in achieving the desired node degree. In contrast, RL-CRC produces a significantly lower average node degree (2.44), fluctuating between 2.25 and 2.64, with an IQR of 0.26 and a Std Dev of approximately 0.14, reflecting high instability in maintaining the node degree. Meanwhile, LTRT results in an average node degree of approximately 5.79, which exceeds $k = 3$, with a fluctuation range of 5.56 to 5.90, an IQR of 0.17, and a Std Dev of around 0.11, indicating a tendency to generate excessive node degrees. These results demonstrate that the DQPLET outperforms the other algorithms in terms of convergence and stability at $k = 3$.

Next, the evaluation results are conducted at $k = 4$ corresponding to Table 2, Figure 1, and Figure 2(b). DQPLET continues to demonstrate its effectiveness, achieving an average node degree of approximately 4.34, with minimal fluctuation between 4.32 and 4.35. The Q1, median, and Q3 values are nearly identical at 4.33, 4.33, and 4.35, respectively, with

TABLE 2. Average node degree index by quartile with $k = 3$, 4, 5

Attribute	**$k = 3$**			**$k = 4$**			**$k = 5$**		
	DQPLET	RL-CRC	LTRT	DQPLET	RL-CRC	LTRT	DQPLET	RL-CRC	LTRT
Min	3.32	2.25	5.56	4.32	2.30	7.04	5.31	2.37	7.72
Max	3.35	2.64	5.90	4.35	2.83	7.91	5.35	2.68	9.73
Q1	3.33	2.31	5.70	4.33	2.36	7.34	5.32	2.50	8.45
Q2 (Med.)	3.33	2.40	5.83	4.33	2.47	7.78	5.33	2.56	9.14
Q3	3.34	2.57	5.87	4.35	2.54	7.86	5.34	2.63	9.67
IQR	0.01	0.26	0.17	0.02	0.18	0.52	0.02	0.13	1.22
Mean	3.33	2.44	5.79	4.34	2.48	7.63	5.33	2.56	9.04
Std Dev	0.01	0.14	0.11	0.01	0.14	0.30	0.01	0.09	0.68

