

Article

Federated Reinforcement Learning with Hybrid Optimization for Secure and Reliable Data Transmission in Wireless Sensor Networks (WSNs)

Seyed Salar Sefati ^{1,2,*} , Seyedeh Tina Sefati ³ , Saqib Nazir ^{1,4} , Roya Zareh Farkhady ⁵ and Serban Georgica Obreja ¹¹ Telecommunications Department, Faculty of Electronics, Telecommunications and Information Technology, National University of Science and Technology POLITEHNICA Bucharest, 060042 Bucharest, Romania² Department of Software Engineering, Faculty of Engineering and Natural Science, Istinye University, Istanbul 34010, Türkiye³ Faculty of Electrical and Computer Engineering, University of Tabriz, 29 Bahman, Tabriz 51664, Iran; t.sefati@tabrizu.ac.ir⁴ Department of Computer Science, Edge Hill University, Ormskirk L39 4QP, UK⁵ Department of Computer Engineering, Institute of Higher Education Roshdiyeh, Tabriz 51368, Iran

* Correspondence: salar.sefati@elcom.pub.ro

Abstract

Wireless Sensor Networks (WSNs) consist of numerous battery-powered sensor nodes that operate with limited energy, computation, and communication capabilities. Designing routing strategies that are both energy-efficient and attack-resilient is essential for extending network lifetime and ensuring secure data delivery. This paper proposes Adaptive Federated Reinforcement Learning-Hunger Games Search (AFRL-HGS), a Hybrid Routing framework that integrates multiple advanced techniques. At the node level, tabular Q-learning enables each sensor node to act as a reinforcement learning agent, making next-hop decisions based on discretized state features such as residual energy, distance to sink, congestion, path quality, and security. At the network level, Federated Reinforcement Learning (FRL) allows the sink node to aggregate local Q-tables using adaptive, energy- and performance-weighted contributions, with Polyak-based blending to preserve stability. The binary Hunger Games Search (HGS) metaheuristic initializes Cluster Head (CH) selection and routing, providing a well-structured topology that accelerates convergence. Security is enforced as a constraint through a lightweight trust and anomaly detection module, which fuses reliability estimates with residual-based anomaly detection using Exponentially Weighted Moving Average (EWMA) on Round-Trip Time (RTT) and loss metrics. The framework further incorporates energy-accounted control plane operations with dual-format HELLO and hierarchical ADVERTISE/Service-ADVERTISE (SrvADVERTISE) messages to maintain the routing tables. Evaluation is performed in a hybrid testbed using the Graphical Network Simulator-3 (GNS3) for large-scale simulation and Kali Linux for live adversarial traffic injection, ensuring both reproducibility and realism. The proposed AFRL-HGS framework offers a scalable, secure, and energy-efficient routing solution for next-generation WSN deployments.

Keywords: Wireless Sensor Networks (WSNs); Federated Reinforcement Learning (FRL); secure routing; energy efficiency; anomaly detection

MSC: 68M25; 68M18; 68M12; 68M10; 97P80



Academic Editor: Jonathan Blackledge

Received: 5 September 2025

Revised: 25 September 2025

Accepted: 30 September 2025

Published: 6 October 2025

Citation: Sefati, S.S.; Sefati, S.T.; Nazir, S.; Zareh Farkhady, R.; Obreja, S.G. Federated Reinforcement Learning with Hybrid Optimization for Secure and Reliable Data Transmission in Wireless Sensor Networks (WSNs). *Mathematics* **2025**, *13*, 3196. <https://doi.org/10.3390/math13193196>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Wireless Sensor Networks (WSNs) are made up of large numbers of small, affordable sensor nodes that function with limited energy reserves. These nodes are often placed in remote or difficult environments where they collect information about physical or environmental conditions, such as temperature, humidity, movement, and air quality [1]. The gathered data are then relayed across multiple nodes until they eventually reach a central sink. Each sensor node typically includes a sensing device, a lightweight processor, a communication module, and a battery, all of which limit its processing power, memory, communication range, and overall energy supply [2]. These limitations make it necessary for WSNs to use routing protocols that are efficient, scalable, and able to extend the network's lifetime. Today, WSN technology is widely used in fields like environmental monitoring, healthcare, industrial automation, agriculture, military applications, and the development of smart cities. However, the decentralized structure and limited resources of these networks create considerable challenges for their design and long-term operation, especially in changing environments [3]. In 2025, most research is focused on improving energy efficiency, ensuring complete area coverage, reducing communication overheads, and enhancing the reliability of routing protocols. Maintaining these networks manually is often impractical, especially in large-scale or hard-to-reach deployments. Therefore, WSNs must be designed to operate autonomously and remain effective over long periods [4]. Table 1 defines the abbreviations and acronyms used in this paper, supporting the reader's understanding.

Table 1. List of abbreviations.

Abbreviation	Full Form
ABC	Artificial Bee Colony
ACO	Ant Colony Optimization
AES	Advanced Encryption Standard
AES-CTR	Advanced Encryption Standard-Counter Mode
AES-GCM	Advanced Encryption Standard-Galois/Counter Mode
AFRL-HGS	Adaptive Federated Reinforcement Learning with Hunger Games Search
BER	Bit Error Rate
BHA	Black Hole Algorithm
BHA-ACO	Black Hole Algorithm-Ant Colony Optimization
CBSO	Chaotic Bird Swarm Optimization
CH	Cluster Head
CIRP	Cluster-based Intelligent Routing Protocol
CP	Constant Power
CS	Cuckoo Search
CSMA/CA	Carrier-Sense Multiple Access with Collision Avoidance
CVSS	Common Vulnerability Scoring System
DBR	Depth-Based Routing Protocol
DECEM	Dynamic Energy-efficient Clustering and Data Aggregation for Efficient Multi-hop Communication
DDoS	Distributed Denial-of-Service
DoS	Denial-of-Service

Table 1. *Cont.*

Abbreviation	Full Form
DQELR	Deep-Q-Network-based Energy- and Delay-Aware Routing Protocol
DQNs	Deep Q Networks
DQN-EQSR	Double Deep-Q-Network-based Efficient Quality of Service Routing
DQIR	Deep-Q-Network-based Intelligent Routing
DTNs	Delay-Tolerant Networks
E2ED	End-to-End Delay
ECE	Expected Energy Consumed
ERE	Expected Remaining Energy
EWMA	Exponentially Weighted Moving Average
FA	Firefly Algorithm
FIFO	First In, First Out
FRL	Federated Reinforcement Learning
GA	Genetic Algorithm
GADA-LEACH	Genetic Algorithm-based Distributed Adaptive Low-Energy Adaptive Clustering Hierarchy Protocol
GAPSO-H	Genetic Algorithm and Particle Swarm Optimization-based Hybrid Routing
GASO	Green Anaconda Skill Optimization
GNS3	Graphical Network Simulator-3
HGS	Hunger Games Search
I-IoT	Intelligent Internet of Things
IDS	Improved Differential Search
IoUT	Internet of Underwater Things
MACs	Message Authentication Codes
MAD	Median Absolute Deviation
NORTH	Node-Optimized Routing Technique for Heterogeneous WSNs
NVD	National Vulnerability Database
ODM	Optimal Data Routing
PDR	Packet Delivery Ratio
PLR	Packet Loss Ratio
PSO	Particle Swarm Optimization
QLFR	Q-Learning-based Localization-Free Routing Protocol
QoS	Quality of Service
QTAR	Q-learning-based Topology-Aware Routing
RFO	Red Fox Optimization
RL	Reinforcement Learning
RPL	Routing Protocol for Low-Power and Lossy Networks
RSSI	Received Signal Strength Indicator
RTT	Round-Trip Time

Table 1. Cont.

Abbreviation	Full Form
SDN	Software-Defined Networking
SECRs	security-aware strategies
SNR	Signal-to-Noise Ratio
TCHS	Trust-based Cluster Head Selection
TDMA	Time-Division Multiple Access
TM	Trust Management
WSNs	Wireless Sensor Networks

Energy-efficient and attack-resilient routing in WSNs aims to reduce power consumption while making sure that data remain secure and reliable, even when threats are present [5,6]. Since sensor nodes usually run on limited battery power and are often placed where access is difficult, it is important to design routing strategies that save energy and defend against attacks like sinkhole, Sybil, and selective forwarding [7]. Good routing protocols share communication tasks across nodes to prevent batteries from draining too quickly, and they include security features such as trust scoring, anomaly detection, and secure route selection to stop malicious actions. Modern approaches combine energy-efficient routing with methods like machine learning, trust-based decisions, and metaheuristic optimization so the network can adjust to changing conditions [8,9]. For example, nodes with higher energy and stronger trust records are chosen more often for routing, while less reliable nodes are avoided. Clustering and multipath routing are also used to balance network traffic and make the system more robust against attacks or failures [10]. Using these combined techniques is especially valuable in fields like environmental monitoring, military defense, and smart infrastructure, where both network lifespan and security are essential.

In this paper, we propose AFRL-HGS, a security-aware and energy-efficient routing framework for WSNs that tightly couples tabular Q-learning at each node, FRL at the sink, and a binary HGS initializer for CH selection and routing. Nodes operate as RL agents over discretized state features (residual energy, distance to sink, congestion, path quality, and security), learn next-hop decisions with an ϵ -greedy policy, and periodically upload compressed Q-tables. The sink performs adaptive, energy-/return-weighted aggregation and returns a global model that is blended locally via a Polyak update to avoid catastrophic forgetting. A lightweight anomaly/trust module-fed by EWMA residuals of RTT and loss-constrains the action set to security-admissible paths, while the control plane maintains RoutingTableUP/DOWN through dual-format HELLO and hierarchical ADVERTISE/SrvADVERTISE messages, which are all fully accounted for in the radio-energy model. We evaluate AFRL-HGS in a hybrid testbed (GNS3 for repeatable large-scale simulation, Kali Linux for live Denial-of-Service (DoS), spoofing, and sinkhole injection and trace replay). The main contributions are as follows:

- **AFRL-HGS architecture:** We introduce a Hybrid Routing framework that integrates per-node tabular Q-learning with adaptive FRL aggregation and binary HGS initializer that jointly optimizes Cluster Heads and feasible routes.
- **Security as a constraint via trust/anomaly fusion:** We design a lightweight link-trust mechanism that fuses smoothed reliability and anomaly flags. Security acts as an admissibility constraint on actions (not as a reward term), decoupling objectives and improving robustness to spoofing, selective forwarding, and sinkhole attacks.

- **Energy-accounted control plane and routing tables:** We formalize dual-format HELLO (detailed for CH/sink; lightweight for nodes) and hierarchical ADVERTISE/SrvADVERTISE to populate RoutingTableUP/DOWN, with all control traffic explicitly costed in the first-order radio model.
- **Adaptive, reliability-aware federated aggregation:** Global Q-tables are computed with weights that depend on residual energy and recent episodic return, and are blended locally (Polyak) to mitigate drift and catastrophic forgetting in resource-constrained nodes.

2. Related Studies

Security remains a major challenge in WSNs, along with other concerns such as limited energy, unreliable data, and unstable communication. Figure 1 categorizes attacks into directed and indirect, where directed attacks lead to packet dropping, while undirected attacks result in different misbehaviors [11]. These attacks are generally categorized as either direct or indirect. In direct attacks, adversaries target the Trust Management (TM) system itself, causing poor performance and faulty routing decisions that weaken the network's overall effectiveness. If the TM process cannot reliably detect malicious nodes, especially during deceptive behaviors like on-off attacks or false feedback, these nodes may be mistakenly trusted to forward data, which leads to higher packet loss [12]. The configuration of TM systems involves several adjustable parameters, such as adaptive trust dynamics, forgetting factors, trust thresholds, reward and penalty systems, trust evaluation intervals, and ways to combine trust values. How well a TM system works depends on how carefully these parameters are tuned for the network's specific needs and threats. Regardless of the approach, a strong TM solution must be able to withstand basic security threats. The main goal of Trust Management is to identify nodes that cannot be trusted and limit their influence, which improves the overall security and stability of the WSN [13]. For example, if a node starts dropping packets on purpose, its trust score should decrease automatically.

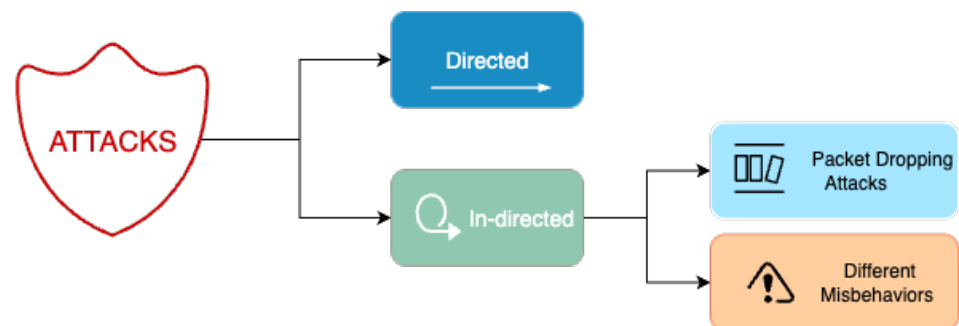


Figure 1. Attack in WSNs.

2.1. Security-Aware Routing

Sameer Dixit et al. [14] proposed a cluster-based routing protocol for WSNs that uses both Red Fox Optimization (RFO) and security-aware strategies (SECRs). Their method relies on the RFO algorithm to select the most suitable CHs by considering multiple factors, including trust value, node centrality, residual and initial energy, and the node's distance to the sink. To secure communication, the protocol uses adaptive Advanced Encryption Standard (AES) encryption. In low-energy situations, it uses Advanced Encryption Standard-Counter (AES-CTR) mode, while in high-threat conditions, it switches to Advanced Encryption Standard-Galois/Counter Mode (AES-GCM). The protocol manages keys through dynamic updates using secure hash functions, and it ensures data integrity with Message Authentication Codes (MACs). The SECR protocol achieves better energy

efficiency, longer network lifetime, and greater resistance to Distributed Denial-of-Service (DDoS) attacks compared to existing solutions such as Dynamic Clustering approach for Energy-hole Mitigation (DECEM), Cluster-based Intelligent Routing Protocol (CIRP), Number of cluster nodes (NORTH), Genetic Algorithm and Particle Swarm Optimization-based Hybrid Routing (GAPSO-H), and Genetic Algorithm-based Distributed Adaptive Low-Energy Adaptive Clustering Hierarchy Protocol (GADA-LEACH). However, it assumes that nodes do not move and that the wireless environment is ideal. Its DDoS detection uses a fixed threshold, which can limit its flexibility in changing conditions. The reliance on the RFO algorithm also adds computational complexity, especially as the number of nodes and iterations increases.

Ferhat Arat et al. [15] developed a security-focused Routing Protocol for Low-Power and Lossy Networks (RPL) framework that uses new objective functions based on real-time risk and vulnerability information to guide routing decisions. Their approach combines data from the National Vulnerability Database (NVD) and Common Vulnerability Scoring System (CVSS) to assess risks at both the node and path levels, helping to detect and prevent rank attacks in IoT networks. The method dynamically updates routing paths by isolating nodes that manipulate rank values and favoring routes through low-risk nodes. Key advantages of this technique include better detection of rank attacks, the ability to assess risk at multiple levels, and the use of real-world vulnerability data, all of which make the protocol more secure and adaptable. However, the framework has limitations: it is designed for static network topologies, has not been tested on large networks beyond 200 nodes, and involves higher computational costs due to complex risk calculations and reliance on APIs. Scalability and real-time adaptability in dynamic environments are still open challenges.

D. Karunkuzhali et al. [16] proposed a comprehensive routing method for IoT-enabled WSNs in smart cities. Their two-phase approach first uses Chaotic Bird Swarm Optimization (CBSO) to cluster sensor nodes and Improved Differential Search (IDS) to select trusted CHs. In the next phase, they apply the Optimal Data Routing (ODM) algorithm for efficient paths and use lightweight signcryption for secure data transmission. Compared to benchmark methods, the NS2 simulation results show significant improvements. These include up to 90.8% less energy consumption, 66.8% longer network lifetime, and up to 80.1% lower latency. The approach blends optimization, trust, and lightweight cryptography for secure and adaptive routing. However, its real-world scalability, flexibility, and performance—especially in highly dynamic or diverse smart city environments—remain untested despite strong simulation outcomes.

