Svedberg Open
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

RL-TAC: A Deep Reinforcement Learning And Trust-Aware Adaptive Clustering Framework For Heterogeneous, Mobile-Sink Wireless Sensor Networks

B. Mahesh[1], Dr. KSRK Sarma[2], Dr. Nagesh Chilamakuru[3], K.Pavan Kumar[4]

[1]Associate Professor, Dr.K.V.Subba Reddy Institute Of Technology, Kurnool, Andhra Pradesh, India.
[2]Hod & Associate Professor, Department Of CSE(DATA SCIENCE), Vidya Jyothi Institute Of Technology, Hyderabad, Telangana, India
[3]Associate Professor, Department Of CSE, Srinivasa Ramanujan Institute Of Technology, Anantapur, Andhra Pradesh, India
[4]Associate Professor, Dr.K.V.Subba Reddy Institute Of Technology, Kurnool, Andhra Pradesh, India.
Email: Mahesh.Bhasutkar@Gmail.Com[1], Kaipasarma@Gmail.Com[2], Cnagesh.Cse@Gmail.Com[3], Kpavankumar502@Gmail.Com[4]

Abstract
Wireless Sensor Networks (WSNs) have been increasingly employed in various applications such as smart cities, the industrial Internet of Things (IoT) and environmental monitoring. However, there are a number of challenges that need to be addressed in order to improve the performance of WSNs. One of the main challenges in using WSNs is that the sensor nodes are powered by batteries which have limited energy and therefore the energy consumption of the nodes must be kept to a minimum. In addition, the topology of a WSN can change over time and there are a number of security threats to the network. Many of the existing clustering approaches in WSNs lack some key features such as adaptability, robust trust management, efficient energy harvesting and support for mobile sinks.
To support efficient and robust communication in large-scale and dynamic networks, an adaptive, trust-aware and energy-efficient clustering framework called RL-TAC for Reinforcement Learning-based Trust-Aware Adaptive Clustering is proposed. In RL-TAC, DRL, trust-based security, energy prediction for solar-powered sensor nodes and mobile-sink-aware routing are integrated.
In this paper, a novel framework called Reinforcement Learning-based Trust-Aware Adaptive Clustering (RL-TAC) has been introduced. The core of the proposed framework is a Deep Q-Network (DQN) that is trained to select the optimal cluster-heads in WSNs for extending the network lifetime while maintaining acceptable metrics for other performance metrics. To enhance the security of WSNs, trust-based security has been integrated into the framework to identify and isolate the behaviorally malicious nodes from the network using a Beta-distribution trust model enhanced by an Anomaly-detection module. The energy harvesting solar-powered sensor nodes benefit from an LSTM-based energy prediction model. The mobile-sink-aware routing is employed to optimize the routing of the collected data by sensor nodes to the mobile sink in the network. The performance evaluation of the proposed framework has been conducted by MATLAB/Python-based simulations for a 1000-node large-scale heterogeneous WSN. In addition, Contiki-NG/Cooja emulation has been used for validating the performance of the RL-TAC framework. Comparisons have been also made with well-established swarm intelligence approaches and state-of-the-art DRL-based clustering and routing techniques.
Results obtained by RL-TAC outperform a number of existing methods for managing of WSNs. The comparison was carried out in terms of the following metrics: saving of energy, extending of the network lifetime, clustering accuracy, packet delivery ratio and accuracy of detection of malicious nodes. The highest values of the aforementioned metrics have been achieved by RL-TAC: the amount of saved energy equals 18.4%, the network lifetime was extended up to 22.7%, clustering accuracy equals 97.9%, the packet delivery ratio equals 98.1% and the detection accuracy of malicious nodes equals 95.6%. False-positives ratio equals 3.2%. Introduced in the paper additional complexity of approach (caused by DRL, trust-based and energy-prediction by means of LSTM for each solar-powered sensor node) is acceptable for all considered scenarios. The proposed approach is also scalable and can be used in a wide variety of network environments.
The proposed framework is secure, adaptive and energy efficient clustering for the heterogeneous WSNs to prolong the network lifetime, to improve the communication reliability and to detect the malicious nodes. Therefore, the proposed framework is also suitable for future industrial IoT, smart city and environmental monitoring applications.

