Chapter 6

Node Fault Prediction Assisted Small-World IoT Networks Using ML Frameworks: Towards Performance Improvement

This chapter extends the proposed node fault prediction and routing framework by incorporating Small-World Characteristics (SWC) to establish long-range links in IoT networks, thereby improving the average clustering coefficient (ACC) and average path length (APL) for enhanced connectivity, reduced latency, and greater scalability. As IoT networks become more complex and large-scale, efficient data routing and node-fault prediction techniques are crucial for ensuring reliable and robust communication. The reliability of individual sensor nodes remains a critical factor in maintaining the overall performance of these networks. The authors in [51] proposed a novel energy-efficient fault diagnosis technique for WSNs, where each sensor node independently detects its faults using its sensed data, reducing communication overhead and delay. Moreover, authors in [136] proposed a Feed-Forward Neural Network model with a hybrid meta-

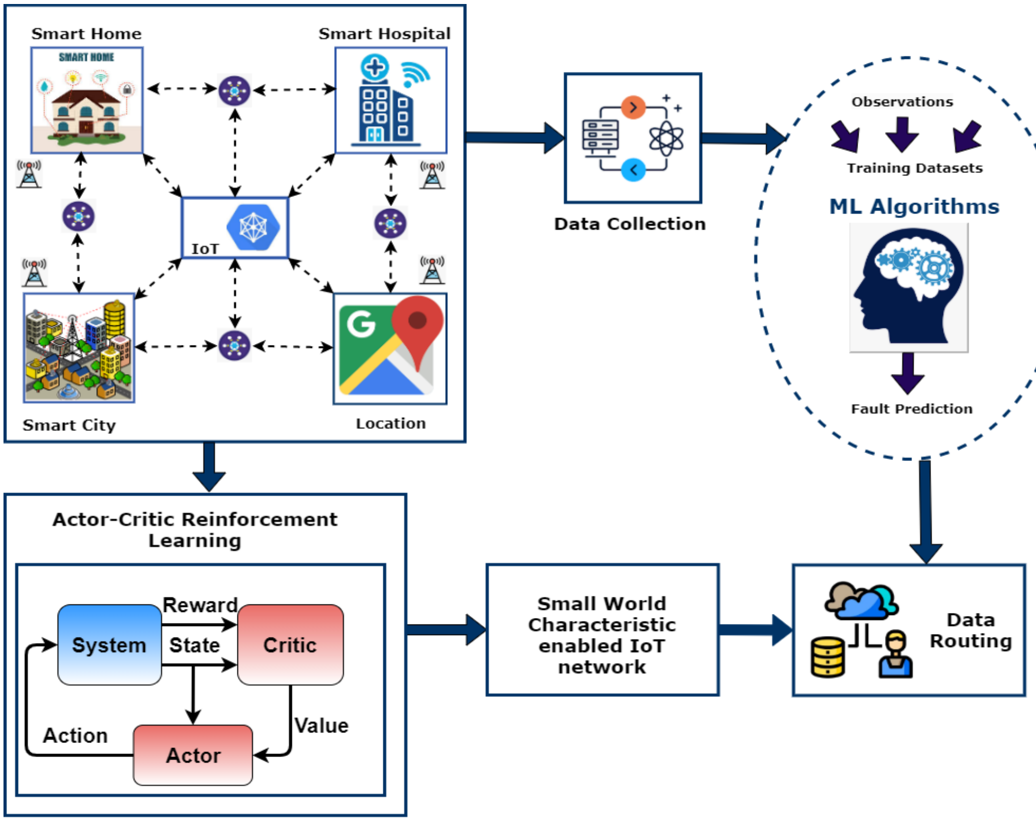

Figure 6.1: An illustration of joint node fault prediction and data routing over SW-IoT network using ML frameworks

heuristic algorithm for fault detection in WSNs. However, this approach has limitations in terms of fault detection accuracy and network lifetime. Predicting faulty nodes in advance can enhance network lifetime and QoS. The existing fault detection and prediction approaches consume a significant amount of energy, leading to the premature failure of the networks, which is subject to the premature death of the networks [44]. They also suffer from poor fault detection and prediction accuracy in highly faulty environments. Therefore, we need an algorithm to create the optimal small-world-enabled IoT network and a fast, reliable fault prediction mechanism that can operate in real time for efficient data routing. Figure 6.1 illustrates a smart IoT network where data from various sources, like smart homes, hospitals, and cities, are collected and processed using machine learning (ML) algorithms for fault prediction. An actor-critic reinforcement learning framework is used to introduce Small-World Characteristics (SWC) into the network, optimizing it for efficient data routing and overall network performance.