2.2. Energy-Aware Routing

Sefati, et al. [17] proposed a cluster-based data transmission method for WSNs that uses the Black Hole Algorithm (BHA) to select CHs and Ant Colony Optimization (ACO) for routing between clusters. The BHA selects CHs based on factors like residual energy, buffer size, and distance to the center of the network, while ACO finds efficient multi-hop paths from CHs to the sink. Their approach, tested in MATLAB R2016b, showed better performance than using ACO alone, the Firefly Algorithm (FA), or Cuckoo Search (CS), with improvements in network lifetime, remaining energy, and the number of packets delivered to the sink. The main strength of this method is its ability to balance energy use and extend the network's life through better CH selection and routing. However, using both metaheuristics increases computational complexity, especially during the setup and routing phases, which can be a drawback in short-term scenarios or small networks.

Sefati et al. [18] introduced a CH selection and routing algorithm for WSNs that uses Software-Defined Networking (SDN) and game theory. Their method features a game-

theoretic clustering approach managed by a central SDN controller. CHs are chosen using utility functions that consider positional advantage, throughput, and remaining energy. For regular data, the protocol uses energy-aware routing based on profit estimation from game theory. High-priority data packets are sent through a First In, First Out (FIFO)-based queue. This approach offers several benefits, including better energy balance, higher PDR, longer network lifetime, and adaptability to both small- and large-scale networks, thanks to SDN's flexibility. However, the method can face limits with real-time responsiveness if the network topology changes rapidly. The use of game-theoretic calculations also adds computational complexity.

P. K. Udayaprasad et al. [19] proposed an energy-efficient routing system for large-scale Intelligent IoT (I-IoT) networks using a distributed SDN-AI framework. Their approach combines three algorithms: Artificial Bee Colony (ABC) for finding the best mobile sink path, Particle Swarm Optimization (PSO) to identify optimal data-gathering points, and Genetic Algorithm (GA) to cluster nodes with different capabilities. The system is deployed on a cloud-based, three-layer SDN architecture and uses a Time-Division Multiple Access (TDMA) sleep/awake schedule for energy-aware routing, dynamic scheduling, and load balancing. Compared to OMS-LB and RM-LB protocols, this method improves data delivery by 95%, reduces energy use by 78%, and extends network lifetime by 63%. However, the reliance on a central SDN controller can introduce latency and create a single point of failure, especially in real-time or adversarial environments. The lack of real-world testing and the assumption that nodes cannot recharge their energy also limit its use in highly dynamic or self-sustaining IoT networks.

2.3. QoS-Aware Routing

Meiyan Zhang et al. [20] proposed the Double deep Network-based Efficient Quality of Service Routing protocol (DQN-EQSR) protocol for the Internet of Underwater Things (IoUT). This method uses two coordinated Deep Q Networks (DQNs): one for selecting relay nodes and another for controlling acoustic communication. The system works with a multi-dimensional state space that includes node and packet information, and it uses a reward function based on energy cost, link quality, End-to-End Delay (E2ED), and packet priority. Thanks to this dual-network design, sensor nodes can adapt their routing strategies in complex underwater environments. DQN-EQSR performs better than protocols such as Depth-Based Routing Protocol (DBR), Deep Q-network-based Intelligent Routing (DQIR), Q-learning-based Localization-Free Routing protocol (QLFR), Q-learning-based Topology-Aware Routing (QTAR), QELAR, and Deep-Q-Network-based Energy- and Delay-Aware Routing Protocol (DQELR) in PDR, residual energy, average delay, and network lifetime. However, the method's higher computational demands, longer inference time, and need for precise parameter tuning may limit its practicality for resource-constrained underwater nodes, which are common in IoUT systems.

Basma Mostafa et al. [21] improved multi-constrained Quality of Service (QoS) routing in WSNs by enhancing the RPL protocol without changing its core design. They noted that strict QoS path computation is very demanding for sensor nodes, so they proposed an edge-assisted system where edge servers handle complex routing tasks. This method uses algorithms for conflict-free routing, backup link creation, and Pareto-optimal path selection under QoS limits. Their solution preserved far more QoS-compliant nodes in tests and showed strong scalability across various network conditions. The framework is especially useful for real-time, QoS-sensitive WSN applications, supporting dynamic reconfiguration and lightweight monitoring. Future plans include benchmarking against other protocols and expanding to hybrid fog and edge-cloud networks.

Ashapu Bhavani et al. [22] developed the Multi-objective Green Anaconda Skill Optimization (GASO) algorithm for energy and cache-based QoS-aware routing in Delay-Tolerant Networks (DTN) tailored for IoT settings. GASO combines Green Anaconda Optimization and the Skill-Optimization Algorithm to select optimal CHs and routes, optimizing variables like cache ratio, predicted and residual energy, throughput, distance, trust, and delay, while considering realistic energy and mobility models. This approach lowers communication distance, energy use, and delay while increasing throughput and offers strong scalability and resilience to attacks like wormhole and blackhole. However, the hybrid design adds computational complexity, which may affect real-time performance in large or dynamic networks. Its Trust Management is also basic, lacking deep security integration, and practical deployment still faces challenges due to hardware limits and interference.

Table 2 presents a comparison of the recent papers discussed, highlighting the advantages and disadvantages of each method.

Table 2. Comparison of recent trust, energy, and QoS-aware routing methods for WSNs.

Author	Proposed Method	Advantages	Disadvantages
Sameer Dixit et al. [14]	Cluster-based routing with Red Fox Optimization (RFO) and security-aware strategies using adaptive AES encryption modes	Improved energy efficiency, longer network lifetime, resistance to DDoS, secure key updates	Assumes static topology, fixed DDoS thresholds, and introduces RFO-related computational overhead
Ferhat Arat et al. [15]	Security-aware RPL with risk-based objective functions using CVSS and NVD data	Better detection of rank attacks, multi-level risk assessment, adaptable to known vulnerabilities	High computational cost, untested on large/dynamic networks, designed for static environments
D. Karunkuzhali et al. [16]	Two-phase method using CBSO for clustering and IDS for CH selection, ODM for path finding	Reduces energy consumption by 90.8%, lowers latency, combines optimization with lightweight cryptography	Lacks real-world testing; uncertain scalability in dynamic smart city environments
Sefati et al. [17]	Cluster-based method using BHA for CH selection and ACO for inter-cluster routing	Better network lifetime and packet delivery, balances energy consumption effectively	Increased complexity due to dual-metaheuristic setup, not optimal for small-scale/short-term scenarios
Sefati et al. [18]	SDN + game theory-based CH selection and energy-aware routing with FIFO queue	Higher PDR, adaptive to various network scales, energy-balanced operation	Complex calculations, possible delays in rapidly changing topologies
P. K. Udayaprasad et al. [19]	Distributed SDN-AI framework using ABC, PSO, and GA for routing, clustering, and mobile sink pathing	95% higher data delivery, 78% less energy use, 63% longer network life	Central controller adds latency and single-point-of-failure risk; assumes non-rechargeable nodes
Meiyan Zhang et al. [20]	Dual DQN-based EQSR protocol for IoUT routing and communication control	Adapts to underwater dynamics, improves PDR, delay, and lifetime compared to DQELR	High computational cost and tuning complexity; unsuitable for low-resource underwater nodes
Basma Mostafa et al. [21]	QoS-enhanced RPL with edge-assisted routing and conflict-free, Pareto-based path selection	Strong scalability, better QoS compliance, supports real-time and dynamic reconfiguration	No direct benchmarking with many protocols yet; deployment in hybrid networks remains open
Ashapu Bhavani et al. [22]	Multi-objective GASO algorithm for QoS-aware routing with energy and trust optimization	Reduces delay and energy use, resists routing attacks, supports scalability	Complex hybrid model, weak Trust Management, hardware deployment barriers remain

3. System Model and Assumptions

We consider a distributed WSN with N static sensor nodes uniformly and independently deployed over a 2D region. Each node has limited compute and memory, a short-range half-duplex IEEE 802.15.4 [5] radio with maximum range $R_{\max}$, and a finite initial energy budget E_{init} . Nodes communicate via multi-hop forwarding to a single sink by default (extension to multiple sinks is straightforward and does not alter the algorithms). Time is slotted: at each slot, the nodes sense enqueue data, and may forward packets toward the sink [23].

Every node acts as an RL agent that selects its next hop using local state features such as residual energy, distance to the sink, queue/load, and a path security index. Neighbor discovery is performed by lightweight HELLO beacons broadcast every T_H with random jitter; links are retained only if their smoothed reliability exceeds a minimum threshold $L_{\min}$. Energy accounting follows the first-order radio model, where transmission, reception, and processing costs directly feed the battery recursion used throughout our simulations and in the energy results [24].

Adversarial conditions are included to stress the routing/learning stack: per round, a subset of paths is corrupted by DoS [25], spoofing, and selective-forwarding effects; we use synthetic traces and live captures to drive loss and jitter anomalies that the trust module converts into behavioral-trust-based updates.

Figure 2 illustrates that each node trains a local Q-table; only the model summaries are sent to the sink for federated aggregation. A binary HGS stage shapes Cluster Heads and parent links to warm-start RL, accelerating convergence and improving energy balance. The anomaly detector continuously updates trust scores, which constrain admissible actions. The overall objective is multi-criteria: minimize energy and delay while maximizing path security.

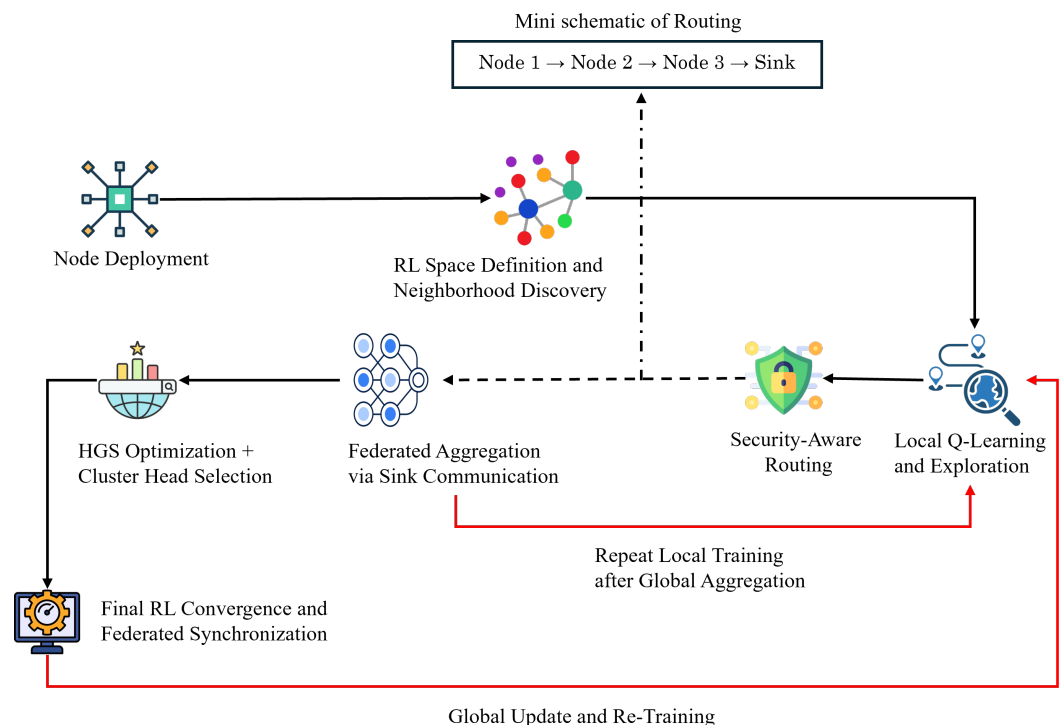


Figure 2. System model of the proposed AFRL-HGS framework.

3.1. Energy and Communication Model

To ensure consistency between the theoretical formulation, the proposed AFRL-HGS method, and the reported results (Figure 3 and performance plots), we adopt the stan-

standard first-order radio model to characterize the energy consumption of sensor nodes in WSNs [26].

Each node spends energy when transmitting, receiving, or locally processing packets. The energy required to transmit a k -bit packet over a distance d is:

$$E_{tx}(k, d) = \begin{cases} k \cdot E_{elec} + k \cdot \epsilon_{fs} \cdot d^2, & d < d_0, \\ k \cdot E_{elec} + k \cdot \epsilon_{mp} \cdot d^4, & d \geq d_0, \end{cases} \quad (1)$$

while the energy required to receive a k -bit packet is:

$$E_{rx}(k) = k \cdot E_{elec}. \quad (2)$$

Here:

- E_{elec} denotes the per-bit circuitry cost for transmitting or receiving.
- ϵ_{fs} and ϵ_{mp} are the amplifier parameters for the free-space (d^2) and multipath (d^4) channel models, respectively.
- The threshold distance is $d_0 = \sqrt{\frac{\epsilon_{fs}}{\epsilon_{mp}}}$.

In addition, each node incurs a computational cost for local reinforcement learning updates and anomaly detection. Thus, the residual energy of node i at time $t + 1$ is updated as:

$$E_i(t + 1) = E_i(t) - \left(E_{tx}^i(k, d) + E_{rx}^i(k) + E_{comp}^i \right), \quad (3)$$

where E_{comp}^i represents per-packet processing overheads such as policy evaluation, Q-learning updates, and trust computation. This term ensures that learning itself is energy-accounted, directly linking the communication model to the AFRL-HGS design.

Nodes transmit at periodic sensing intervals defined by the sensing rate, ρ . A node becomes inactive if its residual energy falls below the minimum threshold, E_{min} , which directly influences network lifetime.

This model ensures that energy is explicitly coupled with routing and learning. Transmission, reception, and computation costs appear in the reward function, residual energy contributes to federated aggregation, the total energy is a core term of the HGS fitness function, and energy also shapes aggregation weights. Algorithm 1 describes the energy consumption and communication model used in WSNs.

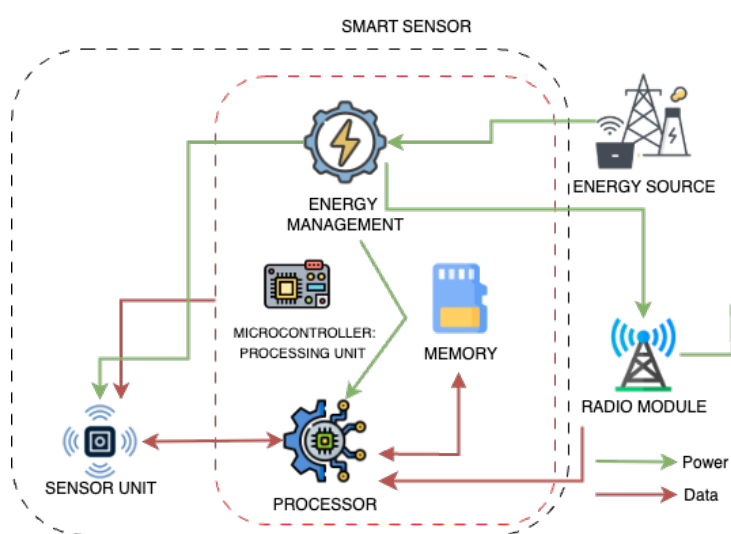


Figure 3. Energy consumption and communication model used in AFRL-HGS. This model underpins the network lifetime, PLR, and energy results.