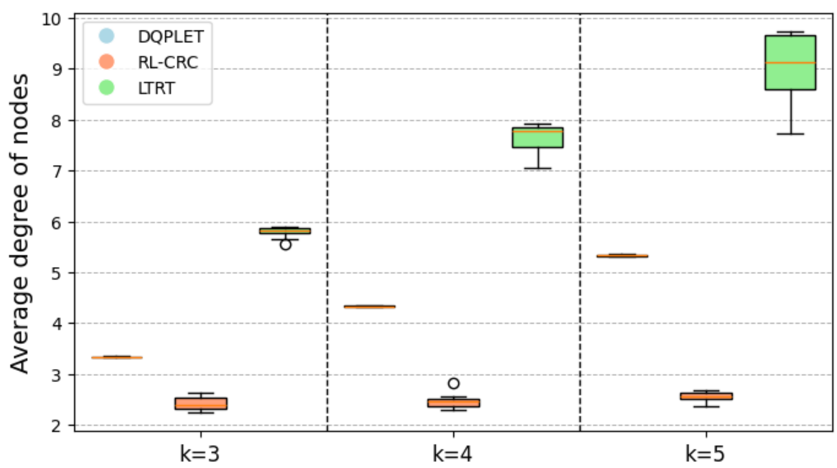

FIGURE 1. Average node degree

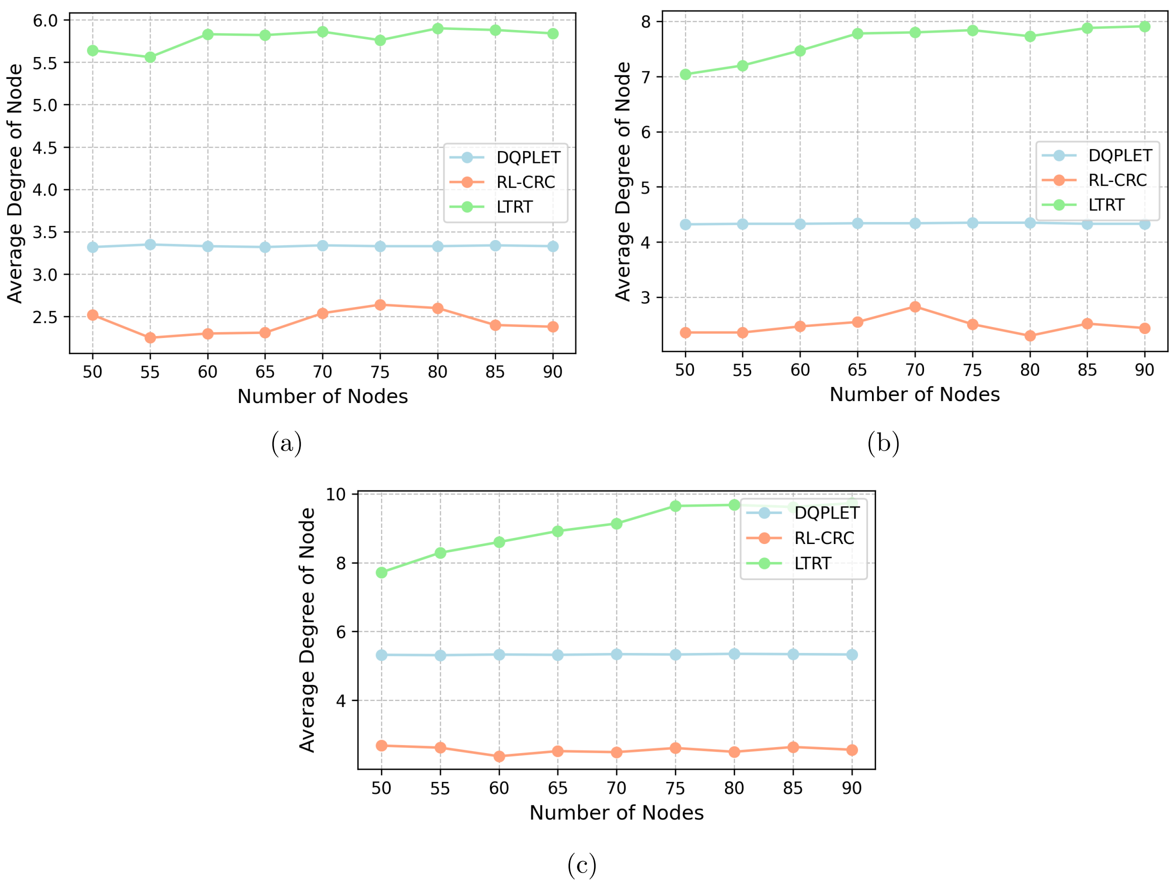

FIGURE 2. Average degree of node when $k = 3$, 4, 5

an IQR of only 0.02 and a standard deviation of approximately 0.01, indicating strong stability and convergence capability. In contrast, RL-CRC maintains a lower-than-desired average node degree Mean 2.48, fluctuating between 2.30 and 2.83, with an IQR of 0.18 and a Std Dev of 0.14, highlighting its inability to achieve the desired node degree. Meanwhile, LTRT results in an average node degree of approximately 7.63, ranging from 7.04 to 7.91, with an IQR of 0.52 and a Std Dev of 0.30, indicating that while it achieves high degree values, it suffers from stability issues in maintaining the desired node degree.

Finally, with $k = 5$ corresponding to Table 2, Figure 1, and Figure 2(c), DQPLET continues to demonstrate its ability to maintain the desired node degree, with an average value fluctuating between 5.31 and 5.35, and a mean of approximately 5.33. The Q1, median, and Q3 values are nearly identical at 5.32, 5.33, and 5.34, respectively, with an IQR of only 0.02 and a standard deviation of approximately 0.01, confirming its convergence performance. In contrast, RL-CRC achieved an average node degree of 2.56, fluctuating between 2.37 and 2.68, IQR of 0.13 and a Std Dev of 0.09. While there is a slight improvement in the stability, it still fails to meet the desired node degree. Meanwhile, LTRT results in a higher-than-desired average node degree of approximately 9.04, ranging from 7.72 to 9.73, with an IQR of 1.22 and a Std Dev of 0.68, indicating a tendency to create excessive node degrees.

In summary, across the desired node degree levels $k = 3$, 4, and 5, the DQPLET algorithm stands out with its ability to maintain node degrees close to the desired values, while exhibiting minimal fluctuations (IQR between 0.01 and 0.02, and Std Dev around 0.01). This demonstrates DQPLET's stability and convergence performance of DQPLET compared with other algorithms. By contrast, RL-CRC consistently produces lower-than-desired node degrees Mean between 2.44 and 2.56 with high variability, indicating its limitations in maintaining the target degree. However, while LTRT achieves high average node degrees Mean between 5.79 and 9.04, its tendency to exceed the desired degree and large fluctuations can lead to unnecessary overhead, reducing the stability of the network structure. Thus, considering both convergence and stability, DQPLET proved to be the optimal choice for applications that require stable network topology control.

4.2.2. *EER.* Next, we investigate the EER, an important metric for evaluating the network's energy consumption, with the results assessed based on Table 3 and Figure 3.

TABLE 3. EER index by quartile with $k = 3$, 4, 5

Attribute	$\boldsymbol{k = 3}$			$\boldsymbol{k = 4}$			$\boldsymbol{k = 5}$		
	DQPLET	RL-CRC	LTRT	DQPLET	RL-CRC	LTRT	DQPLET	RL-CRC	LTRT
Min	25.60	26.01	39.26	31.62	24.98	57.09	37.99	26.72	70.31
Max	42.94	42.23	69.17	50.05	43.55	79.46	63.06	43.02	88.47
Q1	30.19	28.06	45.07	39.25	27.24	61.59	46.31	28.61	70.39
Q2 (Med.)	32.27	32.72	55.20	39.37	36.05	65.60	47.53	32.22	79.24
Q3	32.56	35.89	59.20	39.56	36.70	73.78	49.86	37.49	84.32
IQR	2.37	7.83	14.13	0.31	9.46	12.19	3.55	8.88	13.93
Mean	32.79	33.18	52.71	41.19	33.90	69.34	48.99	33.10	78.70
Std Dev	5.75	4.88	10.84	5.80	6.05	7.89	8.36	5.61	6.85