The contributions of this chapter are as follows:

- An actor-critic reinforcement learning approach is used to incorporate SWC in the IoT network. Small-world networks consist of long-range links that reduce average path length (APL) while maintaining a high average clustering coefficient (ACC).
- A joint method for node fault prediction and data routing in small-world IoT network is proposed, which significantly outperforms conventional methods.
- A comparative study is then conducted to analyze various fault prediction models, including the suggested method, such as density-based spatial clustering (DBSCAN).
- Data routing has been performed on the proposed small-world enabled IoT networks with dynamic fault node prediction to evaluate network performance in terms of lifetime, throughput, and latency. The proposed method gives significantly better results than existing approaches.

6.1 Preliminary of Small-World Phenomena

A small-world network (SWN) is a way of organizing connections such that everything stays close and easy to reach, even if the network gets really big. While introducing small-world phenomena in an IoT network, long-range links are added that bring the network to a small-world state. Long-range links enhance the network's data transmission capability with less energy consumption and low latency, enhancing its lifetime and throughput. Moreover, it enhances network robustness and resilience, allowing the system to withstand random failures. To incorporate the most optimal SWC into the network, we require an algorithm capable of efficient exploration in this complex environment. This leads us to the soft actor-critic (SAC) reinforcement learning (RL) algorithm. A network is characterized by its APL and ACC. An efficient SWN should have a low APL and a high ACC, leading to a high small-worldness (SW), which is the ratio of ACC and APL. The terminologies and their definitions are shown in Table 6.1.

Table 6.1: Terminologies and definitions

Symbols/Parameters	Meaning
$\mathcal{N}$	Set of all nodes in the network
N, M	Number of nodes and gateways
l, w	Length and width of the network
L	Average Path Length (APL)
C_{av}	Average Clustering Coefficient (ACC)
$d_{i,j}$	Distance between ith and jth nodes
C_i	Clustering coefficient of the ith node
$\mathcal{E}$	Environment of RL formulation
G	Network adjacency matrix
$\mathcal{A}$	Action space in RL formulation
S	Set of all states in RL formulation
$R(s_t, a_t)$, (w_1, w_2)	Reward function and reward parameters
θ, ϕ	Parameters of the policy network and Q-function
α, π_θ	Temperature parameter and policy network
Q_ϕ	Q-function
$\mathcal{D}$, $\mathcal{C}$	Replay buffer and its capacity
s_t, a_t, r_t	State, action, and reward at time t
s_{t+1}	Next state at time $t+1$
N_{tr}, y_i	Mini-batch size and Q-function target
γ, $\mathcal{H}_{\text{target}}$	Discount factor and target entropy
a'_{i+1}	Action sampled from policy at next state
S_{avg}	Average size of a data packet
τ	Soft update parameter
sec	Second

6.1.1 Average Path Length (APL):

APL represents the average length of the shortest paths between all pairs of nodes in the network, where $d_{i,j}$ in the formula is the distance between nodes i and j.

$$L = \sum_{\substack{i,j \leq N \\ i \neq j}} \frac{d_{i,j}}{N(N-1)}.$$

6.1.2 Average Clustering Coefficient (ACC):

It denotes the average clustering of all the nodes present in the network, where clustering of a node quantifies how close its neighbours tend to form a complete graph. The clustering coefficient of node i is given as:

$$C_i = \frac{\text{number of triangles formed with node } i}{\text{number of triads formed with node } i}.$$

Therefore, ACC (C_{av}) is given as:

$$C_{av} = \frac{1}{N} \sum_{i \in N} C_i \tag{6.1}$$

6.2 Network Model

This work considers an IoT network as shown in Figure 6.2 with N number of sensor nodes and M number of gateway in the area of size $l \times w\ m^2$ where l and w (in meters (m)) are the length and width of the network. The location of each node is denoted by red color, and the green lines represent the link between the nodes. The primary objective of this work is to strategically select node pairs for establishing long-range links, thereby optimizing the network structure to achieve an optimal small-world topology.