Algorithm 1 Energy consumption and communication model

Input: Message size k , node position $\vec{x}_i$, neighbor $\vec{x}_j$, current energy $E_i(t)$, radio params $E_{\text{elec}}, \epsilon_{\text{fs}}, \epsilon_{\text{mp}}, d_0$, threshold E_{min}

Output: Updated energy $E_i(t+1)$

Compute distance $d \leftarrow \|\vec{x}_i - \vec{x}_j\|$;

if $d < d_0$ **then**

$E_{\text{tx}} \leftarrow k \cdot E_{\text{elec}} + k \cdot \epsilon_{\text{fs}} \cdot d^2$;

else

$E_{\text{tx}} \leftarrow k \cdot E_{\text{elec}} + k \cdot \epsilon_{\text{mp}} \cdot d^4$;

end

$E_{\text{rx}} \leftarrow k \cdot E_{\text{elec}}$;

Compute processing cost $E_{\text{comp}} \leftarrow$ function of RL/trust updates;

Update energy:

$$E_i(t+1) \leftarrow E_i(t) - (E_{\text{tx}} + E_{\text{rx}} + E_{\text{comp}})$$

if $E_i(t+1) < E_{\text{min}}$ **then**

 Node i is marked as **inactive**;

Node i remains **active**;

3.2. Routing Tables in WSN

In WSNs, routing tables are essential for ensuring efficient, reliable, and energy-aware communication, which becomes even more critical in large-scale or dynamic networks [27]. To support the AFRL-HGS framework, two routing tables are maintained at each node: `RoutingTableUP` for upward communication toward the sink, and `RoutingTableDOWN` for downward control and advertisement.

Routing Table Up: `RoutingTableUP` is used for transmitting sensed data and control packets toward the sink. Parent selection is based on two criteria: (i) minimum hop count, and (ii) maximum recorded Signal-to-Noise Ratio (SNR). This dual criterion reduces both energy consumption and packet loss. Parent nodes periodically broadcast HELLO messages containing hop count and link quality; child nodes update `RoutingTableUP` accordingly. Each entry includes the following:

- `dev, dst, m_seqNo, m_iface, m_hops, nextHop, m_lifeTime, m_minSnr`.

RoutingTableDOWN: `RoutingTableDOWN` supports downward communication from sink to nodes. It is populated from sink-initiated advertisement messages, which are relayed and extended hop-by-hop. Unlike the UP table, the DOWN direction relies only on hop count, since the sink lacks direct SNR information about child links. Thus, `m_minSnr` is omitted. Each entry includes the following:

- `dev, dst, m_seqNo, m_iface, m_hops, nextHop, m_lifeTime`.

3.3. HELLO Messages: Structure, Functionality, and Generation in WSNs

HELLO messages are essential for neighbor discovery and route maintenance in WSN routing protocols [28]. To reduce energy overhead, an adaptive broadcast strategy is used: only sink nodes and those with at least one valid upward route transmit HELLO messages. Each HELLO message carries routing metadata including hop count, residual energy, and minimum SNR of the forwarding link. These fields are directly consumed by the following:

- **RoutingTableUP:** updated with parent candidates based on hop count and SNR.
- **RL state space:** residual energy and distance metrics are discretized into state features (Step 3).

- **Trust evaluation:** anomaly and reliability scores are initialized using the reported SNR values (Step 4).

Because HELLO messages are lightweight, they impose minimal energy cost on low-power nodes. The adaptive broadcast interval and hop-limit policy ensure scalability and efficient use of bandwidth, which contributes directly to the observed improvements in Packet Delivery Ratio, Packet Loss Ratio, and End-to-End Delay reported. Algorithm 2 presents the routing table management mechanism for the WSN.

Algorithm 2 Routing table management in WSN

Input: Incoming messages: HELLO or Advertisement
Output: Updated RoutingTableUP and RoutingTableDOWN
Initialize tables: RoutingTableUP $\leftarrow \emptyset$, RoutingTableDOWN $\leftarrow \emptyset$;
foreach HELLO message from n_j **do**
 Update or insert entry $\langle \text{dev, dst, m_seqNo, m_iface, m_hops, nextHop, m_lifeTime, m_minSnr} \rangle$ in RoutingTableUP.
end
CHOOSEBESTPARENT Select parent maximizing

$$M(r) = \alpha \cdot \frac{1}{r.m_hops} + \beta \cdot r.m_minSnr.$$

foreach Advertisement from sink or parent **do**
 Update or insert entry $\langle \text{dev, dst, m_seqNo, m_iface, m_hops, nextHop, m_lifeTime} \rangle$ in RoutingTableDOWN.
end
FORWARDPACKETpacket, direction **if** direction == “UP” **then**
 Use CHOOSEBESTPARENT().
end
else
 Use matching entry in RoutingTableDOWN.
end
Send packet.

3.4. HELLO Messages: Detailed Format and Role in WSN Routing

Figure 4 shows two formats for HELLO messages in WSNs. One is a lightweight version for sensor nodes with limited energy (Figure 4b). The other is a detailed version used by CHs or sink nodes (see Figure 4a).

The detailed HELLO format (Figure 4a) includes the following fields:

- RecordType (1 byte): Identifies the message as a HELLO message.
- RecordVersion (2 bytes): Indicates the protocol version.
- RecordLength (2 bytes): Total length of the message.
- HandshakeType (1 byte): Shows the operation type, such as update or discovery.
- MessageLength (3 bytes): Size of the message payload.
- MessageVersion (2 bytes): Version number for the HELLO message.
- Random (32 bytes): Contains a nonce to prevent replay attacks.
- SessionIDLength (1 byte): Length of the session ID.
- SessionID: Unique session identifier (variable length).
- CipherSuitesLength (2 bytes): Number of supported encryption algorithms.
- CipherSuites (variable): List of supported encryption methods.
- CompressionMethodsLength (1 byte): Number of supported compression methods.
- CompressionMethods (variable): List of compression methods.
- ExtensionsLength (2 bytes): Total length of the extension section.
- Extensions (variable): Modular extension blocks, each containing:

- ExtensionType (2 bytes): Type of extension, such as residual energy or SNR threshold.
- ExtensionLength (2 bytes): Size of the extension data.
- ExtensionData (variable): Actual routing or network metric data.

Lightweight HELLO Format: (Figure 4b) This format includes only the essential information for routing, leaving out optional fields like encryption, compression, and extensions. It is designed for simple sensor nodes with limited processing power and memory.

Generation Mechanism: The variable `m_helloTimerExpire` is a timer that triggers the creation of HELLO messages at regular intervals. Sink nodes always broadcast HELLO messages, but intermediate nodes send them only if they have a valid upward route. When generating a HELLO message, the node includes the following:

- HopCount: The smallest known number of hops to the sink.
- MinSNR: The minimum Signal-to-Noise Ratio for the best route.
- OriginatorAddress: Address of the node that created the message.
- SequenceNumber: Incremented to prevent routing loops.

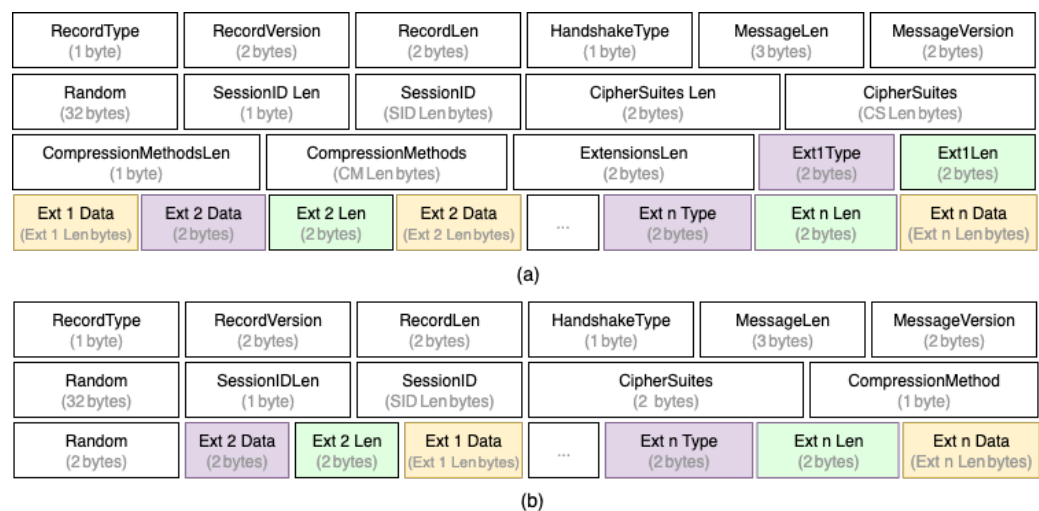


Figure 4. HELLO message formats in WSNs: (a) detailed structure for cs/CHs sinks; (b) lightweight structure for simple nodes.

In more advanced WSN setups, extra fields like SessionID and Extensions can be added. These support session tracking and enable multi-metric routing.

The dual-format design of HELLO messages lets WSNs work well in different environments. Regular sensor nodes use simple HELLO formats to save energy. CHs and sink nodes use more detailed messages for better coordination across the network [28]. This modular approach helps make WSNs more scalable, efficient, and flexible in many deployment scenarios.

3.5. HELLO Message Generation in WSNs

Regular HELLO message generation is important for keeping routing information up to date in WSNs. This process is managed by the `m_helloTimerExpire` parameter and runs at set intervals, as shown in Figure 5. The steps in the generation process are as follows:

1. **Node Role Identification:** Sink nodes act as the root of the routing hierarchy and always generate HELLO messages. Simple nodes create HELLO messages only if they have at least one valid UP-route entry in their `RoutingTableUP`.

2. **Field Initialization:** Sink nodes directly set fields like HopCount, MinSNR, OriginatorAddress, and SequenceNumber. Simple nodes fill in these fields based on their best route in RoutingTableUP.
3. **Broadcasting:** After building the HELLO message, the node broadcasts it to neighbors. Each recipient checks if the message offers a better route to the sink and updates its routing table if needed.

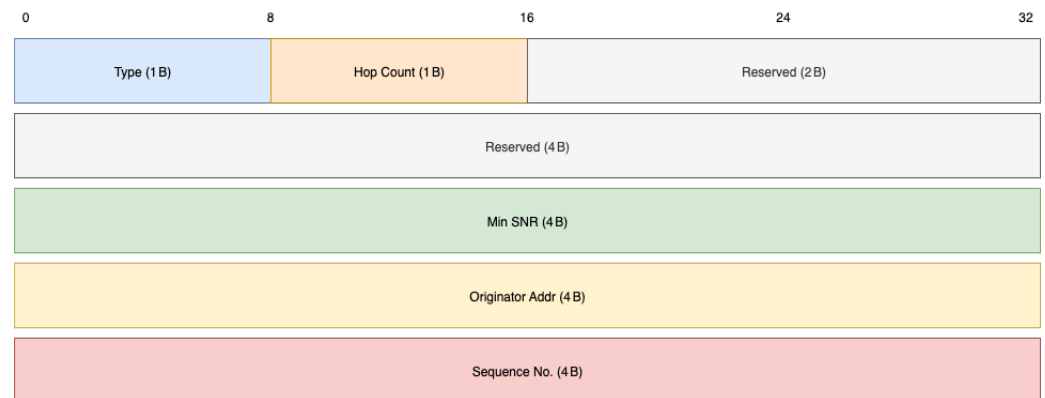


Figure 5. Structure of a HELLO message in WSN.

3.6. Advertising Messages in WSNs

To maintain routing consistency and scalability, the protocol relies on two types of advertisement messages: ADVERTISE and SrvADVERTISE. These messages ensure that both intra-cluster and inter-cluster routing information is regularly refreshed and distributed across the network [29].

3.6.1. ADVERTISE Messages

Simple nodes generate ADVERTISE messages to inform their parent (Cluster Head or sink) of their presence and availability within the cluster. These messages primarily support the population of downward routes in RoutingTableDOWN. Algorithm 3 describes the processing of ADVERTISE messages. A typical ADVERTISE message includes the following:

- HopCount: Number of hops required to reach the originator.
- OriginatorAddress: Unique address of the node that generated the message.
- SequenceNumber: Incremental identifier to avoid duplication and ensure freshness.

Processing ADVERTISE Messages:

Algorithm 3 ADVERTISE message processing

Input: Received ADVERTISE message

Output: Updated RoutingTableDOWN

Extract HopCount, OriginatorAddress, and SequenceNumber **if** message offers a shorter or fresher path **then**

 Update or insert entry in RoutingTableDOWN;

end

if node is a sink **then**

 Transform ADVERTISE into an SrvADVERTISE by adding sink address

 Forward to server;

end

3.6.2. SrvADVERTISE Messages

SrvADVERTISE messages extend advertisement to the inter-cluster domain. These are created by sink nodes to advertise their own addresses upward to the server, enabling communication across multiple clusters. Algorithm 4 presents the SrvADVERTISE message processing procedure. Each SrvADVERTISE message contains the following:

- HopCount: Distance from the originating node to the sink.
- OriginatorAddress: Address of the originating node.
- SinkAddress: Address of the sink initiating the message.
- SequenceNumber: Unique identifier for freshness and loop prevention.

Processing SrvADVERTISE Messages:

Algorithm 4 SrvADVERTISE message processing

Input: Received SrvADVERTISE message

Output: Updated RoutingTableDOWN

Extract HopCount, OriginatorAddress, SinkAddress, and SequenceNumber **if**
message improves or refreshes the route then

 | Update or insert entry in RoutingTableDOWN;

end

3.6.3. Advertising Message Generation

Both ADVERTISE and SrvADVERTISE messages are generated periodically to ensure fresh routing information. Algorithm 5 illustrates the generation of advertising messages. Their generation depends on node roles:

Algorithm 5 Advertising message generation

Input: Node role, entries in RoutingTableDOWN

Output: Generated ADVERTISE or SrvADVERTISE messages

if *node is a simple node then*

 | Build ADVERTISE with HopCount, OriginatorAddress, SequenceNumber;

 | Send to parent node (CH or sink);

end

else if *node is a sink then*

 | Periodically generate SrvADVERTISE with HopCount, SinkAddress,
 SequenceNumber;

 | Forward received ADVERTISE messages as SrvADVERTISE to the server;

end

3.7. Proposed Method

Step 1: Initial Modeling of Nodes, Network Topology, and Physical Resources.

We consider a static WSN deployed in a rectangular region $\Omega \triangleq [0, L_x] \times [0, L_y] \subset \mathbb{R}^2$ with a single sink at $\vec{x}_{\text{sink}} \in \Omega$ (extension to multiple sinks $\mathcal{S} = \{s_1, \dots, s_{|\mathcal{S}|}\}$ is straightforward). Nodes are placed i.i.d. uniformly at random in Ω and have limited battery, compute, and memory resources.

Node positions and initial energy. Let N be the number of sensor nodes. The position of node i is $\vec{x}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \in \Omega$, and all nodes start with the same initial energy:

$$\forall i \in \{1, \dots, N\} : \quad \vec{x}_i \sim \mathcal{U}(\Omega), \quad E_i(0) = E_{\text{init}}. \quad (4)$$

We assume ideal localization (each node knows $\vec{x}_i$ and the sink broadcasts $\vec{x}_{\text{sink}}$); if localization noise is present, it can be modeled as $\tilde{\vec{x}}_i = \vec{x}_i + \epsilon_i$ with $\epsilon_i \sim \mathcal{N}(\mathbf{0}, \sigma_{\text{loc}}^2 \mathbf{I}_2)$.

Distance and neighborhood. The Euclidean distance between nodes i and j is

$$d_{i,j} \triangleq \|\vec{x}_i - \vec{x}_j\|_2, \quad d_{i,\text{sink}} \triangleq \|\vec{x}_i - \vec{x}_{\text{sink}}\|_2. \quad (5)$$

A directed link $i \rightarrow j$ is feasible if $d_{i,j} \leq R_{\text{max}}$, where R_{max} is the PHY-layer communication range.