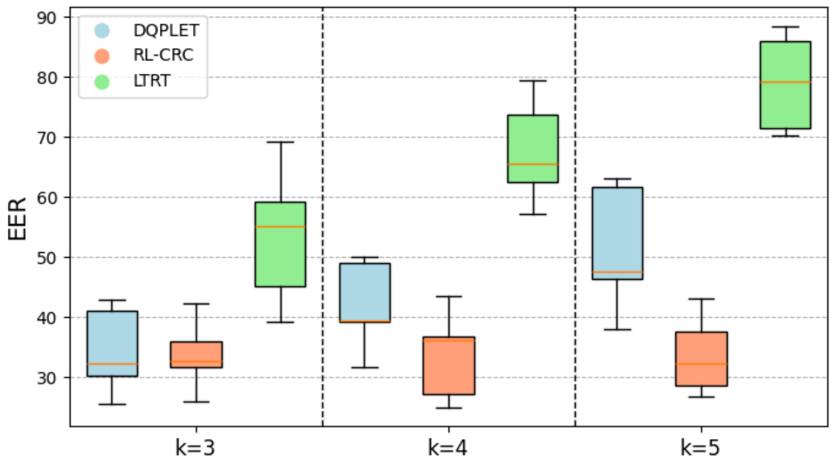

FIGURE 3. EER

In WSNs, EER measures the relative energy consumption while maintaining the desired node degree, providing insights into the efficiency of each algorithm. A quartile analysis of this metric, including Min, Q1, Q2 (median), Q3, Max, IQR, Mean, and Std Dev, was conducted to evaluate the stability and effectiveness of each approach. This analysis focuses on DQPLET, RL-CRC, and LTRT under the desired node degrees $k = 3$, 4, and 5, emphasizing the performance of DQPLET compared to the other two algorithms.

We first evaluate the EER metric with $k = 3$ corresponding to Table 3, Figure 3, and Figure 4(a). DQPLET achieves an EER ranging from 25.60 to 42.94, with a median of 32.27, IQR = 2.37, Mean = 32.79, and Std Dev = 5.75, indicating high stability with minimal fluctuation around the median value. In contrast, while RL-CRC has a similar median (32.72) and Mean (33.18) close to DQPLET, its IQR reaches 7.83, reflecting

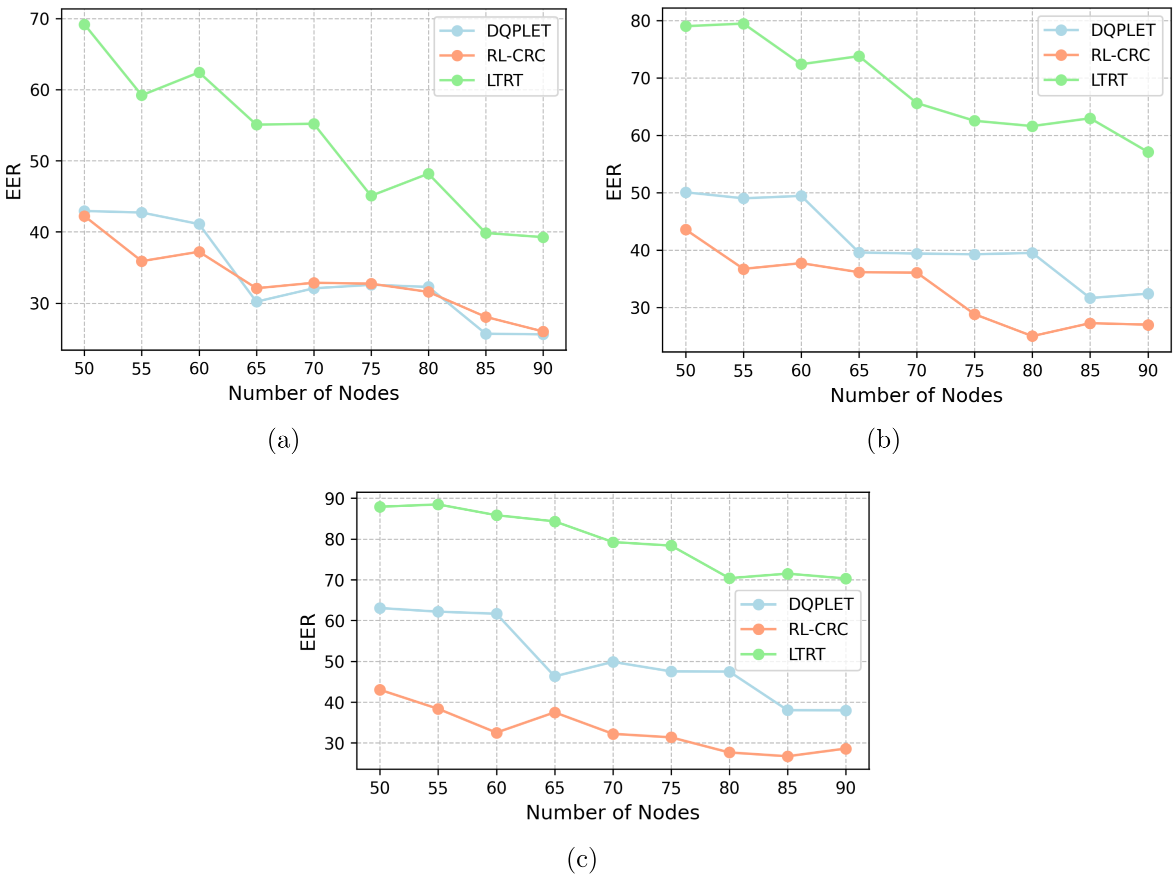

FIGURE 4. EER when $k = 3, 4, 5$

significant variability in energy efficiency. Meanwhile, LTRT exhibits the highest EER values, with a median of 55.20, Mean = 52.71, and IQR of 14.13, indicating considerable instability.