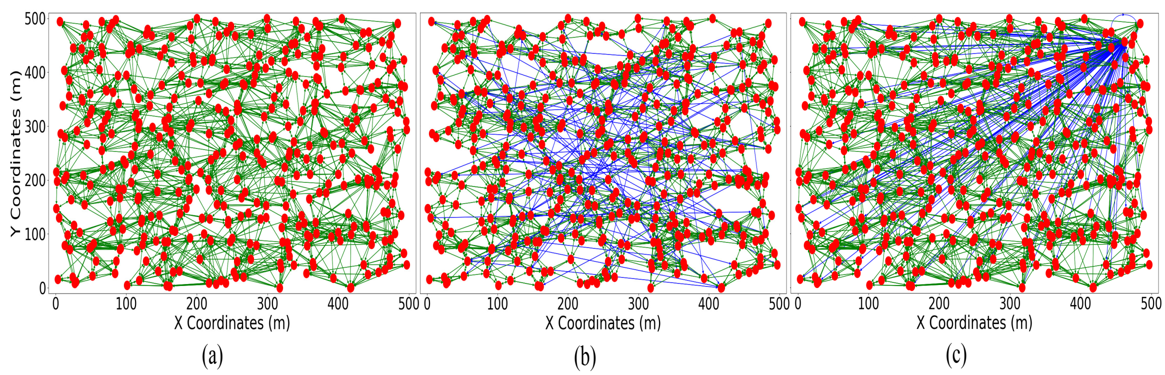

Figure 6.2: An illustration of **(a)** conventional IoT network, **(b)** small-world IoT network after introducing links among the sensor nodes, and **(c)** small-world IoT network after introducing links between node-gateway pairs

6.3 Node-Fault Prediction Assisted Data Routing over SW-IoT Network

Exploring the integration of ML within IoT networks, this section addresses the implementation of SWC into the network using an actor-critic RL algorithm. Thereafter, a joint method for dynamic node fault prediction and data routing within small-world IoT networks using advanced ML frameworks is discussed. This section has been divided into two subsections: (1) RL framework for introduction of SWC and (2) ML framework for node fault prediction and data routing.

6.3.1 RL Framework for Introduction of SWC

To infuse networks with small-world properties, this framework adapts actor-critic RL techniques to dynamically adjust network topology. It aims to optimize communication and resource utilization in complex systems like IoT networks. RL is defined using an agent, a set of states, and a set of actions. In RL, each action taken by the agent is associated with a reward. These rewards accumulate over time as the agent continues to take action. An agent learns to decide actions in an environment to maximize the reward.

Agent: The agent, represented by the SAC policy network π_θ, is responsible for deciding which pairs of nodes should be selected to introduce long-range links in the IoT network. The policy network observes the current state of the network and learns to add links that improve the network's topology.

State: The state s_t of the environment $\mathcal{E}$ is defined by the current configuration of the IoT network, encapsulated in a flattened 2D adjacency matrix G. This matrix represents the connectivity between node pairs and serves as input to the agent's decision-making process.

Action: An action a_t corresponds to selecting a pair of nodes (N_a, N_b) from the action space $\mathcal{A}$, where $\mathcal{A}$ represents the set of all possible node pairs that can potentially form long-range links in the network. The agent's goal is to strategically connect these nodes to enhance the network's topology.

Environment: The simulated IoT network acts as the environment $\mathcal{E}$ for the RL formulation. The agent interacts with this environment by proposing node pairs for link creation. Each action modifies the network topology, leading to a transition to a new state based on the action taken.