Radio/energy model. We adopt the first-order radio model with a free-space/multipath crossover distance d_0 . Let k be the packet size in bits, E_{elec} the electronics energy per bit (TX/RX), ϵ_{fs} and ϵ_{mp} the amplifier coefficients for free-space ($\eta = 2$) and multipath ($\eta = 4$) propagation, respectively. The per-packet transmit and receive energies are:

$$E^{\text{tx}}(k, d) = \begin{cases} E_{\text{elec}} k + \epsilon_{\text{fs}} d^2 k, & d < d_0, \\ E_{\text{elec}} k + \epsilon_{\text{mp}} d^4 k, & d \geq d_0, \end{cases} \quad (6)$$

$$E^{\text{rx}}(k) = E_{\text{elec}} k, \quad (7)$$

$$d_0 \triangleq \sqrt{\frac{\epsilon_{\text{fs}}}{\epsilon_{\text{mp}}}}. \quad (8)$$

If on-node processing is non-negligible, we include a per-bit CPU cost E_{proc} so that the processing energy for k bits is $E^{\text{proc}}(k) = E_{\text{proc}} k$.

Battery dynamics. Let $\mathcal{T}_i(t)$ and $\mathcal{R}_i(t)$ denote the sets of packets transmitted to and received by node i during slot/epoch t , and let $d_{i,\cdot}$ be the corresponding TX distances. The battery recursion is

$$E_i(t+1) = E_i(t) - \sum_{p \in \mathcal{T}_i(t)} \left(E^{\text{tx}}(k_p, d_{i,\text{dst}(p)}) + E^{\text{proc}}(k_p) \right) - \sum_{p \in \mathcal{R}_i(t)} \left(E^{\text{rx}}(k_p) + E^{\text{proc}}(k_p) \right), \quad (9)$$

with node i considered dead when $E_i(t) \leq 0$. Control traffic (e.g., HELLO/ACK, FRL uploads) is accounted for by using (6) and (7) with the appropriate control packet sizes.

Step 2: Neighbor Discovery and Initialization of Communication Metrics.

Each node discovers neighbors and initializes link metrics (distance, reliability, trust primitives) using a lightweight HELLO/ACK procedure over IEEE 802.15.4 Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA).

MAC/PHY and timing. Time is partitioned in discovery periods of length T_H . To reduce collisions, each node transmits one HELLO per period at time $t_i^{\text{HELLO}} = t_0 + u_i$ where $u_i \sim \mathcal{U}[0, J_H]$ is a jitter ($0 < J_H < T_H$). We use unslotted CSMA/CA with maximum re-transmissions r_{max} for data/ACK frames. Let k_H and k_A be the HELLO and ACK payload sizes (bits).

Control frame formats. Node i broadcasts $\text{HELLO}_i = \langle ID_i, \vec{x}_i, E_i, ts_i \rangle$ (ts_i is a local timestamp). Upon reception, neighbor j unicasts $\text{ACK}_{j \rightarrow i} = \langle ID_j, \vec{x}_j, E_j, ts_j \rangle$.

Neighbor set and distances. The (directed) neighbor set of i at period t is $\mathcal{N}_i(t) = \{j \neq i \mid \text{ACK received from } j \text{ in period } t\}$. Distances follow Step 1: $d_{i,j}(t) = \|\vec{x}_i - \vec{x}_j\|_2$, $d_{i,\text{sink}}(t) = \|\vec{x}_i - \vec{x}_{\text{sink}}\|_2$.

Link reliability estimator. Let $z_{i,j}(t) \in \{0, 1\}$ indicate whether a valid ACK from j to i was received in period t . We maintain both a sliding-window estimator over the last W_L periods and an EWMA:

$$L_{i,j}^{\text{win}}(t) = \frac{\sum_{\tau=t-W_L+1}^t z_{i,j}(\tau)}{W_L}, \quad (10)$$

$$\hat{L}_{i,j}(t) = (1 - \lambda_L) \hat{L}_{i,j}(t-1) + \lambda_L z_{i,j}(t), \quad \hat{L}_{i,j}(0) = L_{i,j}^{\text{win}}(0), \quad (11)$$

with $\lambda_L \in (0, 1]$. A link is considered usable if $\hat{L}_{i,j}(t) \geq L_{\min}$. Neighbors that fail to respond for W_{miss} consecutive periods are pruned.

RTT and RSSI sampling. For each ACK, node i estimates Round-Trip Time $\text{RTT}_{i,j}(t) = t_{\text{rx}}(\text{ACK}_{j \rightarrow i}) - t_{s_i}$, and records receiver-side RSSI (from the radio), yielding time series $\{\text{RTT}_{i,j}(t)\}$ and $\{\text{RSSI}_{i,j}(t)\}$.

Lightweight anomaly detector (for trust primitives). Per link (i, j) , define residuals for a generic scalar metric $X \in \{\text{RTT}, \text{loss}\}$, where $\text{loss}_{i,j}(t) = 1 - z_{i,j}(t)$. Maintain EWMA mean/variance:

$$m_X(t) = (1 - \beta) m_X(t-1) + \beta X(t), \quad (12)$$

$$v_X(t) = (1 - \beta) v_X(t-1) + \beta (X(t) - m_X(t-1))^2, \quad (13)$$

with $\beta \in (0, 1]$. The standardized residual is $r_X(t) = \frac{X(t) - m_X(t-1)}{\sqrt{v_X(t-1) + \epsilon}}$ ($\epsilon > 0$ small). We flag an anomaly on (i, j) at time t if either $|r_{\text{RTT}}(t)| > \kappa$ or the windowed loss $\overline{\text{loss}}_{i,j}(t) = 1 - L_{i,j}^{\text{win}}(t)$ exceeds τ_{loss} , for at least M consecutive periods. The Boolean flag $\text{anomaly}_{i,j}(t) \in \{0, 1\}$ is passed to Step 4's trust update.

Control plane energy accounting. Let $p_{i \rightarrow j}(t)$ denote the one-shot success probability for $i \rightarrow j$ transmissions in period t (estimated by $\hat{L}_{i,j}(t)$ or PHY metrics). The expected number of transmissions (including CSMA/CA retries, truncated at r_{max}) is $\phi_{i \rightarrow j}(t) = \mathbb{E}[\text{\#TX}] = \sum_{r=1}^{r_{\text{max}}} r p_{i \rightarrow j}(t) (1 - p_{i \rightarrow j}(t))^{r-1} + r_{\text{max}} (1 - p_{i \rightarrow j}(t))^{r_{\text{max}}}$. Assuming a fixed broadcast range d_i^{bc} for HELLOs, the expected control energy of node i per discovery period is

$$E_i^{\text{ctrl}}(t) = E^{\text{tx}}(k_H, d_i^{\text{bc}}) + \sum_{j \in \mathcal{N}_i(t)} E^{\text{rx}}(k_A) + \sum_{j \in \mathcal{N}_i(t)} \phi_{i \rightarrow j}(t) E^{\text{tx}}(k_A, d_{i,j}(t)), \quad (14)$$

which is debited in the battery recursion (Equation (9)).

Initialization of trust primitives. Behavioral trust is initialized neutrally: $T_{i,j}^{\text{behav}}(0) = T_0 \in (0, 1)$ ($T_0 = 0.5$) and will be updated in Step 4 using $\text{anomaly}_{i,j}(t)$, $\hat{L}_{i,j}(t)$, and RTT residuals. Algorithm 6 presents the neighbor discovery and metric initialization procedure.

Step 3: Defining the Reinforcement Learning Space for Network Nodes (Modeling RL Agents).

Each node n_i is modeled as an RL agent that selects its next-hop parent based on local state features. To ensure tractability, continuous state variables are discretized into finite bins, enabling tabular Q-learning.

State space. The state of agent i at time t is represented as:

$$s_t^i = [\text{bin}_E(E_i(t)), \text{bin}_d(d_{i,\text{sink}}), \text{bin}_\sigma(\sigma_i(t)), \text{bin}_q(q_i(t)), \text{bin}_P(P_{i \rightarrow \text{sink}}^{\text{sec}}(t))]. \quad (15)$$

- $E_i(t)$: residual energy of node i , discretized into B_E bins (e.g., 10 uniform bins between 0 and E_{init}).
- $d_{i,\text{sink}}$: distance to sink, discretized into concentric rings $[0, R_1), [R_1, R_2), \dots, [R_{B_d-1}, \infty)$.
- $\sigma_i(t)$: queue congestion index, discretized into B_σ levels (e.g., $\{\text{empty}, \text{low}, \text{medium}, \text{high}\}$).
- $q_i(t)$: path quality score (delay/PDR composite), mapped to B_q bins using quantiles or thresholds.
- $P_{i \rightarrow \text{sink}}^{\text{sec}}(t)$: estimated end-to-end security probability from i to the sink, discretized into B_P intervals in $[0, 1]$.

Thus, the effective state space is finite with cardinality $|S_i| = B_E \cdot B_d \cdot B_\sigma \cdot B_q \cdot B_P$.

Action space. At each step, node i selects one neighbor as the next-hop parent:

$$A_i(t) = \{a_j \mid j \in \mathcal{N}_i(t), E_j(t) > E_{\min}\}. \quad (16)$$

Here, $\mathcal{N}_i(t)$ is the neighbor set from Step 2 and $E_{\min}$ is the minimum energy threshold for participation.

Reward function. The reward encourages low-energy, low-delay, and secure routing. To avoid mixing different physical units, each term is normalized:

$$r_t^i = -\alpha \cdot \frac{E_i^{\text{tx}}(t)}{E_{\max}^{\text{tx}}} - \beta \cdot \frac{D_t^i}{D_{\max}} + \gamma \cdot \frac{P_{i \rightarrow \text{sink}}^{\text{sec}}(t) - \theta_{\text{sec}}}{1 - \theta_{\text{sec}}}, \quad (17)$$

where:

- $E_i^{\text{tx}}(t)$: transmit energy consumed by node i in epoch t , normalized by a maximum reference $E_{\max}^{\text{tx}}$.
- D_t^i : End-to-End Delay from node i to sink in epoch t , normalized by a maximum tolerable delay $D_{\max}$.
- $P_{i \rightarrow \text{sink}}^{\text{sec}}(t)$: path-level security probability, computed as in Step 4, normalized relative to the threshold θ_{sec} .

The weights $\alpha, \beta, \gamma \geq 0$ satisfy $\alpha + \beta + \gamma = 1$, balancing the trade-offs among energy, delay, and security.

Algorithm 6 Neighbor discovery and metric initialization (per period of length T_H)

Input: Node set $\mathcal{V} = \{1, \dots, N\}$, positions $\vec{x}_i$, initial E_i , params

$T_H, J_H, r_{\max}, W_L, \lambda_L, \beta, \kappa, \tau_{\text{loss}}, M, L_{\min}$

Output: $\mathcal{N}_i(t), d_{i,j}(t), \hat{L}_{i,j}(t), \text{RTT}_{i,j}(t), \text{anomaly}_{i,j}(t), E_i^{\text{ctrl}}(t)$

foreach node $i \in \mathcal{V}$ **do**

 Schedule HELLO at $t_i^{\text{HELLO}} = t_0 + u_i, u_i \sim \mathcal{U}[0, J_H]$ // Broadcast HELLO

 with CSMA/CA; truncated retries

 Transmit $\text{HELLO}_i = \langle ID_i, \vec{x}_i, E_i, ts_i \rangle$; accrue $E^{\text{tx}}(k_H, d_i^{\text{bc}})$;

end

foreach node j receiving HELLO_i **do**

 Measure RSSI; timestamp ts_j Unicast $\text{ACK}_{j \rightarrow i}$ with CSMA/CA (up to $r_{\max}$);

 accrue $\phi_{j \rightarrow i} E^{\text{tx}}(k_A, d_{j,i})$;

end

foreach node i **do**

foreach ACK from j **do**

 Update neighbor set $\mathcal{N}_i(t) \leftarrow \mathcal{N}_i(t) \cup \{j\}$ $d_{i,j}(t) \leftarrow \|\vec{x}_i - \vec{x}_j\|_2$ $z_{i,j}(t) \leftarrow 1$;

 update $L_{i,j}^{\text{win}}, \hat{L}_{i,j}$ $\text{RTT}_{i,j}(t) \leftarrow t_{\text{rx}} - ts_i$; update m_X, v_X and residuals; set

$\text{anomaly}_{i,j}(t)$;

 Accrue $E^{\text{rx}}(k_A)$;

end

foreach j with no ACK **do**

$z_{i,j}(t) \leftarrow 0$; update $L_{i,j}^{\text{win}}, \hat{L}_{i,j}$; update anomaly logic;

end

 Remove j with $\hat{L}_{i,j}(t) < L_{\min}$ or W_{miss} consecutive misses Compute $E_i^{\text{ctrl}}(t)$ via (14) and subtract in Equation (9);

end

Decision policy. Each agent follows an ε -greedy policy over its Q-table:

$$\pi(a_t | s_t^i) = \begin{cases} \text{random}(a \in A_i(t)), & \text{with probability } \varepsilon, \\ \arg \max_{a \in A_i(t)} Q_i(s_t^i, a), & \text{with probability } 1 - \varepsilon. \end{cases} \quad (18)$$

The exploration rate ε decays over time, ensuring sufficient exploration initially while converging toward exploitation of high-value routes.

Step 4: Security Evaluation and Path Trustworthiness Assessment.

To ensure that the paths chosen in Step 3 remain resilient against misbehavior and channel anomalies, each node maintains a dynamic trust score per link (i, j) .

Hybrid link trust. For each neighbor $j \in \mathcal{N}_i(t)$, node i computes a hybrid link trust:

$$P_{i,j}^{\text{sec}}(t) = \rho T_{i,j}^{\text{hist}}(t) + (1 - \rho) T_{i,j}^{\text{behav}}(t), \quad (19)$$

where:

- $T_{i,j}^{\text{hist}}(t) \in [0, 1]$ is the historical trust, estimated from the smoothed link reliability $\hat{L}_{i,j}(t)$.
- $T_{i,j}^{\text{behav}}(t) \in [0, 1]$ is the behavioral trust, adapted from anomaly detector outputs ($\text{anomaly}_{i,j}(t)$).
- $\rho \in [0, 1]$ balances long-term reliability vs. short-term anomaly evidence.

Behavioral trust update. At each discovery period, behavioral trust is updated as:

$$T_{i,j}^{\text{behav}}(t+1) = \begin{cases} \max(0, T_{i,j}^{\text{behav}}(t) - \delta), & \text{if } \text{anomaly}_{i,j}(t) = 1, \\ \min(1, T_{i,j}^{\text{behav}}(t) + \mu), & \text{otherwise,} \end{cases} \quad (20)$$

with $\delta \in (0, 1]$ the penalty decrement and $\mu \in (0, 1]$ the recovery increment.

Path-level security aggregation. To avoid underflow from multiplicative products, path security is aggregated in the log domain:

$$\log P_{\text{path}}^{\text{sec}}(t) = \sum_{(u,v) \in p} \log P_{u,v}^{\text{sec}}(t), \quad P_{\text{path}}^{\text{sec}}(t) = \exp\left(\frac{1}{|p|} \sum_{(u,v) \in p} \log P_{u,v}^{\text{sec}}(t)\right). \quad (21)$$

This formulation yields the geometric mean of link trust scores, ensuring longer paths are not unfairly penalized while still reflecting weak links.

Constraint vs. reward. To avoid double counting, we enforce a *constraint-only* role for security: a candidate path p is admissible if

$$P_{\text{path}}^{\text{sec}}(t) \geq \theta_{\text{sec}}, \quad \theta_{\text{sec}} \in (0, 1), \quad (22)$$

otherwise, it is excluded from the action set $A_i(t)$ in Step 3. This ensures that the reward need not include a separate security term, keeping objectives decoupled.

Parameter ranges. Typical values are: $\rho \in [0.3, 0.7]$ (balance), $\delta = 0.1\text{--}0.2$ (penalty), $\mu = 0.05\text{--}0.1$ (recovery), and $\theta_{\text{sec}} = 0.6\text{--}0.8$ (acceptance threshold). These can be tuned in the simulation setup. Algorithm 7 describes the process of security evaluation and trust updating.

Step 5: Local Q-Learning Model Training for Nodes.

Each node n_i trains its own Q-learning agent over the discretized state/action space defined in Step 3, subject to the admissibility constraints from Step 4. Learning proceeds in episodes, where each episode corresponds to a sequence of packet transmissions and acknowledgments until either the sink is reached or a timeout occurs.