Then, with $k = 4$ corresponding to Table 3, Figure 3, and Figure 4(b), DQPLET achieves energy efficiency, with EER values ranging from 31.62 to 50.05, a median of 39.37, and a low IQR of only 0.31, demonstrating the tight convergence of the measured values. DQPLET's Mean is 41.19, with a Std Dev of 5.80, reflecting stability. In contrast, while RL-CRC results in lower energy consumption (Mean = 33.90), its IQR of 9.46 indicates high variability and reduced overall reliability. LTRT continues to show high energy consumption (Mean = 69.34), although its IQR decreases to 12.19 compared to $k = 3$, it still falls short in terms of stability compared to DQPLET.

The EER metric will vary with $k = 5$ corresponding to Table 3, Figure 3, and Figure 4(c), DQPLET achieves an EER ranging from 37.99 to 63.06, with a median of 47.53, IQR = 3.55, Mean = 48.99, and Std Dev = 8.36, maintaining good stability across the dataset. Meanwhile, RL-CRC continued to show the lowest energy consumption (Mean = 33.10) but had a high IQR of 8.88, indicating significant data fluctuations. On the other hand, LTRT, despite achieving Mean = 78.70 and an IQR of 13.93, which indicates improved stability compared to $k = 4$, still exhibits high energy consumption.

In summary, $k = 3$, 4, and 5, DQPLET stands out because of its stability, with minimal fluctuations (low IQR) and strong convergence around the median value, resulting in consistent and reliable EER measurements. While RL-CRC consistently showed the lowest energy consumption, its high data variability reduced the consistency of the results.

Meanwhile, LTRT, despite improving the stability at higher desired node degrees, suffers from excessive energy consumption, which may not accurately reflect the true energy efficiency of the system. Therefore, considering convergence and stability in EER, DQPLET is evaluated as the optimal method, providing a balance between energy efficiency and stability and establishing a solid foundation for WSNs applications.

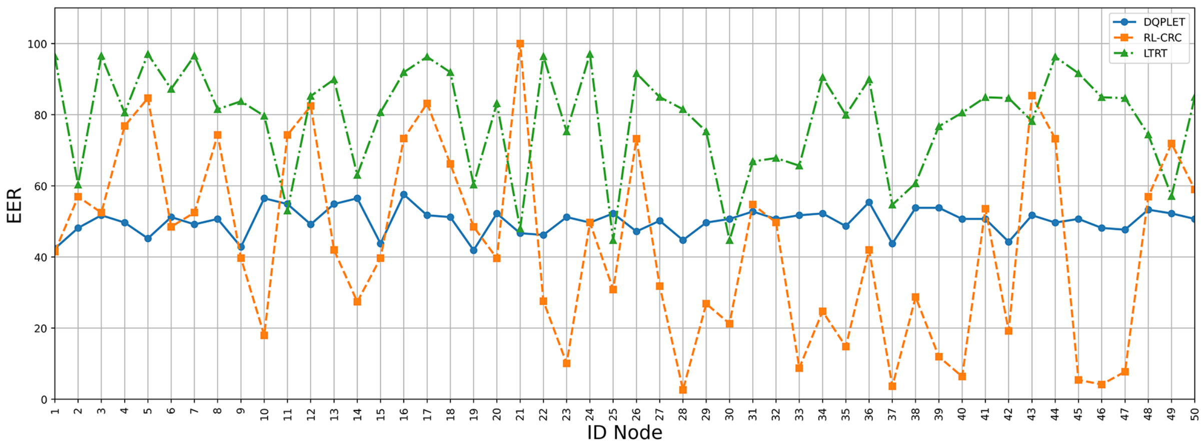

FIGURE 5. Energy consumption per node in the network of the three algorithms

4.2.3. *Energy consumption evaluation per node in the network for three algorithms (at $k = 4$, $N = 50$).* Based on the energy consumption data (EER) at 50 sensor nodes and an desired degree of 4 for the three algorithms DQPLET, RL-CRC, and LTRT, the following observations can be made: The DQPLET algorithm demonstrates an average energy consumption of approximately 50.05 units with a very low standard deviation of only 3.76, indicating high stability and consistency in energy usage among the sensor nodes. This reflects its effective communication region control, enabling balanced energy distribution, minimizing discrepancies between nodes, and avoiding situations in which some nodes consume significantly more or less energy than others. In contrast, although the RL-CRC algorithm has the lowest average EER, around 43.55 units, it had a very high standard deviation of 26.47. This highlights the severe instability in energy consumption across the sensor nodes, where some nodes operate with high efficiency while others consume excessive energy, leading to imbalance and adversely affecting the overall network lifespan. Meanwhile, the LTRT algorithm exhibited the highest average energy consumption, reaching nearly 79 units, with a standard deviation of 14.79. This indicates that it is an energy-intensive algorithm with a considerable level of fluctuation that reflects suboptimal energy control capabilities. In conclusion, when comparing all three algorithms, DQPLET stands out because of its balance between reasonable energy efficiency and high stability across a sensor network. RL-CRC, while energy-saving, lacks stability, whereas LTRT is both energy-intensive and inconsistent, making it less effective for optimal communication region control.

4.2.4. *Path loss ratio.* The path loss metric quantifies signal attenuation during transmission, with statistical indicators such as Min, Q1, Q2, Q3, Max, IQR, Mean, and Std Dev providing insights into the range, central tendency, and data dispersion. A stable measurement system typically produces closely clustered path loss values, whereas a high IQR and Std Dev indicate significant variability, reflecting inconsistent results.