Reward: The reward function $R(s_t, a_t)$: $S \times \mathcal{A} \rightarrow \mathbb{R}$ is designed to guide the agent's actions. It encourages the formation of long-range links that decrease the APL of the network while maintaining a high ACC. It is presented as $R(s_t, a_t) =$

$w_1 C_{av} + \frac{w_2}{L}$. The agent's objective is to maximize the cumulative reward over time, i.e., $\max_{\theta,\phi} \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} R(s_t, a_t)\right]$, thereby optimizing the network towards an optimal small-world topology. Soft actor-critic, as shown in *Algorithm 6.1*, is a model-free off-policy RL algorithm having one policy network (actor) π_θ and one or more value networks (critics) Q_{ϕ_1} and Q_{ϕ_2}. All the networks are deep neural networks containing two fully-connected layers having 256 neurons and ReLU activations. The policy network directly generates the action $a_t \in \mathcal{A}$ given the current state $s_t \in \mathcal{S}$, while the value networks estimate the expected cumulative rewards R_t of being in a particular state and following the policy. One key feature of SAC is its focus on maximizing the entropy $\mathcal{H}(\pi_\theta)$ of the policy. Entropy regularization encourages exploration by adding

Table 6.2: Small-Worldness (SW) Analysis

Algorithm	**Links connection**	**APL**	**ACC**	**SW**
Conventional	-	7.039	0.585	0.083
SAC Small-World	To Server	3.399	0.569	0.1674
	Inter Node	3.953	0.532	0.1346

randomness to action selection. The policy is updated to maximize the expected reward $\mathbb{E}[r_t]$ while also considering the policy's entropy $\mathcal{H}(\pi_\theta)$. This balance between reward maximization and exploration helps the agent find a good trade-off between exploration and exploitation. The value function learns in a way that minimizes the squared Bell-

Table 6.3: Performance Comparison of Various RL Algorithms towards Introducing SWC

Parameter	**SAC**	**Q-Learning**	**PPO**
APL	3.991	5.621	3.514
ACC	0.541	0.585	0.493
SW	0.135	0.104	0.140
Inference Time (s)	10.78	37.54	40.08

man error while maximizing the policy's entropy $\mathcal{H}(\pi_\theta)$. It introduces a temperature parameter α that scales the entropy term in the objective function. The objectives of SAC include maximizing the expected cumulative reward $\mathbb{E}[r_t]$, maximizing entropy $\mathcal{H}(\pi_\theta)$, and minimizing the discrepancy between the value functions of the critics and the target value networks. Figures 6.2b and 6.2c show the optimal small-world networks generated by the proposed method. Table 6.2 shows the change in APL, ACC, and SW values after introducing small-world phenomena in the network. Table 6.3 compares the proposed algorithm with other popular algorithms used for introducing the small-world characteristics. The proposed approach successfully produces optimal results with the least inference time.

6.3.2 ML Framework for Node Fault Prediction & Data Routing

In large-scale IoT networks, sensor nodes often have limited energy and may fail during data transmission. To enhance efficiency and energy use, faulty nodes are identified and excluded. This begins with data preprocessing, where input data is standardized. Density based spatial clustering (DBSCAN) is then used to detect dense clusters and manage irregular shapes and outliers, labeling nodes outside the primary cluster as faulty. The fault level of each cluster is computed, and nodes with high fault levels are removed from the network. A faulty node dataset is simulated by introducing noise into the network properties.

Algorithm 6.1 Soft Actor-Critic (SAC) Algorithm for Introducing Small-World Characteristics in an IoT Network

1. **Initialize:** Network with nodes $\mathcal{N}= \{N_1, N_2,\cdots,N_j,\cdots,N_N\}$, Actor network π_θ, Critic networks Q_{ϕ_1} and Q_{ϕ_2}, target networks $Q_{\phi_1}^{\text{target}}$ and $Q_{\phi_2}^{\text{target}}$, temperature parameter α
2. **Initialize:** Replay buffer $\mathcal{D}$ with capacity $\mathcal{C}$
3. **Initialize:** Learning rates $\lambda_\theta, \lambda_\phi, \lambda_\alpha$, discount factor γ, target smoothing coefficient τ
4. **For** each episode **do**
5.   **For** each time step t **do**
6.   Get network adjacency matrix G. Initial state $s_0 = G$
7.   Note initial APL and ACC as L and C_{av}.