Q-learning update. At each decision epoch t , node i observes its current state $s_t^i \in S_i$, selects an action $a_t \in A_i(t)$, obtains a normalized reward r_t^i (Equation Step 3), and observes the next state s_{t+1}^i . Its Q-table is updated as:

$$Q_i(s_t^i, a_t) \leftarrow Q_i(s_t^i, a_t) + \eta_t \left[r_t^i + \gamma \max_{a'} Q_i(s_{t+1}^i, a') - Q_i(s_t^i, a_t) \right], \quad (23)$$

where:

- $\eta_t \in (0, 1]$ is the learning rate at epoch t , scheduled to decay as $\eta_t = \eta_0 / (1 + \zeta t)$ with $\eta_0 \in [0.1, 0.5]$, $\zeta > 0$.
- $\gamma \in (0, 1)$ is the discount factor, typically $0.8 \leq \gamma \leq 0.95$.
- r_t^i reflects normalized energy, delay, and path security feasibility (from Steps 1–4).

Algorithm 7 Security evaluation and trust update

Input: Reliability $\hat{L}_{i,j}(t)$, anomaly flags $\text{anomaly}_{i,j}(t)$
Output: Updated link trust $P_{i,j}^{\text{sec}}(t)$, path trust $P_{\text{path}}^{\text{sec}}(t)$

foreach *link* (i, j) **do**
 Compute historical trust: $T_{i,j}^{\text{hist}}(t) \leftarrow \hat{L}_{i,j}(t)$ Update behavioral trust
 $T_{i,j}^{\text{behav}}(t+1)$ using anomaly flag and δ, μ Fuse:
 $P_{i,j}^{\text{sec}}(t) \leftarrow \rho T_{i,j}^{\text{hist}}(t) + (1 - \rho) T_{i,j}^{\text{behav}}(t);$
end
foreach *candidate path* p **do**
 Compute log domain average: $\log P_{\text{path}}^{\text{sec}}(t) \leftarrow \frac{1}{|p|} \sum_{(u,v) \in p} \log P_{u,v}^{\text{sec}}(t);$
 Recover: $P_{\text{path}}^{\text{sec}}(t) \leftarrow \exp(\log P_{\text{path}}^{\text{sec}}(t))$ **if** $P_{\text{path}}^{\text{sec}}(t) < \theta_{\text{sec}}$ **then**
 Exclude p from admissible actions $A_i(t);$
 end
end

Exploration-exploitation policy. Node i selects actions using an ε -greedy rule:

$$\pi(a_t | s_t^i) = \begin{cases} \text{random}(a \in A_i(t)), & \text{with prob. } \varepsilon_t, \\ \arg \max_{a \in A_i(t)} Q_i(s_t^i, a), & \text{with prob. } 1 - \varepsilon_t, \end{cases} \quad (24)$$

where ε_t decays over episodes, e.g., $\varepsilon_t = \max(\varepsilon_{\min}, \varepsilon_0 \cdot e^{-\kappa t})$, with $\varepsilon_0 \in [0.2, 0.3]$, $\varepsilon_{\min} \approx 0.05$.

Stopping criterion and practical convergence. Unlike infinite-horizon Markov Decision Processes (MDPs), WSN agents operate under finite energy budgets and time-varying topologies. Thus, we adopt *empirical convergence*: training stops after $E_{\max}$ episodes or when the Q-values stabilize, i.e., $\|Q_i^{\tau+1} - Q_i^{\tau}\|_{\infty} < \epsilon_Q$ for M consecutive episodes, with $\epsilon_Q \ll 1$. This practical stopping rule avoids reliance on ergodicity assumptions that are unrealistic in battery-limited WSNs.

Q-table structure and memory footprint. Each Q-table has dimension $|S_i| \times |A_i|$, where $|S_i|$ is the product of the discretization bins from Step 3, and $|A_i|$ is the number of admissible neighbors. For example, with $B_E = 10$, $B_d = 5$, $B_{\sigma} = 4$, $B_q = 5$, $B_p = 4$, we obtain $|S_i| = 10 \cdot 5 \cdot 4 \cdot 5 \cdot 4 = 4000$. If the average neighbor set has $|A_i| = 6$, the Q-table has, at most, 24,000 entries. With each Q-value stored as a 4-byte float, the per-node memory requirement is ≈ 96 kB, feasible for constrained sensor platforms with ≥ 128 kB RAM.

Step 6: Adaptive Federated Aggregation of Q-Learning Models.

To align local policies across the WSN without transmitting raw data, we adopt FRL. Each node n_i periodically uploads a compressed summary of its Q-table after local training, and the sink aggregates these contributions into a global Q-model. This reduces energy overhead compared to centralized training, while ensuring robustness across heterogeneous nodes.

Aggregation model. After local training episode τ , node i has a Q-table $Q_i^\tau(s, a)$. The sink aggregates them as:

$$Q_{\text{global}}(s, a) = \sum_{i=1}^N \omega_i^\tau Q_i^\tau(s, a), \quad (25)$$

where $\omega_i^\tau \geq 0$ and $\sum_{i=1}^N \omega_i^\tau = 1$. Thus, nodes with more reliable contributions influence the global model more.

Adaptive weighting. We define ω_i^τ based on both residual energy and local performance:

$$\omega_i^\tau = \lambda_1 \cdot \frac{E_i(\tau)}{\sum_{j=1}^N E_j(\tau)} + \lambda_2 \cdot \frac{\bar{G}_i^\tau}{\sum_{j=1}^N \bar{G}_j^\tau}, \quad (26)$$

where:

- $E_i(\tau)$: residual energy of node i , consistent with the radio/battery model (Step 1).
- $\bar{G}_i^\tau$: average return of node i 's policy, i.e., the mean episodic reward over the last W_G episodes (normalized to $[0, 1]$).
- $\lambda_1, \lambda_2 \geq 0$: trade-off weights with $\lambda_1 + \lambda_2 = 1$.

Temporal process. Federated exchange occurs every T_{fed} local episodes. To reduce communication cost, only nonzero entries or quantized Q-values are transmitted, incurring energy consumption

$$E_i^{\text{FRL}}(\tau) = \phi_{i \rightarrow \text{sink}} E^{\text{tx}}(k_Q, d_{i, \text{sink}}),$$

where k_Q is the payload size (bits), $d_{i, \text{sink}}$ is the distance to sink, and $\phi_{i \rightarrow \text{sink}}$ is the expected re-transmissions.

Local synchronization. Upon receiving the global Q-table, each node blends it with its local table using a Polyak update (to prevent catastrophic forgetting):

$$Q_i^{\tau+1}(s, a) \leftarrow (1 - \xi) Q_i^\tau(s, a) + \xi Q_{\text{global}}(s, a), \quad \xi \in (0, 1). \quad (27)$$

This ensures stability while gradually aligning local policies with the global one.

Stopping condition. Aggregation continues until either (i) the global Q-values stabilize ($\|Q_{\text{global}}^{\tau+1} - Q_{\text{global}}^\tau\|_\infty < \epsilon_{\text{FRL}}$), or (ii) the average number of alive nodes drops below a threshold. Algorithm 8 presents the adaptive federated aggregation of local Q-learning models.

Step 7: Optimization of Clustering and Routing Using Hunger Games Search.

To this end, we apply a binary variant of the HGS algorithm to jointly optimize CH selection and routing paths. The resulting topology minimizes energy consumption, delay, and vulnerability, while ensuring structural feasibility.

Chromosome representation. Each candidate solution is encoded as:

- A binary CH vector $\vec{z} = [z_1, \dots, z_N]$ where $z_i = 1$ if node i is a CH and $z_i = 0$ otherwise.
- A directed binary routing matrix $R \in \{0, 1\}^{N \times N}$, where $R_{ij} = 1$ denotes a parent-child link $i \rightarrow j$.

Feasibility constraints. Each solution $(\vec{z}, R)$ must satisfy:

$$\sum_j R_{ij} = 1, \quad \forall i \text{ with } z_i = 0 \quad (\text{each non-CH has one parent}), \quad (28)$$

$$R_{ij} = 0, \quad \text{if } d_{i,j} > R_{\max} \quad (\text{within CH range}), \quad (29)$$

$$R_{ii} = 0, \quad \forall i \quad (\text{no self-links}), \quad (30)$$

$$R \text{ acyclic}, \quad \text{with at least one path to the sink}, \quad (31)$$

$$\deg(i) \leq \Delta_{\max}, \quad \forall i \quad (\text{max degree constraint}). \quad (32)$$

Infeasible candidates are repaired (reassigning orphan nodes) or penalized in the fitness score.

Algorithm 8 Adaptive federated aggregation of Local Q-Learning Models

Input: Node set $\mathcal{N} = \{1, \dots, N\}$, local Q-tables $Q_i(s, a)$, learning rate η_t , discount factor γ , exploration schedule ε_t , federation interval T_{fed}

Output: Global Q-table $Q_{\text{global}}(s, a)$ distributed to all nodes

for episode $\tau = 1$ **to** T **do**

foreach node $i \in \mathcal{N}$ **do**

 Run local Q-learning update (Step 5) using admissible actions from Step 4;

end

if $\tau \bmod T_{\text{fed}} = 0$ **then**

foreach node i **do**

 Compute mean episodic reward $\bar{G}_i^\tau$ over last W_G episodes Measure

 residual energy $E_i(\tau)$ Compute weight: $\omega_i^\tau = \lambda_1 \frac{E_i(\tau)}{\sum_j E_j(\tau)} + \lambda_2 \frac{\bar{G}_i^\tau}{\sum_j \bar{G}_j^\tau}$;

 Transmit compressed Q-table entries to sink (energy cost added to E_i);

end

 Sink aggregates: $Q_{\text{global}}(s, a) = \sum_i \omega_i^\tau Q_i^\tau(s, a)$ **foreach** node i **do**

 Update local table: $Q_i^{\tau+1}(s, a) \leftarrow (1 - \xi) Q_i^\tau(s, a) + \xi Q_{\text{global}}(s, a)$;

end

end

end

Fitness function. The objective is to minimize:

$$F(\vec{z}, R) = \lambda_1 E_{\text{total}}(\vec{z}, R) + \lambda_2 D_{\text{avg}}(\vec{z}, R) + \lambda_3 (1 - P_{\text{secure}}(\vec{z}, R)), \quad \lambda_1 + \lambda_2 + \lambda_3 = 1. \quad (33)$$

Metrics are defined as:

- E_{total} : sum of transmit and receive energy for one data collection round:

$$E_{\text{total}} = \sum_{(i,j): R_{ij}=1} (E^{\text{tx}}(k, d_{i,j}) + E^{\text{rx}}(k)).$$

- D_{avg} : average End-to-End Delay across all source-sink paths, computed as sum of per-hop delays (including MAC queuing).
- P_{secure} : geometric mean of per-path security probabilities (Step 4):

$$P_{\text{secure}} = \left(\prod_{p \in \mathcal{P}} P_{\text{path}}^{\text{sec}}(p) \right)^{1/|\mathcal{P}|}.$$

HGS update mechanism (binary variant). In the continuous HGS update, candidate solutions evolve via:

$$\hat{z}_i^{t+1} = z_i^t + \psi H_i^t (z_{\text{best}}^t - z_i^t) + \delta \mathcal{N}(0, 1).$$

To preserve binary encoding, we apply a sigmoid mapping and Bernoulli sampling:

$$P(z_i^{t+1} = 1) = \sigma(\hat{z}_i^{t+1}) = \frac{1}{1 + \exp(-\hat{z}_i^{t+1})}, \quad (34)$$

then sample $z_i^{t+1} \sim \text{Bernoulli}(\sigma(\hat{z}_i^{t+1}))$. This ensures binary Cluster Head assignments.

The hunger factor is computed as:

$$H_i^t = \frac{1}{1 + \exp(-h_i^t)},$$

where h_i^t reflects the fitness-induced hunger of solution i .

Step 8: Integrating HGS Output with Reinforcement Learning for Faster Convergence and Structured Stability.

The binary HGS optimization yields an initial Cluster Head assignment $\bar{z}_{\text{opt}}$ and a feasible routing matrix R_{opt} . These outputs are now used to warm-start the Q-learning agents of Step 5, ensuring that training begins from a near-optimal topology rather than from a random policy. This accelerates convergence and improves stability in the early phases of federated aggregation.

Inputs. The initialization procedure uses:

- $\bar{z}_{\text{opt}} \in \{0, 1\}^N$: optimized CH selection vector.
- $R_{\text{opt}} \in \{0, 1\}^{N \times N}$: optimized directed routing matrix.
- $\bar{E}(t)$: residual energy of all nodes (Step 1).
- $P_i^{\text{sec}}(t)$: per-node security index (Step 4).

Q-table initialization. For each node i , the Q-table entries corresponding to admissible actions $A_i(t)$ are initialized as:

$$Q_i(s_0, a_j) = \begin{cases} Q_{\text{init}}^{\text{high}}, & \text{if } R_{\text{opt}}[i, j] = 1, \\ Q_{\text{init}}^{\text{low}}, & \text{if } R_{\text{opt}}[i, j] = 0, \end{cases} \quad (35)$$

where $Q_{\text{init}}^{\text{high}} > Q_{\text{init}}^{\text{low}} \geq 0$ are bounded constants (e.g., $Q_{\text{init}}^{\text{high}} = 1.0$, $Q_{\text{init}}^{\text{low}} = 0.1$). This ensures that routes favored by HGS start with higher expected utility.

Policy initialization. The initial decision policy is biased toward the HGS solution:

$$\pi_0(a \mid s_0) = \begin{cases} 1 - \varepsilon_0, & \text{if } a = \arg \max_{a' \in A_i} Q_i(s_0, a'), \\ \varepsilon_0 / |A_i|, & \text{otherwise,} \end{cases} \quad (36)$$

where $\varepsilon_0 \in [0.1, 0.3]$ allows limited exploration from the start. This avoids purely deterministic choices and ensures that suboptimal HGS assignments can still be corrected.

Effect on convergence. With HGS-guided initialization, the number of training episodes needed to reach a near-optimal policy is reduced. There exists $\tau_0 < \tau_{\text{random}}$ such that:

$$\|Q_i^{\tau_0} - Q_i^*\|_{\infty} < \varepsilon, \quad (37)$$

meaning that the hybrid HGS-RL process attains an ε -close approximation of the optimal Q-function faster than a random-start Q-learning agent. This warm-start effect improves

energy efficiency by reducing the number of early, exploratory transmissions that would otherwise consume scarce battery resources.

Step 9: Adaptive Re-Aggregation of Learning Policies after HGS-RL Convergence.

After nodes complete their local HGS-guided Q-learning episodes, a second federated aggregation is performed to refine and synchronize policies. This re-aggregation stage ensures that energy-depleted nodes do not dominate decisions, while reinforcing secure and reliable routes across the network.

Inputs. Each node n_i provides:

- Its locally updated Q-table $Q_i^{\text{HGS-RL}}(s, a)$, which reflects both RL training and HGS initialization.
- Residual energy $E_i(t)$.
- Reliability score R_i defined as the Packet Delivery Ratio (PDR) over the last W_R transmissions.
- Security index P_i^{sec} .

Adaptive weighting. The global aggregation weights are computed as:

$$\omega_i = \lambda_1 \cdot \frac{E_i(t)}{\sum_j E_j(t)} + \lambda_2 \cdot \frac{R_i}{\sum_j R_j} + \lambda_3 \cdot \frac{P_i^{\text{sec}}}{\sum_j P_j^{\text{sec}}}, \quad \lambda_1 + \lambda_2 + \lambda_3 = 1, \quad (38)$$

with $\sum_i \omega_i = 1$. This ensures balanced contributions from energy-rich, reliable, and secure nodes.