First, the path loss metric is analyzed at $k = 3$ corresponding to Table 4, Figure 6, and Figure 7(a), DQPLET records path loss values ranging from 79.48 to 81.66, with a median of 80.45, IQR = 1.62, and Std Dev = 0.82, indicating moderate signal loss with relatively high stability. Compared to RL-CRC, which exhibits a lower median path loss (78.29) but greater variability (IQR = 2.08, Std Dev = 1.24), DQPLET demonstrates superior consistency in measurements. Furthermore, while LTRT shows high stability with IQR = 0.57 and Std Dev = 0.31, it results in higher path loss values (median = 83.38). This excessive path loss may distort actual signal attenuation estimates, reducing reliability.

TABLE 4. Path loss index by quartile with $k = 3, 4, 5$

Attribute	$k = 3$			$k = 4$			$k = 5$		
	DQPLET	RL-CRC	LTRT	DQPLET	RL-CRC	LTRT	DQPLET	RL-CRC	LTRT
Min	79.48	76.84	82.91	80.12	77.03	82.65	80.65	77.47	82.64
Q1	79.97	77.29	83.09	80.60	77.29	83.05	81.21	77.61	83.18
Q2 (Med.)	80.45	78.29	83.38	81.15	78.76	83.44	81.69	78.40	83.46
Q3	81.59	79.37	83.66	82.17	79.70	83.57	82.82	79.60	83.51
Max	81.66	80.61	83.84	82.25	80.61	83.63	82.86	80.85	83.61
IQR	1.62	2.08	0.57	1.57	2.41	0.52	1.61	1.99	0.33
Mean	80.61	78.43	83.37	81.25	78.66	83.31	81.85	78.69	83.32
Std Dev	0.82	1.24	0.31	0.81	1.28	0.33	0.84	1.18	0.30

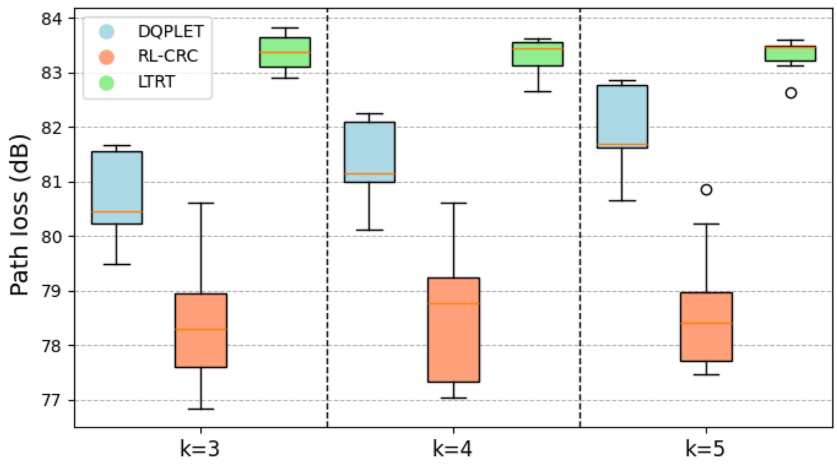

FIGURE 6. Path loss (dB)

Through the next analysis step with $k = 4$ corresponding to Table 4, Figure 6, and Figure 7(b), DQPLET achieves path loss values ranging from 80.12 to 82.25, with a median of 81.15, IQR = 1.57, and Std Dev = 0.81, demonstrating high stability and reliability in measurements. In contrast, RL-CRC produced a lower median (78.76), but exhibited greater variability (IQR = 2.41, Std Dev = 1.28), indicating reduced stability in path loss estimation. While LTRT maintains stability with IQR = 0.52 and Std Dev = 0.33, it results in a higher path loss value (median = 83.44). This overestimation could distort the accuracy of the signal attenuation measurements.

The final analysis step for path loss is conducted with $k = 5$ corresponding to Table 4, Figure 6, and Figure 7(c), DQPLET records path loss values ranging from 80.65 to 82.86, with a median of 81.69, IQR = 1.61, and Std Dev = 0.84, demonstrating stable measurement performance with moderate signal loss. In contrast, RL-CRC exhibits higher variability, with a median of 78.40, IQR = 1.99, and Std Dev = 1.18, leading to less reliable measurement results. While LTRT continues to show stability with IQR = 0.33 and Std