8. Select action $a_t \sim \pi_\theta(\cdot|s_t)$

9. Execute action a_t by adding new link. Get new matrix G'

10. Observe reward r_t and next state $s_{t+1} = G'$

11. Note new APL and ACC and mark as L' and C_{av}'.

12. $\Delta L = L - L'$ and $\Delta C_{av} = C_{av} - C'_{av}$

13. Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$

14. Sample a random minibatch of N_{tr} transitions (s_i, a_i, r_i, s_{i+1}) from $\mathcal{D}$

15. Compute target value:

$$y_i = r_i + \gamma \left(\min_{j=1,2} Q_{\phi_j}^{\text{target}}(s_{i+1}, a'_{i+1}) - \alpha \log \pi_\theta(a'_{i+1}|s_{i+1}) \right)$$

16. Update Critic networks by minimizing the loss:

$$L_Q = \frac{1}{N_{tr}} \sum_i \left(Q_{\phi_j}(s_i, a_i) - y_i \right)^2$$

17. Update Actor network by minimizing the loss:

$$L_\pi = \frac{1}{N_{tr}} \sum_i \left(\alpha \log \pi_\theta(a_i|s_i) - Q_{\phi_1}(s_i, a_i) \right)$$

18. Update temperature parameter α by minimizing the loss:

$$L_\alpha = \frac{1}{N_{tr}} \sum_i \left(-\alpha \left(\log \pi_\theta(a_i|s_i) + \mathcal{H}_{\text{target}} \right) \right)$$

19. Soft update target networks:

$$\phi_j^{\text{target}} \leftarrow \tau \phi_j + (1 - \tau) \phi_j^{\text{target}} \quad \text{for } j = 1, 2$$

20. **If** $\Delta C_{av} > \Delta L$

21. **break**

22. End If

23. End For

24. End For

To detect anomalies, various ML models—including one-class SVM, elliptic envelope, isolation forest, local outlier factor (LOF), and DBSCAN—are trained and evaluated. In small-world IoT (SW-IoT) networks, energy-efficient routing is achieved through three steps: weight initialization, shortest path finding via Dijkstra's algorithm, and weight updates. In weights initialization, a weight matrix representing the network graph is initialized. The weights in this matrix correspond to the energy cost of communication between nodes. Thereafter, Dijkstra's algorithm finds the shortest path to the sink node. In this context, the "shortest path" refers to the path with the minimum cumulative weight, where the weights reflect the energy costs. This step ensures that the chosen path minimizes the overall energy consumption for data transmission, leveraging nodes with higher energy reserves and thus balancing the network load. Once the RL algorithm is trained, we input the current network graph to obtain an optimized configuration with additional small-world links. We then initiate routing by randomly selecting nodes and transferring data packets to the server node via a multi-hop. At each hop, the trained fault-node prediction algorithm verifies if the node is active; if not, Dijkstra's algorithm recalculates an alternative path of active nodes to the server. Finally, after a successful data transmission along the chosen path, the weights are updated in the weight matrix. The RL algorithm trains in 1 hour and performs inference in 10.78 seconds on a single NVIDIA RTX 3090 GPU, enabling efficient data routing with fault-node prediction.

6.3.3 Experimental Set-Up

The network size considered for simulation is 500×500. In this area, a total of 400 sensor nodes are deployed. Data routing is done using a multi-hop method, starting from randomly selected nodes to the server node. This approach allows us to study network parameters and analyze the performance of the proposed small-world network with dynamic node fault prediction. The routing algorithm runs until 50% of the nodes die out. These parameters are checked under the following settings: (1) data transmitted: 2000 bits, (2) transmission rate: 250 kbps, (3) initial energy: 2 J, (4) data arrival rate: 100 kbps, and (5) service rate: 1000 kbps.

6.3.4 Measurement Models

6.3.4.1 Network Lifetime

The total time taken in each iteration of data transmission until the routing process terminates, i.e., $T_{total} = \sum_{i=1}^{n} t_i$.

6.3.4.2 Latency

The average time to deliver a data packet, i.e., $T_{avg} = \frac{\sum_{i=1}^{n} t_i}{n}$

6.3.4.3 Throughput

The end-to-end throughput measures the number of packets received at the destination per second.

6.3.4.4 Data Transmitted

The total amount of data (in KB) transmitted by all nodes within the network until the routing process concludes.

6.3.4.5 Data Received

The total amount of data (in KB) received by all the receiver nodes within the network until the routing process concludes.

6.3.5 State-of-the-art Methods

6.3.5.1 For Node Fault Prediction

The suggested method is compared with the following state-of-the-art methods.