Global aggregation. The sink computes the new federated Q-function:

$$Q_{\text{global}}(s, a) = \sum_{i=1}^N \omega_i Q_i^{\text{HGS-RL}}(s, a). \quad (39)$$

Local synchronization. Instead of overwriting local tables, each node blends the global update into its own Q-table:

$$Q_i^{\text{next}}(s, a) \leftarrow (1 - \xi) Q_i^{\text{HGS-RL}}(s, a) + \xi Q_{\text{global}}(s, a), \quad \xi \in (0, 1). \quad (40)$$

This prevents catastrophic forgetting and ensures smoother convergence. Algorithm 9 presents the adaptive re-aggregation of HGS-RL Q-tables.

Algorithm 9 Adaptive re-aggregation of HGS-RL Q-tables

Input: Local Q-tables $Q_i^{\text{HGS-RL}}(s, a)$, residual energies $E_i(t)$, reliability scores R_i , security indices P_i^{sec}

Output: Updated local Q-tables $Q_i^{\text{next}}(s, a)$

foreach node i **do**

 Transmit compressed $Q_i^{\text{HGS-RL}}(s, a)$ to sink;

end

Sink computes normalized weights ω_i using $E_i, R_i, P_i^{\text{sec}}$ Compute global

Q-function: $Q_{\text{global}}(s, a) = \sum_i \omega_i Q_i^{\text{HGS-RL}}(s, a)$ **foreach** active node $i \in N_{\text{alive}}$ **do**

 Update local Q-table: $Q_i^{\text{next}}(s, a) \leftarrow (1 - \xi) Q_i^{\text{HGS-RL}}(s, a) + \xi Q_{\text{global}}(s, a)$;

end

4. Dataset and Simulation Setup

The packet source is chosen uniformly at random along the deployment boundary for each simulation run. We compare the proposed **AFRL-HGS** framework against RFO [14], TCHS [30], BHA-ACO [17], and DQN-EQSR [20].

A hybrid environment is employed:

- **GNS3:** simulates multi-hop IoT/WSN topology with IEEE 802.15.4 MAC, Q-learning agents, federated aggregation, and HGS optimization.
- **Kali Linux (v2023.1):** connected via bridged virtual interfaces to inject live adversarial traffic (DoS, spoofing, sinkhole). Captured traces are replayed in GNS3 as packet loss/delay patterns, ensuring realistic perturbations.

Synthetic IoT topology generation: The simulation setup in GNS3 is summarized as Table 3.

Table 3. IoT topology configuration.

Parameter	Value
Deployment area	$100 \times 100 \text{ m}^2$
Nodes	100
Placement	Uniform random
Tx range $R_{\max}$	25 m
Initial energy E_{init}	2 J
Packet size	256 B (2048 bits)
Data generation	0.2 pkt/s
Control sizes	HELLO: 256 bits; ACK: 128 bits
HELLO interval T_H	5 s (± 1 s jitter)
Rounds simulated	1000

Radio/Energy model: We adopt the first-order radio model from Step 1:

$$E_{\text{tx}}(k, d) = E_{\text{elec}} k + \epsilon(d) k d^\eta, \quad E_{\text{rx}}(k) = E_{\text{elec}} k \quad (41)$$

with $E_{\text{elec}} = 50 \text{ nJ/bit}$, $E_{\text{proc}} = 5 \text{ nJ/bit}$, $\epsilon_{fs} = 10 \text{ pJ/bit/m}^2$, $\epsilon_{mp} = 0.0013 \text{ pJ/bit/m}^4$, and path-loss exponent $\eta \in \{2, 4\}$. The crossover distance is $d_0 = \sqrt{\epsilon_{fs}/\epsilon_{mp}}$.

Synthetic attacks in GNS3: 10% of routing paths are compromised per round with:

- DoS (ACK suppression, re-transmission flooding).
- Selective forwarding (packet drops).
- Sinkhole (Q-value manipulation).
- Spoofing (false advertisements).

Table 4 demonstrates the attack tools and targets used in this paper.

Table 4. Attack Tools and Targets.

Tool	Attack	Layer	Effect in GNS3
hping3	UDP/TCP flooding	Transport	DoS traces (pkt loss, jitter)
scapy	Injection/spoofing	Network	Control plane anomalies
ettercap	ARP poisoning	Link	Traffic redirection
Wireshark	Passive sniffing	All	Ground-truth capture

All events are logged to attacktrace.csv (timestamps, node IDs, attack tags, PDR, RTT, Q-deltas). Table 5 demonstrates the RL and HGS parameters used in this simulation.

Classification performance metrics: To assess the effectiveness of AFRL-HGS under adversarial conditions, we use three widely adopted metrics: Accuracy, F1-score, and a robust Z-score (denoted Z1) to capture anomaly sensitivity. Each metric is computed over 10 independent runs, each lasting 100 simulation rounds, with adversarial traces injected as described in Section 4.

Table 5. RL and HGS parameters.

Parameter	Value
RL algorithm	Tabular Q-learning + FRL
Learning rate η_t	$\eta_0 = 0.2$, decay $\eta_t = \eta_0 / (1 + 0.001t)$
Discount factor γ	0.9
Exploration ε_t	$\varepsilon_0 = 0.25 \rightarrow \varepsilon_{\min} = 0.05$ (exp decay)
Federation interval T_{fed}	50 episodes
FRL blend factor ζ	0.2 (Polyak update)
Performance window W_G	20 episodes
HGS population/iterations	50/100
Binarization	Sigmoid + Bernoulli
Reward weights $(\lambda_1, \lambda_2, \lambda_3)$	(0.4, 0.3, 0.3)
Security threshold θ_{sec}	0.7
Trust parameters	$\rho = 0.5, \delta = 0.1, \mu = 0.05$
Reliability window W_L	10 (EWMA, $\lambda_L = 0.2$)

Accuracy measures the fraction of correctly classified decisions (secure vs. compromised paths):

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}. \quad (42)$$

F1-score balances precision and recall:

$$\text{F1-score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}. \quad (43)$$

Robust Z-score (Z1) quantifies the deviation of error observations from the median, using the Median Absolute Deviation (MAD):

$$Z1_i = \frac{0.6745 \cdot |X_i - \text{Median}(X)|}{\text{Median}(|X_j - \text{Median}(X)|)}. \quad (44)$$

Here X_i is the deviation score of observation i (packet loss or Q-value drift). Lower Z1 indicates that routing behavior remains close to the nominal median even under attacks, i.e., stronger anomaly resilience.

The proposed AFRL-HGS framework achieves the highest overall Accuracy and F1-score, while maintaining the lowest Z1, thereby demonstrating both strong classification reliability and robustness against adversarial perturbations. In contrast, baseline methods such as DQN-EQSR and BHA-ACO exhibit sharp performance degradation under spoofing and sinkhole attacks, resulting in elevated Z1 values (indicating more volatile behavior) and reduced F1-scores. The detailed results are reported in Table 6.

Table 6. Classification performance.

Method	Accuracy (%)	F1-Score	Z1-Score
Proposed (AFRL-HGS)	98.1	0.962	1.12
RFO	94.5	0.924	1.46
TCHS	93.0	0.905	1.55
BHA-ACO	92.2	0.889	1.64
DQN-EQSR	90.0	0.860	1.77

4.1. Energy Consumption

We report energy using the first-order radio model from Step 1. The total energy is

$$E_{\text{consumption}} = \sum_{(i,j) \in \mathcal{E}} \left(E^{\text{tx}}(k_{ij}, d_{ij}) + E^{\text{rx}}(k_{ij}) \right) + E_{\text{ctrl}} + E_{\text{FRL}}, \quad (45)$$

where $E^{\text{tx}}(k, d) = E_{\text{elec}}k + \epsilon(d)kd^\eta$, $E^{\text{rx}}(k) = E_{\text{elec}}k$ (Step 1), E_{ctrl} accounts for HELLO/ACK control traffic (Equation (14)), and E_{FRL} accounts for compressed FRL uploads.

Figure 6 compares the proposed method with four existing approaches: RFO, TCHS, BHA-ACO, and DQN-EQSR. The figure shows energy consumption as the number of nodes increases. All methods use more energy as the network size grows from 0 to 100 nodes. However, the proposed approach consistently uses less energy than the others at every network size. DQN-EQSR and BHA-ACO have the highest energy consumption, especially when the network exceeds 60 nodes. AFRL's adaptive reward shaping explicitly penalizes unnecessary transmissions and balances traffic across Cluster Heads, while HGS optimizes cluster formation to reduce multi-hop relay distance. This dual mechanism lowers per-packet energy cost and extends lifetime.

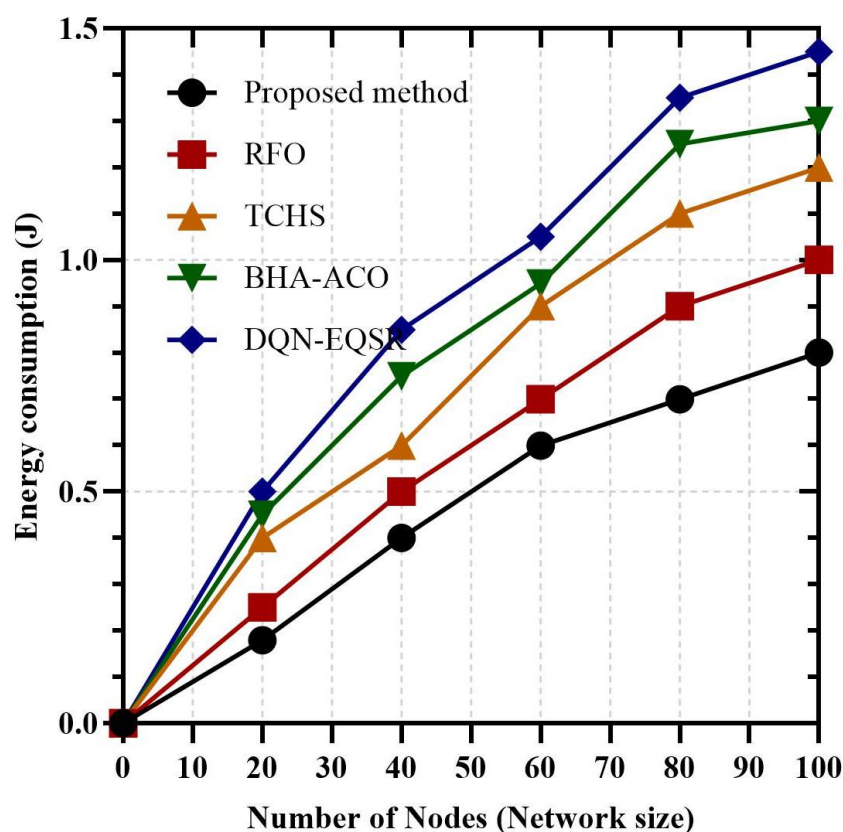


Figure 6. Energy consumption.

4.2. Network Lifetime

Network lifetime is the period between the deployment of sensor nodes and the point when the network is no longer operational. Once the first sensor node fails, the network begins to lose coverage, may become fragmented, and more nodes start to go offline.

The estimated network lifetime is defined as:

$$E_{\text{lifetime}} = \frac{0 - \mathbb{E}[\text{ERE}]}{CP + \mathbb{E}[\text{ECE}]} \quad (46)$$

Here, CP is the Constant Power consumption of the network, $\mathbb{E}[ERE]$ is the Expected Remaining Energy after processing, and $\mathbb{E}[ECE]$ is the Expected Energy Consumed for data communication per unit of time. The initial, non-rechargeable energy budget is denoted by 0. Traditional methods generally result in a shorter network lifetime compared to the proposed approach.

Figure 7 shows how network lifetime (measured in rounds) changes as the number of nodes increases from 0 to 100. The proposed method keeps the network alive longer at every network size and consistently outperforms RFO, TCHS, BHA-ACO, and DQN-EQSR, while all approaches experience a decrease in network lifetime as more nodes are added mainly due to higher energy use and communication overhead the proposed strategy declines more slowly. This suggests better load balancing and energy management. In contrast, BHA-ACO and DQN-EQSR have the shortest network lifetimes, especially when the network exceeds 60 nodes.

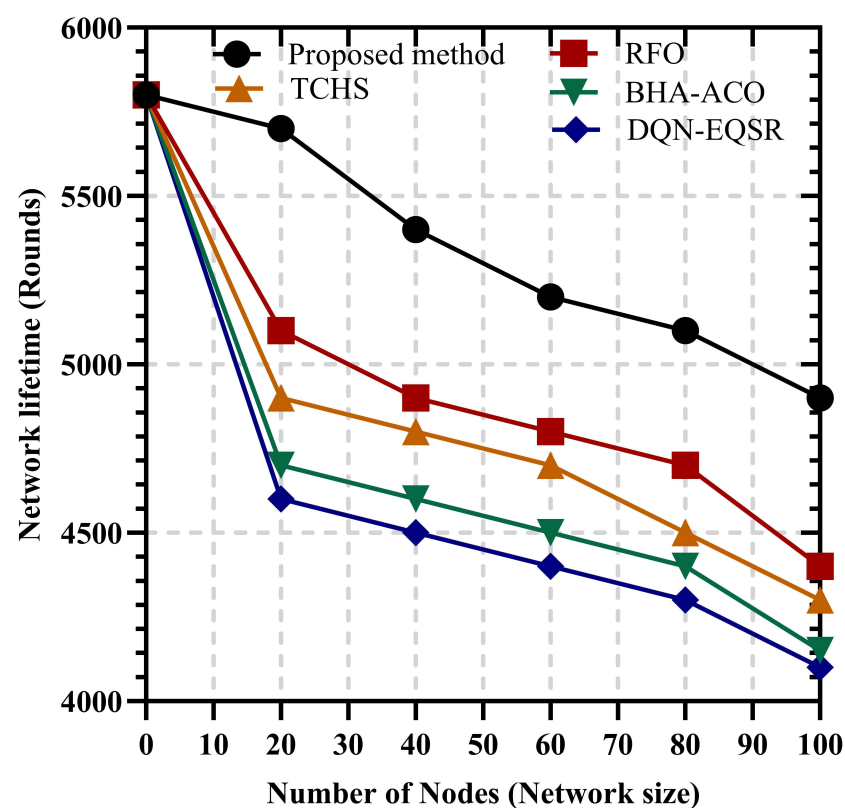


Figure 7. Network lifetime.

4.3. Packet Delivery Ratio

The cluster-based routing strategy builds routes using the shortest paths and prioritizes CHs with higher residual energy. This approach reduces the number of packet re-transmissions and energy consumption, while also making packet forwarding more reliable and efficient. Simulation results show that the PDR usually decreases as the number of nodes increases. The mathematical definition of PDR is:

$$PDR = \frac{\text{Total number of data packets received by the receiver}}{\text{Total number of data packets transmitted by the transmitter}} \quad (47)$$

A higher PDR means that data is transferred more reliably across the network.

Figure 8 shows the PDR as a percentage versus network size, with the number of nodes ranging from 10 to 100. The proposed method keeps the PDR close to 99% even as the network grows, achieving the best results across all node densities. In contrast, as

the number of nodes increases, the PDR of the other methods RFO, TCHS, BHA-ACO, and DQN-EQSR especially drops noticeably. This means that data delivery becomes less reliable when the network is congested or under scalability stress. DQN-EQSR performs the worst, with its PDR dropping below 95% at 100 nodes. These findings demonstrate that the proposed method can maintain high Accuracy in data delivery, even in large and crowded networks. The security constraint ensures that only reliable and authenticated paths are chosen, reducing packet drops from malicious or unstable nodes. At the same time, the HGS optimization avoids bottlenecks by distributing load, improving reliability.