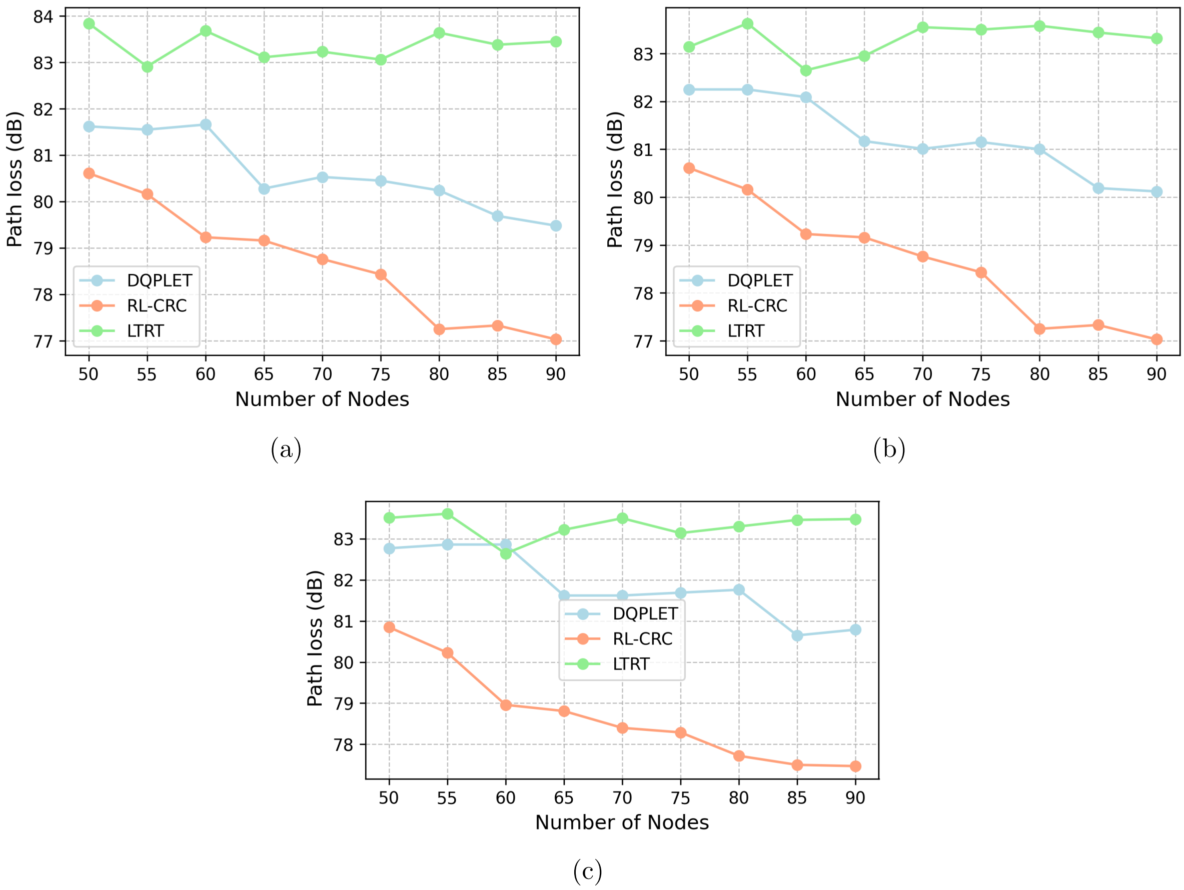

FIGURE 7. Path loss when $k = 3, 4, 5$

Dev = 0.30, its high path loss values (median = 83.46) suggest potential overestimation, which may not accurately reflect transmission conditions.

In summary, based on the analysis of the three datasets for $k = 3$, 4, and 5, DQPLET consistently demonstrated a balance between realistic path loss values and acceptable stability. While RL-CRC tends to produce lower values with high variability and LTRT achieves high stability but overestimates path loss, DQPLET stands out for delivering moderate, accurate, and reliable measurements. This makes the DQPLET the choice for applications requiring precise and stable path loss estimation, ensuring a realistic evaluation of system performance.

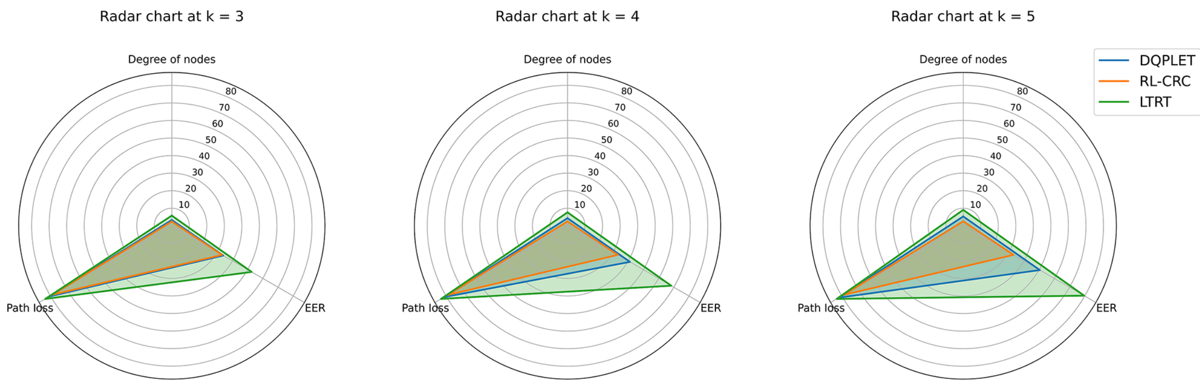

FIGURE 8. Energy consumption per node in the network of the three algorithms

4.2.5. *Evaluation of the relationship between network performance parameters.* For node degree metrics, RL-CRC consistently yields significantly lower values than the desired degree k (e.g., only around 2.5 when $k = 4$), resulting in a sparse network that reduces both EER and path loss but fails to ensure connectivity. Conversely, LTRT exhibits substantial overshooting, with degrees around 7.8 (for $k = 4$) or 9.7 (for $k = 5$), leading to very high EER and path loss due to extended transmission ranges and increased distances. DQPLET, on the other hand, maintains node degrees close to $k \pm 0.3$ (e.g., $k = 3$, 4, 5), ensuring sufficient connectivity neither under nor over-provisioned while keeping EER and path loss at an optimal moderate level. Overall, there is a direct relationship between degree, EER, and path loss: and as the degree increases, the algorithm must extend the average transmission range, causing both EER and path loss to rise. When the degree decreases, the EER and path loss decrease; however, the network becomes loosely connected. DQPLET successfully identifies a balanced point where the degree is just sufficient, enabling both EER and path loss to reach a harmonious trade-off between energy efficiency and communication quality. DQPLET demonstrates superiority in three aspects: first, high degree accuracy with minimal deviation from the desired k; second, strong energy efficiency with EER at only 33%-49% of $R_{\max}$, significantly lower than LTRT while still maintaining connectivity; third, path loss in the range of 80-82, lower than LTRT and acceptable compared to RL-CRC. In summary, DQPLET not only accurately achieves the required network topology in terms of node degree, but also optimizes both EER and path loss, offering a balanced solution that outperforms RL-CRC and LTRT in both energy saving and communication quality.