1. *One-Class SVM:* It is an unsupervised technique for anomaly detection. It learns to identify a specific class of data points in unlabeled data by creating a boundary around them. Any data falling outside the boundary is considered an anomaly.

2. *Elliptic Envelope*: It establishes an imaginary elliptical boundary around a specified dataset. Data points within this boundary are classified as normal, while those outside are identified as outliers.
3. *Isolation Forest (IF)*: Isolation forest is an unsupervised anomaly detection method that utilizes decision trees. It isolates anomalies by randomly splitting the data and observing how easily each data point can be separated from the rest. Points that are easily isolated (shorter branches) are considered more likely to be anomalies.
4. *Local Outlier Factor (LOF)*: It is an unsupervised anomaly prediction method that computes the local density deviation of a given data point with respect to its neighbors. Data points with significantly lower density than their neighbors are identified as outliers.

6.3.5.2 For Data Transmission

The proposed method, i.e., SAC derived small-world network with dynamic node fault prediction (SAC-NFP) is compared with the following state-of-the-art methods.

1. *Direct Transmission*: In direct transmission, each node sends data directly to the server node, leading to high energy consumption, often resulting in early depletion of the sensor node's energy, a significant reduction in network lifetime, and failure to deliver data successfully.
2. *Q-Learning Small-World*: Q-learning is another RL algorithm for introducing SWC. It requires extensive exploration to converge and adapts slowly to topology changes in dynamic IoT environments. This results in suboptimal long-range link placements under time-varying conditions, leading to reduced throughput and lifetime, and higher latency compared to SAC-NFP.
3. *PPO Small-World*: Proximal policy optimization (PPO) is another popular actor-critic algorithm for introducing SWC. It performed better than Q-learning; how-

ever, its conservative policy update constraints limit its ability to rapidly adapt long-range link configurations under fast-changing network conditions. As a result, PPO still exhibits higher latency and lower lifetime than SAC-NFP.

6.3.6 Numerical Results

6.3.6.1 Node Fault Prediction Analysis

The performance of the suggested method for node fault prediction is evaluated in terms of accuracy, precision, recall, and F-1 score.