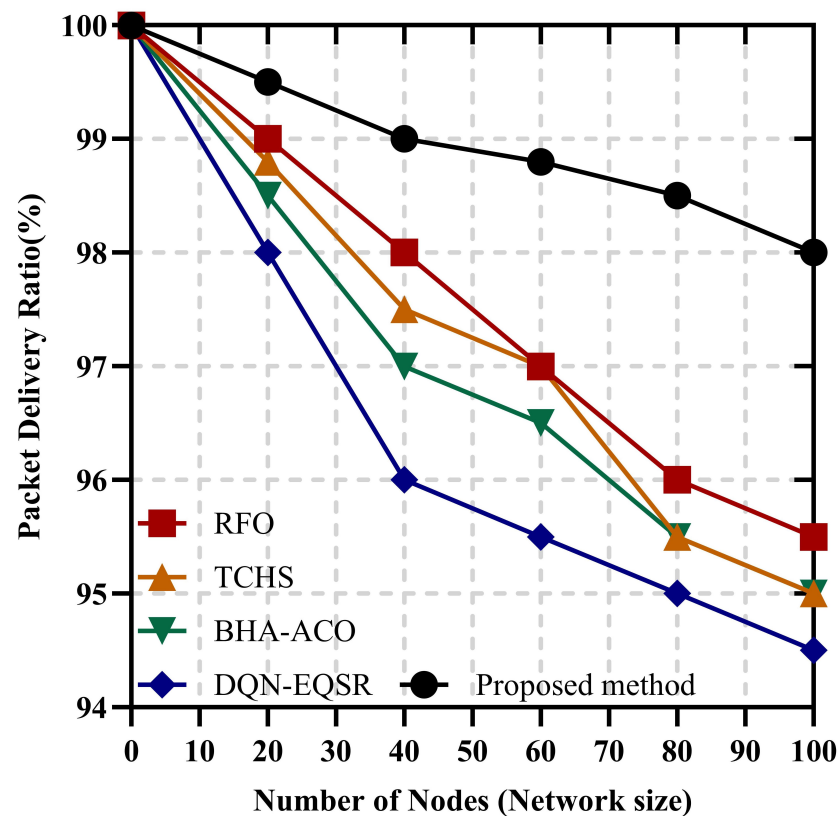


Figure 8. PDR.

4.4. End-to-End Delay (E2ED)

The average per-packet delay is

$$E2ED = \frac{1}{P} \sum_{i=1}^P (t_i^{\text{received}} - t_i^{\text{sent}}), \quad (48)$$

with P successfully delivered packets. As network size increases, E2ED rises due to longer paths and higher contention. Figure 9 shows the packet drop ratio as the network size increases from 10 to 100 nodes. The packet drop ratio measures the percentage of packets that are not delivered successfully, so higher values mean worse performance. The proposed method consistently achieves the lowest packet drop ratio across all network sizes, showing its reliability and ability to maintain data integrity as the network grows. In contrast, DQN-EQSR and BHA-ACO have the highest drop ratios, both above 5% at 100 nodes. The FRL agent learns to prioritize next-hops with higher SNR and lower hop count. This reduces re-transmissions and queue buildup, thereby lowering delay compared to static schemes.

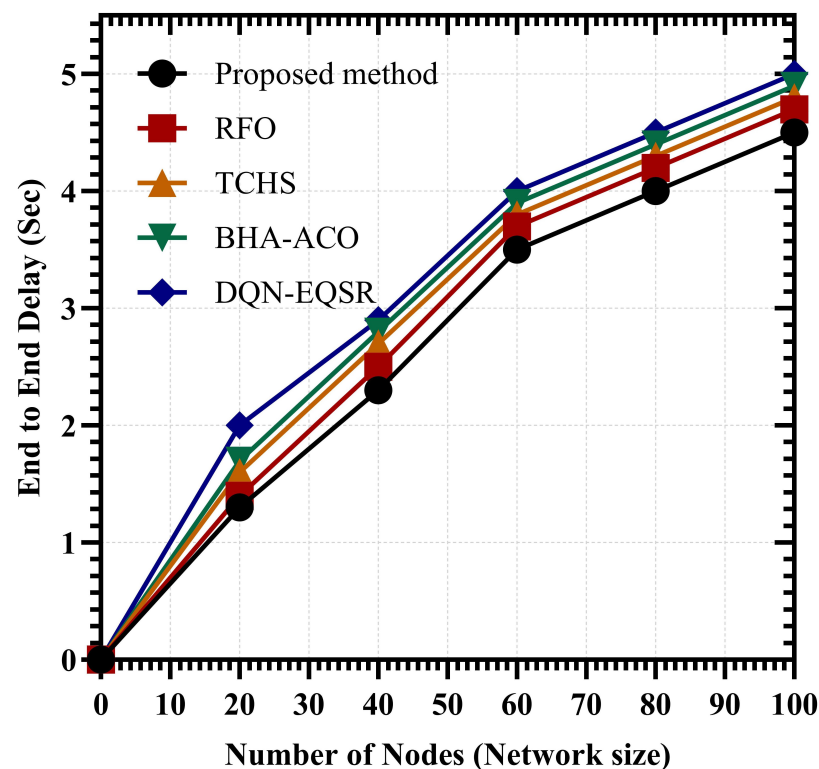


Figure 9. E2ED.

4.5. Packet Loss Ratio (PLR)

We use the identity $PLR = 1 - PDR$. Accordingly, Figure 10 mirrors the trend in Figure 8. For example, at $N = 100$ the proposed method attains $PDR \approx 98\%$; therefore, $PLR \approx 2\%$, which is the lowest among all the methods. Figure 10 shows the PLR for different routing strategies as a percentage, based on network size. All methods see an increase in PLR as the number of nodes grows from 10 to 100, mostly due to congestion and higher communication costs. However, the proposed method consistently achieves the lowest PLR for all network sizes, showing it can reduce packet loss even in crowded networks. DQN-EQSR and BHA-ACO have the highest loss rates, going above 12% at 100 nodes. This indicates less reliability in larger networks.

4.6. Bit Error Rate

Bit Error Rate (BER) is an important performance metric that measures how accurately data are transferred in WSNs. A lower BER means better signal quality and more reliable data transmission. Factors such as transmission distance, modulation methods, noise, and interference all affect the BER. It is defined as the ratio of the total number of bits transmitted to the number of bits received incorrectly:

$$BER = \frac{\text{Number of erroneous bits received}}{\text{Total number of bits transmitted}} \quad (49)$$

Figure 11 shows the BER as a percentage as the network size increases from 10 to 100 nodes. All methods display a downward trend, meaning that data accuracy improves as the network becomes larger. However, the proposed solution consistently achieves the lowest BER at every network size, reducing it from 15% down to just 4% as the network grows. In comparison, DQN-EQSR maintains the highest BER, staying above 8% even at 100 nodes, while BHA-ACO is the second highest. The strong performance of the proposed method shows its ability to provide reliable and error-resistant data transfer.

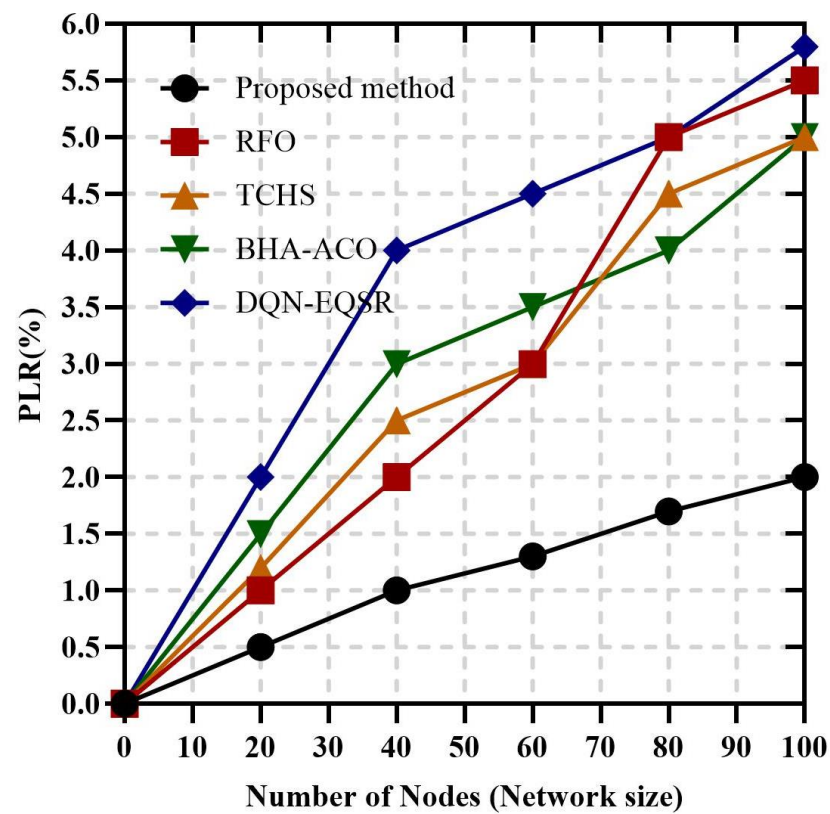


Figure 10. PLR.

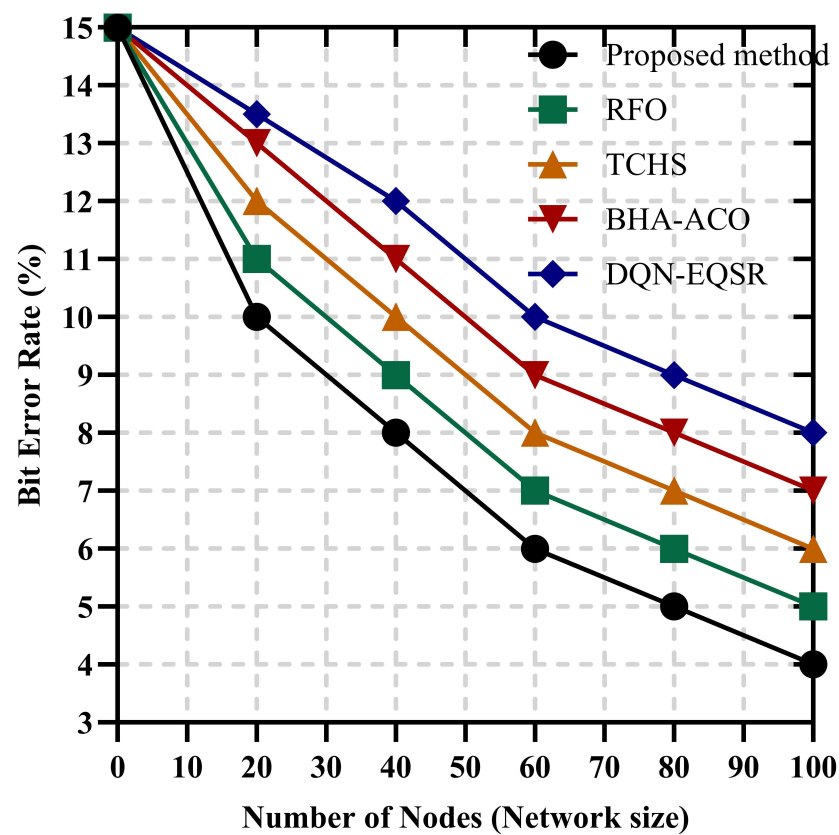


Figure 11. BER.

4.7. Channel Load

We report *average per-node channel occupancy*

$$C_L^{\text{node}} = \frac{1}{N} \sum_{i=1}^N \frac{D_i}{TB}, \quad (50)$$

where D_i is the number of bits transmitted by node i over interval T and B is the channel bandwidth. Figure 12 shows slightly higher occupancy for AFRL-HGS at small N , stabilizing as N grows, consistent with higher PDR and lower PLR. This can lead to packet loss, collisions, and increased latency. Effective load balancing and congestion control are needed to maintain optimal network performance.

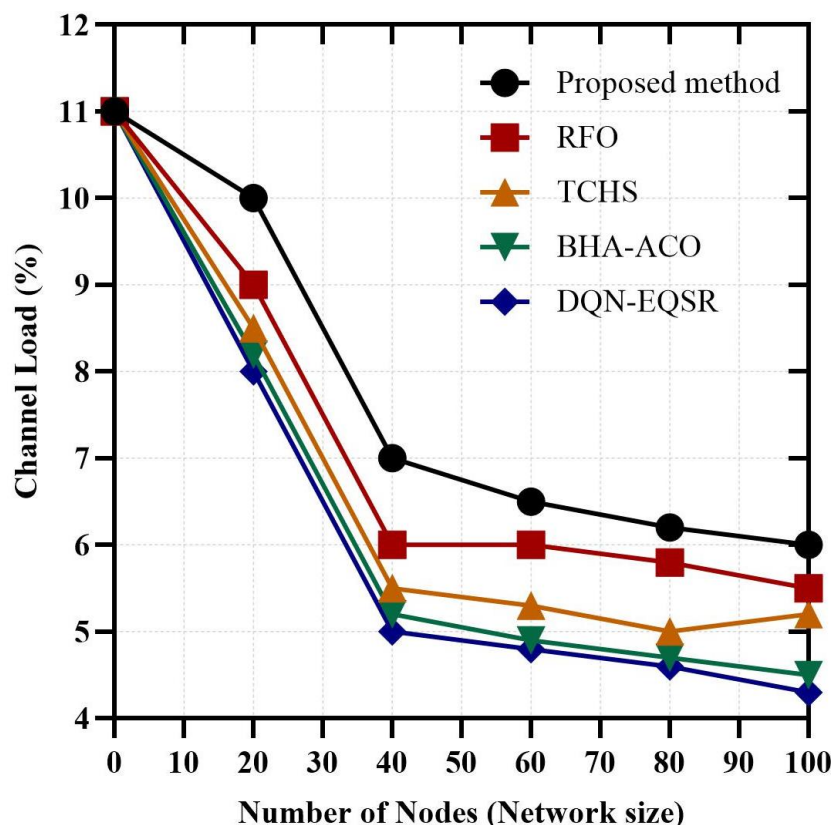


Figure 12. Channel load.

4.8. Computational Complexity Analysis

Let N be the number of nodes, E be the number of links (with $E = O(N)$ in sparse networks), G be the HGS population size, T be the number of HGS generations until convergence, A be the number of next-hop actions per node, M be the number of participants per FRL round, and d be the parameter dimension communicated per participant. In our setting, d is constant and does not scale with N .

HGS component. Each generation evaluates G candidate solutions. A single fitness evaluation requires a traversal of the network, which costs $O(N + E)$. Therefore, one generation has cost $O(G(N + E))$, and over T generations the HGS search requires

$$O(TG(N + E)) = \begin{cases} O(TGN), & \text{if } E = O(N) \text{ (sparse graphs).} \end{cases}$$

Since G and T are typically small constants, this component scales nearly linearly with N in sparse networks.

FRL component (per round). Assume each participant (a network node) performs a single local update per round. Each local step evaluates a maximum over A actions, costing $O(A)$. Thus, the total local compute is $O(MA)$. The server aggregates M parameter vectors

of dimension d , costing $O(Md) = O(M)$ for constant d . Uplink communication involves M parameter vectors ($O(Md)$), while downlink costs $O(d)$ with broadcast (or $O(Md)$ if unicast). Hence, the per-round FRL system's work is

$$O(MA),$$

with communication $O(Md)$ uplink + $O(d)$ broadcast downlink.

Overall complexity. The end-to-end per-cycle system's work (one HGS run plus one FRL round) is

$$\text{Compute: } O(TG(N + E) + MA) = \begin{cases} O(TGN + MA), & (\text{sparse graphs}). \end{cases}$$

$$\text{Communication: } O(Md) \text{ uplink} + O(d) \text{ (broadcast downlink).}$$

Thus, when T , G , and A are bounded, and $M = O(N)$ or capped, both computation and communication scale in a near-linear manner with N in sparse networks. The comparative complexities of AFRL-HGS and baseline methods are summarized in Table 7.

Table 7. System work complexity comparison between AFRL-HGS and baseline methods.