5. **Conclusion.**

Based on the above analysis, the DQPLET algorithm demonstrates several outstanding advantages over RL-CRC and LTRT.

- In experiments with k values of 3, 4, and 5, the algorithm consistently maintained the average node degree closest to the desired levels. In contrast, RL-CRC often results in degrees lower than required, whereas LTRT significantly exceeds them. This indicates the DQPLET's effective balance between the structural requirements and communication efficiency.
- In terms of energy consumption, DQPLET achieves a significantly lower EER than the other two approaches, proving its capability to preserve network connectivity in an energy-efficient manner. Meanwhile, LTRT tends to employ broader transmission ranges, which increases the EER and reduces the overall energy efficiency.
- Regarding path loss, DQPLET frequently yields lower or comparable values to RL-CRC and clearly outperforms LTRT in most scenarios. This highlights its ability to maintain a strong signal quality without necessarily extending the communication range, thus minimizing interference and conserving energy.
- Moreover, the algorithm maintained stable performance across various input datasets, as evidenced by low average metrics and minimal variance – demonstrating strong generalization capabilities and adaptability to diverse network configurations.
- Finally, DQPLET offers flexible control over transmission ranges, enabling it to satisfy the target node degrees while maintaining global connectivity and optimizing energy consumption across the network.

This paper proposes a new method called DQPLET, which combines DQL using the LM method and the "Hello" mechanism in a WSN environment. The DQPLET successfully met the defined objectives regarding the average node degree, path loss ratio, and EER. Compared to RL-CRC and LTRT, DQPLET demonstrates superior performance across all evaluation metrics. The proposed DQPLET framework establishes a foundation for further development and scalability in complex and demanding network environments.

Specifically, in future research, we aim to enhance the algorithm for application in 3D network models by incorporating continuous action spaces and highly dynamic conditions.

REFERENCES

[1] L. H. Binh and T. V. T. Duong, A novel and effective method for solving the router nodes placement in wireless mesh networks using reinforcement learning, *PLoS One*, vol.19, no.4, e0301073, 2024.
[2] L. H. Binh, T.-V. T. Duong and V. M. Ngo, TFACR: A novel topology control algorithm for improving 5G-based MANET performance by flexibly adjusting the coverage radius, *IEEE Access*, vol.11, pp.105734-105748, 2023.
[3] H. T. C. Hai, L. H. Binh and L. D. Huy, A topology control algorithm taking into account energy and quality of transmission for software-defined WSNs, *International Journal of Computer Networks & Communications (IJCNC)*, vol.16, no.2, pp.107-116, 2024.
[4] T. T. T. Le and S. Moh, An energy-efcient topology control algorithm based on reinforcement learning for WSNs, *International Journal of Control and Automation*, vol.10, pp.233-244, DOI: 10. 14257/ijca.2017.10.5.22, 2017.
[5] K. Miyao, H. Nakayama, N. Ansari and N. Kato, LTRT: An efficient and reliable topology control algorithm for ad-hoc networks, *IEEE Transactions on Wireless Communications*, vol.8, no.12, pp.6050-6058, 2009.
[6] T. Yoo, S. Lee, K. Yoo and H. Kim, Reinforcement learning based topology control for UAV networks, *Sensors*, vol.23, no.2, 921, 2023.
[7] X. Meng, H. Inaltekin and B. S. Krongold, Deep reinforcement learning-based topology optimization for self-organized WSNs, *2019 IEEE Global Communications Conference (GLOBECOM)*, pp.1-6, 2019.
[8] A. Waqas and H. Mahmood, A game theoretical approach for topology control in wireless ad hoc networks with selfish nodes, *Wireless Personal Communications*, vol.96, pp.249-263, 2017.
[9] Z. Geng, W. Xia, W. Cao, T. Wu, F. Yan, L. Shen and J. Pang, An energy-efficient hierarchical topology control algorithm in software-defined wireless sensor network, *Proc. of the 2021 13th International Conference on Wireless Communications and Signal Processing (WCSP)*, pp.1-6, 2021.
[10] X. Zhang, Y. Wang, D. Li, W. Chen and X. Ding, Robust t-path topology control algorithm in wireless ad hoc networks, *Algorithmic Aspects in Information and Management*, pp.229-239, 2021.
[11] M. Kadivar, An adaptive yao-based topology control algorithm for wireless ad-hoc networks, *2020 10th International Conference on Computer and Knowledge Engineering (ICCKE)*, pp.457-462, 2020.
[12] D. Wei, E. Yang and Y. He, Topology control and optimization of self-organized networks for wireless personal communication, *2023 IEEE 6th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE)*, pp.462-467, 2023.
[13] Z. Ding, L. Shen, H. Chen, F. Yan and N. Ansari, Energy-efficient topology control mechanism for IoT-oriented software-defined WSNs, *IEEE Internet of Things Journal*, vol.10, pp.13138-13154, 2023.
[14] P. Rahmani and H. H. Javadi, Topology control in MANETs using the Bayesian pursuit algorithm, *Wireless Personal Communications*, vol.106, pp.1089-1116, 2019.
[15] Y. Xue et al., UAV-assisted control design with stochastic communication delays, *2022 IEEE 8th International Conference on Computer and Communications (ICCC)*, pp.707-710, 2022.
[16] J. Li, P. Yi, T. Duan, Y. Wang, Z. Zhang, J. Yu and T. Hu, Centroid-guided target-driven topology control method for UAV ad-hoc networks based on tiny deep reinforcement learning algorithm, *IEEE Internet of Things Journal*, vol.11, pp.21083-21091, 2024.
[17] P. Singla and A. Munjal, Topology control algorithms for WSNs: A review, *Wireless Personal Communications*, vol.113, pp.2363-2385, 2020.
[18] I. Banerjee et al., Self-organizing topology for energy-efficient ad-hoc communication networks of mobile devices, *Complex Adaptive Systems Modeling*, vol.8, pp.1-21, 2020.
[19] M. Song et al., Locally adaptive power control for optimizing age of information in wireless networks, *2022 IEEE Wireless Communications and Networking Conference (WCNC)*, pp.1599-1604, 2022.
[20] X. Qi et al., CDS-based topology control in FANETs via power and position optimization, *IEEE Wireless Communications Letters*, vol.9, pp.2015-2019, 2020.
[21] D. Patra, S. Chavhan, D. Gupta, A. Khanna and J. J. P. C. Rodrigues, V2X communication based dynamic topology control in VANETs, *Adjunct Proceedings of the 2021 International Conference on Distributed Computing and Networking (ICDCN'21)*, New York, NY, USA, pp.62-68, 2021.