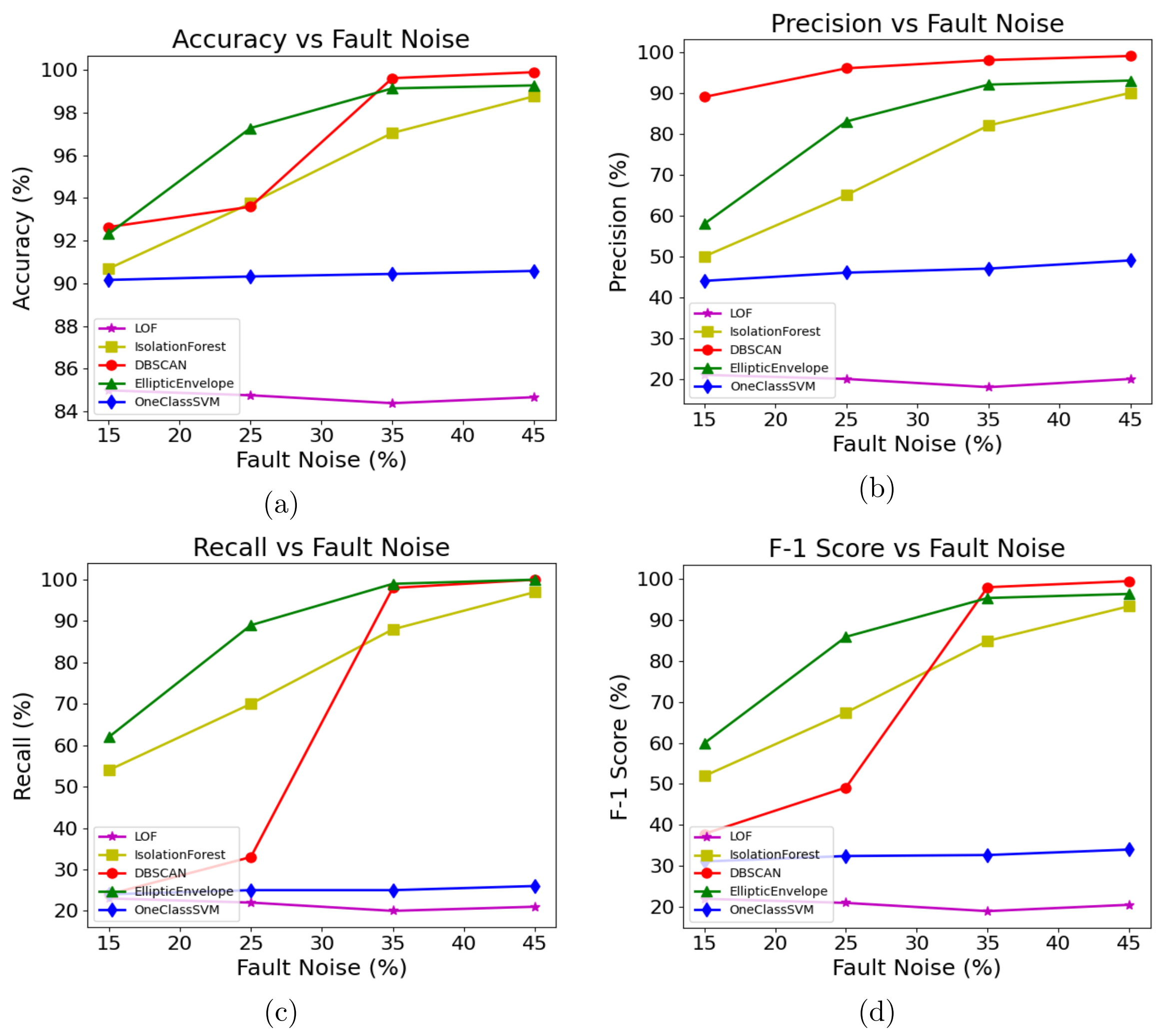

Figure 6.3: Performance analysis of various algorithms in the presence of fault noise across different metrics. (a) accuracy vs fault noise, (b) precision vs fault noise, (c) recall vs fault noise, and (d) F1-Score vs fault noise, respectively

Figure 6.3 shows the performance of anomaly prediction algorithms under varying fault noise levels. As shown in Figure 6.3a, at higher faulty noise levels, DBSCAN gives more accuracy than the other methods, like LOF, isolation forest, elliptic envelope, and one-class SVM. Similarly, it gives more than 90% precision, which is better than other methods, as shown in Figure 6.3b. Moreover, Figure 6.3c concludes that the recall value is lower for DBSCAN at low fault noise levels and higher at high fault noise levels. Additionally, as shown in Figure 6.3d, the F-1 score for DBSCAN is more than 90%, which is greater than other methods at high fault noise levels. Therefore, DBSCAN is the most efficient approach for this work. It outperforms the other popular approaches and performs better on accuracy, precision, recall, and F-1 score metrics.

6.4 Performance Evaluation

6.4.1 Quality-of-Service Analysis

Table 6.4: Performance comparison of the proposed method with various data routing algorithms under similar network conditions

Algorithm	Links connection	Lifetime	Latency (s)	Throughput (bits/s)	Data Transmitted (kbits)	Data Received (kbits)
Conventional	–	5312	0.108	18469.78	108206	97582
Direct Transmission	–	258	0.009	217972.04	916	0
SAC Small-World (proposed)	**To Server**	**5588**	**0.0918**	**21773.25**	**98532**	**88264**
	Inter Node	**5459**	**0.0892**	**22418.06**	**108686**	**97768**
Q-Learning Small-World	To Server	4729	0.108	18459.28	94634	85354
	Inter Node	4948	0.112	17788.45	103100	93198
PPO Small-World	To Server	4621	0.107	18671.13	91746	82276
	Inter Node	4766	0.110	18178.38	99880	90344

Table 6.4 compares the data routing performance of the proposed small-world network method using SAC with PPO and Q-learning, as well as with conventional network routing using multi-hop methods. The metrics considered include throughput, latency, network lifetime, data transmitted, and data received. The comparison is made for

two scenarios: data routing for the network having long-range links among the nodes and data routing for the network having long-range links between node-gateway pairs. Comparing SAC with PPO and Q-learning, SAC maintains a high throughput (∼22,000 bps), low latency (∼0.090 s), and a significantly higher network lifetime (∼5500 units). Due to its high lifetime, it's able to transmit more data, which is successfully received as indicated by the data received and the throughput metric. PPO and Q-learning networks suffer from low throughput and lifetime, and high latency, with PPO generally performing better than Q-learning but still lagging behind SAC-NFP. The conventional IoT network, when tested using multi-hop routing, shows significantly low throughput (18469.78 bps) and network lifetime (5312 s). Direct data transmission transmits data consuming large amounts of sensor node energies, leading to its early die out and failure to successfully transmit any data packet. This shows the advantage of long-range links in small-world networks, which optimize their topology for enhanced performance.