Scheme	Complexity
LEACH [31] (cluster setup, typical)	$O(N^2)$
PEGASIS [32] (chain formation, greedy)	$O(N^2)$
Centralized Q-learning [33] (per step, N agents)	$O(NA)$
Secure RL [34] (centralized, privacy-preserving aggregation)	$O(NA + N^2)$
AFRL-HGS (proposed)	$O(TG(N + E) + MA)$

5. Open Issues and Future Work

Although the AFRL-HGS framework brings notable improvements, several open challenges remain in achieving optimal energy efficiency and robust security in WSN routing. These challenges cover algorithmic, architectural, deployment, and performance aspects. This section highlights the main unresolved issues and suggests possible directions for future research:

- **Scalability to heterogeneous and ultra-dense networks:** The proposed framework has been tested with 100 nodes. Real-world WSNs, especially in smart cities and industrial IoT, may involve thousands of nodes with different capabilities. Future work should explore hierarchical federated architectures or clustered aggregation to scale up without overloading the sink node or causing bottlenecks.
- **Support for dynamic topologies and node mobility:** AFRL-HGS assumes a static or slowly changing network. Applications like vehicular networks (VANET), wildlife tracking, or mobile asset monitoring need support for dynamic topologies. Routing policies and learning intervals should adapt to mobility to keep performance high.
- **Cold-start issue and trust bootstrapping:** Initial trust values for unknown nodes are set heuristically, which can slow convergence or create security risks. Approaches like reputation inheritance, behavioral fingerprinting, or knowledge from past deployments may help solve this problem.
- **Sturdiness against intelligent and adaptive adversaries:** The current approach defends against traditional attacks but may be exposed to adversaries using advanced techniques like mimicking attacks or adversarial RL. Co-evolutionary models or adversarial training could help agents in adapting to smarter threats.
- **Energy-aware strategies for attack defense:** The architecture focuses on energy-aware routing, but energy-aware attacks are not fully considered. Attackers could target

energy limits to force re-clustering or early routing changes. Game-theoretic defense strategies could improve resistance.

- **Tailored federated synchronization in real-time:** Federated Q-learning reduces centralized data transfer but periodic synchronization still adds overhead and delay. Finding the right balance between sync frequency, speed of convergence, and communication cost is a key challenge, especially for low-bandwidth or delay-tolerant WSNs.
- **Opportunities for cross-layer optimization:** The current method focuses on network-layer routing. Cross-layer optimization combining MAC scheduling, data compression, or adaptive duty cycling could further improve the energy-security trade-off. Integrated RL across multiple layers is a promising direction.
- **Explainability and human trust:** As WSNs are used in critical fields like healthcare and environmental safety, routing decisions by learning agents must be explainable. Techniques like explainable RL (XRL) and providing confidence scores or policy traces can boost trust and simplify debugging.
- **Integration with fog and edge computing:** Currently, aggregation is centralized at the sink. Offloading learning and aggregation to edge or fog nodes could reduce latency and boost fault tolerance. Decentralized federated learning with edge-fog coordination is worth exploring.
- **Energy harvesting and hybrid energy models:** The AFRL-HGS model assumes static, non-rechargeable nodes. New deployments may use energy harvesting (solar, kinetic, thermal). Routing strategies need to estimate energy availability, adapt to fluctuations, and balance processing load with harvesting efficiency.
- **Federated communication security:** While federated learning keeps data local, Q-value updates can be intercepted or manipulated. Secure aggregation, differential privacy, or blockchain-based validation could make global model updates more secure.
- **Domain adaptation and transfer learning:** Many WSN deployments are similar in design. Sharing knowledge (pretrained Q-values or trust matrices) across deployments could save training time and reduce initialization overhead.
- **Validation on hardware testbeds:** Simulations in GNS3 and adversarial injection via Kali Linux are helpful, but real hardware validation is essential. Testing AFRL-HGS on devices like TelosB, MicaZ, or IoT-LAB under real-world conditions would give a more complete evaluation.
- **Multi-server and multi-sink settings:** The current model uses a single sink node. In practice, large-scale networks often use multiple sinks or cloud-edge hybrids. Extending AFR-HGS to support multi-sink coordination and federated learning across sinks remains an open challenge.
- **Verification of RL-based routing policies:** Many WSNs are safety-critical, so learned routing policies must not cause risky situations like partitioning or loops. Future research could use formal verification, such as model checking or theorem proving, to guarantee safe agent behavior.

Once these open challenges are resolved, the AFRL-HGS framework can become more practical for real-world deployments, offering enhanced scalability, interpretability, and robustness beyond academic research.

6. Conclusions

This study introduced AFRL-HGS, a Hybrid Routing method that combines FRL and the HGS algorithm to solve problems with energy efficiency and attack resistance in WSNs. In this approach, each sensor node is modeled as a Q-learning agent, which enables decentralized policy learning through federated aggregation and eliminates the need for centralized raw data exchange. The HGS algorithm further improves early routing quality

and speeds up learning convergence by optimizing initial clustering and routing based on objectives like energy use, delay, and path security. The proposed framework also uses dynamic trust evaluation and anomaly detection to identify and isolate malicious nodes, which strengthens the network against threats such as selective forwarding, sinkhole, and spoofing attacks. Simulation results from the hybrid GNS3 and Kali Linux testbed show that AFRL-HGS outperforms existing protocols like RFO, TCHS, BHA-ACO, and DQN-EQSR. It uses less energy, increases network lifetime, improves classification Accuracy and PDR, and is more resilient to attacks.

Future research will focus on adding explainable and secure federated learning, supporting mobile nodes, enabling multi-sink coordination, and adapting to energy-harvesting scenarios. The real-world applicability of the AFRL-HGS framework will also be tested through deployments in diverse IoT settings and on actual hardware testbeds.

Author Contributions: Conceptualization, S.S.S.; Methodology, S.S.S., S.N. and S.G.O.; Software, S.N.; Validation, S.T.S. and R.Z.F.; Formal analysis, S.S.S. and S.G.O.; Investigation, S.S.S. and S.N.; Data curation, S.T.S.; Writing-original draft, S.S.S.; Writing-review and editing, S.T.S., R.Z.F. and S.G.O.; Visualization, S.T.S.; Supervision, S.G.O.; Project administration, S.S.S.; Funding acquisition, S.G.O. All authors have read and agreed to the published version of the manuscript.

Funding: This work is the result of the successful implementation of the project “Increasing the innovative competitiveness of SC Ad Net Market Media SRL through initial innovation investments in order to create a SmartDelta technological platform, within a newly established unit for joint R&D activities”, funded by the EU under the Competitiveness Operational Program (POC), priority axis 1, action 1.2.1. The article was funded by the National University of Science and Technology POLITEHNICA Bucharest through the ‘PubArt’ Programme.

Data Availability Statement: No new data were created or analyzed in this study.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Suganthi, S.; Umapathi, N.; Venkateswaran, N.; Rajarajan, S. Enhancing energy efficiency in wireless sensor networks via predictive model for node status classification and coverage integrity using relational Bi-level aggregation graph convolutional network. *Expert Syst. Appl.* **2025**, *296*, 129029. [\[CrossRef\]](#)
2. Khalifeh, A.; Mazunga, F.; Nechibvute, A.; Nyambo, B.M. Microcontroller unit-based wireless sensor network nodes: A review. *Sensors* **2022**, *22*, 8937. [\[CrossRef\]](#)
3. Ali, A.; Ming, Y.; Chakraborty, S.; Iram, S. A comprehensive survey on real-time applications of WSN. *Future Internet* **2017**, *9*, 77. [\[CrossRef\]](#)
4. Al-Turjman, F.M.; Hassanein, H.S.; Ibnkahla, M. Towards prolonged lifetime for deployed WSNs in outdoor environment monitoring. *Ad Hoc Netw.* **2015**, *24*, 172–185. [\[CrossRef\]](#)
5. Hassija, V.; Chamola, V.; Agrawal, A.; Goyal, A.; Luong, N.C.; Niyato, D.; Yu, F.R.; Guizani, M. Fast, reliable, and secure drone communication: A comprehensive survey. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 2802–2832. [\[CrossRef\]](#)
6. Selvi, M.; Santhosh Kumar, S.; Thangaramya, K.; Abdul Gaffar, H. Energy efficient trust aware secure routing algorithm with attribute based encryption for wireless sensor networks. *Sci. Rep.* **2025**, *15*, 19724. [\[CrossRef\]](#)
7. Oztoprak, A.; Hassanpour, R.; Ozkan, A.; Oztoprak, K. Security challenges, mitigation strategies, and future trends in wireless sensor networks: A review. *ACM Comput. Surv.* **2024**, *57*, 1–29. [\[CrossRef\]](#)
8. Nigam, R.; Kumar Sharma, D.; Jain, S.; Krishna Bhardwaj, K.; Banyal, S. AI-enabled trust-based routing protocol for social opportunistic IoT networks. *Trans. Emerg. Telecommun. Technol.* **2024**, *35*, e4330. [\[CrossRef\]](#)
9. Arasteh, B.; Arasteh, K.; Kiani, F.; Sefati, S.S.; Fratu, O.; Halunga, S.; Tirkolaee, E.B. A bioinspired test generation method using discretized and modified bat optimization algorithm. *Mathematics* **2024**, *12*, 186. [\[CrossRef\]](#)
10. Abujassar, R.S. An Innovative Algorithm for Multipath Routing and Energy Efficiency in IoT Across Varied Network Topology Densities. *Int. J. Netw. Distrib. Comput.* **2025**, *13*, 14. [\[CrossRef\]](#)

11. Ganesh, S.S.; Kalpana, V.; Lakshmi, T.V.; SreeKanth, N. Intelligent QoS-Driven Ad Hoc On-Demand Distance Vector Routing for 5G MANET Using Physically Recurrent Neural Network. *Int. J. Commun. Syst.* **2025**, *38*, e70148. [\[CrossRef\]](#)
12. Illi, E.; Qaraqe, M.; Althunibat, S.; Alhasanat, A.; Alsafasfeh, M.; De Ree, M.; Mantas, G.; Rodriguez, J.; Aman, W.; Al-Kuwari, S. Physical layer security for authentication, confidentiality, and malicious node detection: A paradigm shift in securing IoT networks. *IEEE Commun. Surv. Tutor.* **2023**, *26*, 347–388. [\[CrossRef\]](#)
13. Gautam, A.K.; Kumar, R. A comprehensive study on key management, authentication and trust management techniques in wireless sensor networks. *SN Appl. Sci.* **2021**, *3*, 50. [\[CrossRef\]](#)
14. Dixit, S.; Qureshi, S. Security-aware, Red fox optimization-based cluster-based routing in wireless sensor network. *Peer Peer Netw. Appl.* **2025**, *18*, 128. [\[CrossRef\]](#)
15. Arat, F.; Akleyek, S. Security-aware RPL: Designing a novel objective function for risk-based routing with rank evaluation. *Comput. Netw.* **2025**, *260*, 111122. [\[CrossRef\]](#)
16. Karunkuzhali, D.; Meenakshi, B.; Lingam, K. A QoS-aware routing approach for Internet of Things-enabled wireless sensor networks in smart cities. *Multimed. Tools Appl.* **2025**, *84*, 17951–17977. [\[CrossRef\]](#)
17. Sefati, S.; Abdi, M.; Ghaffari, A. Cluster-based data transmission scheme in wireless sensor networks using black hole and ant colony algorithms. *Int. J. Commun. Syst.* **2021**, *34*, e4768. [\[CrossRef\]](#)
18. Sefati, S.S.; Tabrizi, S.G. Cluster head selection and routing protocol for wireless sensor networks (WSNs) based on software-defined network (SDN) via game of theory. *J. Electr. Electron. Eng.* **2021**, *9*, 100–115. [\[CrossRef\]](#)
19. Udayaprasad, P.; Shreyas, J.; Srinidhi, N.; Kumar, S.D.; Dayananda, P.; Askar, S.S.; Abouhawwash, M. Energy efficient optimized routing technique with distributed SDN-AI to large scale I-IoT networks. *IEEE Access* **2024**, *12*, 2742–2759. [\[CrossRef\]](#)
20. Zhang, M.; Liu, Y.; Chen, H.; Cai, W. Double DQN-based Efficient Quality of Service Routing protocol in Internet of Underwater Things with mobile nodes. *Ad Hoc Netw.* **2025**, *175*, 103856. [\[CrossRef\]](#)
21. Mostafa, B.; Molnar, M. Offloaded Computation for QoS Routing in Wireless Sensor Networks. *Information* **2025**, *16*, 464. [\[CrossRef\]](#)
22. Bhavani, A.; Venkataramana, A.; Chakravarthy, A. Multi-Objective Hybrid Green Anaconda Skill Optimization Enabled Energy and Cache Based QoS Aware Routing in Delay Tolerant-IoT Network. *Sustain. Comput. Inform. Syst.* **2025**, *47*, 101158. [\[CrossRef\]](#)
23. Sefati, S.S.; Fartu, O.; Nor, A.M.; Halunga, S. Enhancing internet of things security and efficiency: Anomaly detection via proof of stake Blockchain techniques. In Proceedings of the 2024 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC), Osaka, Japan, 19–22 February 2024; IEEE: New York, NY, USA, 2024; pp. 591–595.
24. Sefati, S.S.; Halunga, S. Mobile sink assisted data gathering for URLLC in IoT using a fuzzy logic system. In Proceedings of the 2022 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), Sofia, Bulgaria, 6–9 June 2022; IEEE: New York, NY, USA, 2022; pp. 379–384.
25. Arasteh, B.; Bouyer, A.; Sefati, S.S.; Craciunescu, R. Effective SQL injection detection: A fusion of binary olympiad optimizer and classification algorithm. *Mathematics* **2024**, *12*, 2917. [\[CrossRef\]](#)
26. Luo, J.; Wu, D.; Pan, C.; Zha, J. Optimal energy strategy for node selection and data relay in WSN-based IoT. *Mob. Netw. Appl.* **2015**, *20*, 169–180. [\[CrossRef\]](#)
27. Lakshmi, M.S.; Ramana, K.S.; Ramu, G.; Shyam Sunder Reddy, K.; Sasikala, C.; Ramesh, G. Computational intelligence techniques for energy efficient routing protocols in wireless sensor networks: A critique. *Trans. Emerg. Telecommun. Technol.* **2024**, *35*, e4888. [\[CrossRef\]](#)
28. Ramya, R.; Brindha, T. Fuzzy-Driven Cluster Head Selection and Deep Learning Prediction on the Basis of Hybrid Optimization Algorithm for Multiobjective Routing in WSN-IoT. *Int. J. Commun. Syst.* **2025**, *38*, e70162. [\[CrossRef\]](#)
29. Bhabani, B.; Mahapatro, J. CluRMA: A cluster-based RSU-enabled message aggregation scheme for vehicular ad hoc networks. *Veh. Commun.* **2023**, *39*, 100564. [\[CrossRef\]](#)
30. Ramalingam, L.; Audithan, S. Trust based cluster head selection algorithm for wireless sensor network. In Proceedings of the Second International Conference on Current Trends in Engineering and Technology (ICCTET), Coimbatore, India, 8 July 2014; IEEE: New York, NY, USA, 2014; pp. 453–457.
31. Palan, N.G.; Barbadekar, B.V.; Patil, S. Low energy adaptive clustering hierarchy (LEACH) protocol: A retrospective analysis. In Proceedings of the 2017 International Conference on Inventive Systems and Control (ICISC), Coimbatore, India, 19–20 January 2017; IEEE: New York, NY, USA, 2017; pp. 1–12.
32. Lindsey, S.; Raghavendra, C.S. PEGASIS: Power-efficient gathering in sensor information systems. In Proceedings of the IEEE Aerospace Conference, Big Sky, MT, USA, 9–16 March 2002; IEEE: New York, NY, USA, 2002; Volume 3, p. 3.

33. Obi, E.; Mammeri, Z.; Ochia, O.E. A Lifetime-Aware Centralized Routing Protocol for Wireless Sensor Networks using Reinforcement Learning. In Proceedings of the 2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Bologna, Italy, 11–13 October 2021; IEEE: New York, NY, USA, 2021; pp. 363–368.
34. Sheela, M.S.; Jayakanth, J.J.; Ramathilagam, A.; Gracewell, J. Secure wireless sensor network transmission using reinforcement learning and homomorphic encryption. *Int. J. Data Sci. Anal.* **2025**, *20*, 2851–2870. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.