[22] E. Testi and A. Giorgetti, Blind wireless network topology inference, *IEEE Transactions on Communications*, vol.69, pp.1109-1120, 2021.
[23] Z. Wang, Q. Huang and J. Yu, Neural network-based direct robust adaptive non-fragile fault-tolerant control of amorphous flattened air-ground wireless self-assembly system, *Robotic Intelligence and Automation*, pp.537-550, 2023.
[24] J. Chen and Z. Wang, Coordination game theory-based adaptive topology control for hybrid VLC/RF VANET, *IEEE Transactions on Communications*, vol.69, pp.5312-5324, 2021.
[25] E. A. Feukeu and M. Sumbwanyambe, Using neural network and Levenberg-Marquardt algorithm for link adaptation strategy in vehicular ad hoc network, *IEEE Access*, vol.11, pp.93331-93340, 2023.
[26] P. Kumar, F. Almeida, B. Nagaraja, A. R. Ajaykumar and Q. M. Al-Mdallal, Neural network model using Levenberg-Marquardt backpropagation algorithm for the Prandtl fluid flow over stratified curved sheet, *IEEE Access*, vol.12, pp.102242-102260, 2024.
[27] J. Wang, S. Lu and C. Li, Uneven clustering routing protocols for multi-hop cognitive radio sensor networks: General design principles and an illustrative example, *International Journal of Innovative Computing, Information and Control*, vol.21, no.1, pp.153-172, 2025.
[28] B. Xiao, X. Xie and C. Yang, Multi-sensor data fusion based on GCN-LSTM, *International Journal of Innovative Computing, Information and Control*, vol.18, no.5, pp.1363-1381, 2022.
[29] D. Xu, Y. Ma, W. Yang, T. Pan and Z. Dou, Sliding mode observer-based sensor fault diagnosis for lithium-ion battery packs, *International Journal of Innovative Computing, Information and Control*, vol.19, no.5, pp.1455-1470, 2023.
[30] M. A. Ouamri, R. Alkanhel, D. Singh, E. M. El-Kenawy and S. S. Ghoneim, Double deep Q-network method for energy efficiency and throughput in a UAV-assisted terrestrial network, *Computer Systems Science and Engineering*, vol.46, pp.73-92, 2023.
[31] M. T. Hagan and M. B. Menhaj, Training feedforward networks with the Marquardt algorithm, *IEEE Transactions on Neural Networks*, vol.5, no.6, pp.989-993, 1994.

Author Biography

Ho Hai Quan graduated with a Bachelor's degree in Information Technology in 2008 and a Master's degree in Computer Science in 2013 from the Hue University of Sciences, Vietnam, respectively. He is currently a lecturer at the Faculty of Information Technology, Ho Chi Minh City University of Industry and Trade, Vietnam. He is currently a Ph.D. student at the University of Sciences, Hue University, Vietnam.

His current research interests include machine learning, deep learning, AI, next-generation network technology, applications of machine learning and artificial intelligence in network technology, network simulation and design.

Le Huu Binh received the B.E. degree in Telecommunications and Electronics from the Da Nang University of Science and Technology, Vietnam, in 2001, the M.Sc. degree in Computer Sciences from the Hue University of Sciences (HUSC), Vietnam, in 2007, and the Ph.D. degree in Informatics from the Vietnam Academy of Science and Technology, Vietnam, in 2020. He was a Senior Engineer of transmission and switching exchange with the Hue Telecommunications Center, Vietnam Posts and Telecommunications Group (VNPT), Thua Thien Hue, Vietnam, from 2001 to 2009. From 2010 to 2021, he was with the Hue Industrial College (HUE-IC), Vietnam, where he was the Dean of the Faculty of Information Technology and Telecommunications. Since 2022, he has been with the Faculty of Information Technology, University of Sciences, Hue University, Vietnam, where he is currently a lecturer. His current research interests include next generation wireless network technologies, software defined networking, application of machine learning, and artificial intelligence in network technology.

Nguyen Dinh Hoa Cuong received his Doctorate in Computer Science from Khon Kaen University, Thailand, in 2016. Currently, he serves as the Dean of the Faculty of Business and Technology at Phu Xuan University, Vietnam. His primary research interests encompass deep learning, text mining, knowledge management systems, and recommender systems.