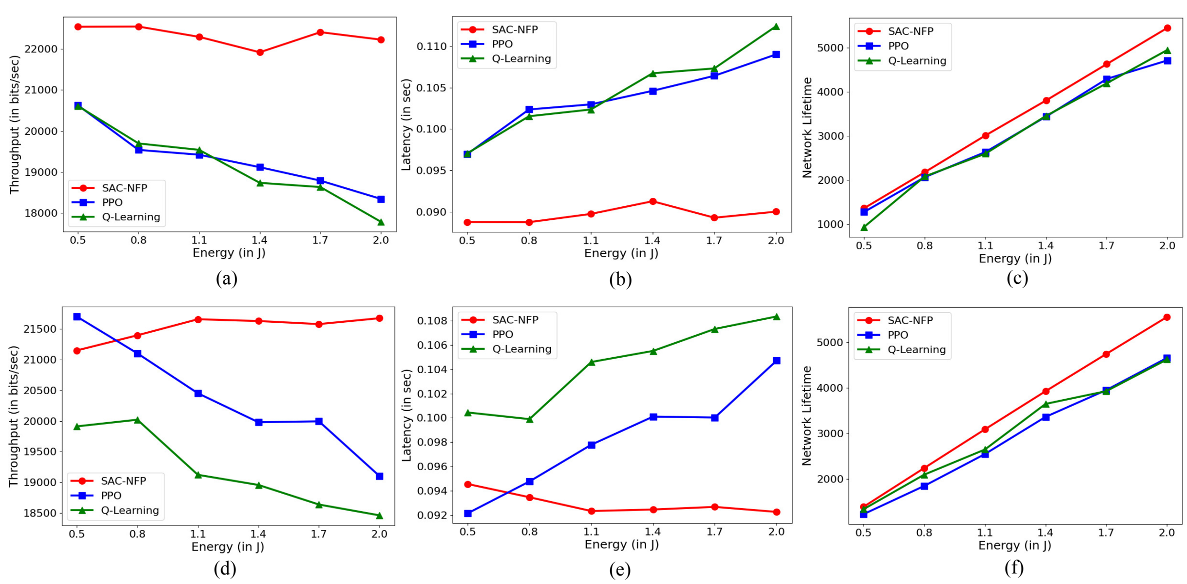

Figure 6.4: Performance analysis of various network parameters with varying initial energy of the sensor nodes using different methods of introduction of SWC. Figures (a, b, c) illustrate throughput, latency, and network lifetime variation when long-range links are introduced among the nodes, while Figures (d, e, f) depict the same performance when the long-range links are created between node-gateway pairs

Data routing for different initial sensor node energies is also carried out as shown in Figure 6.4. Figures 6.4(a, b, c) show the comparison of the proposed method (SAC-NFP) with Q-learning and PPO small-world networks when routing is performed between the network and Figures 6.4(d, e, f) show the comparison for routing performed with the server node. SAC-NFP outperforms both of them, taking a significantly higher throughput and network lifetime with lower latency for all energy values.

6.5 Conclusions

This work presents a machine learning-based approach to introduce small-world characteristics in IoT networks for efficient data routing with dynamic node fault prediction. Using an actor-critic reinforcement learning framework, the network gains SWC, which, combined with a DBSCAN-based fault prediction model, significantly improve network performance. Experimental results demonstrate superior network performance in terms of lifetime, throughput, and latency compared to existing methods, highlighting the potential of the proposed method for robust IoT systems. The proposed approach is currently limited to static IoT networks. As part of future work, novel methods can be proposed over time-varying IoT networks, investigating the node fault prediction and introduction of SWC.