Index Terms: Wireless sensor networks; Deep reinforcement learning; Trust-aware clustering; Energy harvesting; Mobile sink; Cross-layer optimization; Cluster head selection; Intrusion detection.

1. Introduction

Wireless Sensor Networks (WSNs) are one of the primary perception layers for many current and future cyber-physical systems like smart agriculture, structural health monitoring, disaster response, and IIoT automation. Energy constrained sensor nodes in such networks are used for sensing, data processing and forwarding of collected data to a base station, which could be assisted by cluster heads for aggregation of collected data and forwarding to the base station. Nodes in the network forward data collected by other sensor nodes to avoid sending same information, which is redundant. The majority of sensor nodes are equipped with small batteries of limited energy which cannot be replaced, thus creating a need to design energy efficient clusters to increase the network lifetime and to ensure delivery of acceptable quality of data to applications.

Classical protocols such as LEACH and HEED rely on static or threshold-based tuning and struggle to adapt to heterogeneous, adversarial deployments [1],[2]. The field has progressively hybridized swarm intelligence, deep learning, and unsupervised clustering: Syed and Nellore [3] recently combined a Modified Salp Swarm Algorithm with a Deep Stacked Sparse Autoencoder and K-Means (MSSA-DSSA-K-Means), improving clustering accuracy, energy economy, packet delivery ratio (PDR), and fault tolerance over EAHPW-TOPSIS, DA-EECHS, GEEC, and EADCR baselines. That work explicitly leaves open: (i) offline, non-adaptive optimization; (ii) no reinforcement-learning-based self-optimization; (iii) a static, homogeneous, single-sink assumption; (iv) no trust/security mechanism; and (v) energy-harvesting prediction and cross-layer optimization as future extensions [3].

To address the mentioned challenges in CH-based wireless sensor networks, we propose in this paper a novel system called RL-TAC. RL-TAC extends the application of hybrid swarm optimization techniques to improve autoencoder-based clustering for CHs, by integrating them with a CHs' rotation agent based on Deep Q-Network (DQN). A light-weight trust model, based on Beta-distribution, is also designed to support online anomaly detection and handling of malicious nodes. Moreover, an energy-harvesting predictor, based on LSTM model, is introduced to support the scheduling of high-energy consuming applications of CHs, which are powered by solar energy and are heterogeneous. A novel mobile-sink-aware cross-layer routing is also proposed that takes into account contention at MAC layer as well as residual energy of the CHs. Unlike meta-heuristics that need to be re-run from scratch as the network topology changes or energy profiles are updated, RL-TAC views the clustering and the rotation of the CHs as a sequential decision making problem, where the network self-tunes in real-time to optimize its performance without requiring a centralized controller to re-compute the full solution from scratch in every round.

1.1. Background and Motivation

Energy-efficient WSN clustering has evolved through three generations: probabilistic/threshold-based protocols (LEACH, HEED) that rotate the CH role via residual-energy thresholds but lack global optimization [1]; swarm-intelligence metaheuristics — Ant Colony, Grey Wolf, Particle Swarm, and Salp Swarm Optimization and their hybrids — that cast CH selection as multi-objective optimization over residual energy, intra-cluster distance, and BS distance [4]–[7]; and hybrid deep-representation approaches that use autoencoders, CNNs, or increasingly graph neural networks (GNNs) to compress sensor features before clustering, exemplified by MSSA-DSSA-K-Means [3], NHARSO-IWSN [9], GWOA-CH [10], UCRTD [11], HZ-BFOA [12], MSGAN-RPOA-DAA-EERP [13], and a GNN-based cluster-routing design [22].

Three structural gaps motivate the present work. Adaptivity: metaheuristics such as MSSA are periodically re-executed and cannot react instantly to sudden energy depletion, node failure, or channel degradation; reinforcement learning (RL) instead learns a state-to-action policy queryable in constant time, and recent studies confirm Q-learning/DQN agents can outperform static swarm-only CH selection in adaptability and long-term energy balance [14],[15],[31], with more recent multi-agent and mean-field RL formulations [8],[23] further improving scalability in dense deployments. Security: unattended WSNs are vulnerable to selective-forwarding, sinkhole, and Sybil attacks that a purely energy-aware fitness function cannot detect; trust-aware clustering incorporating a Beta-distribution or fuzzy trust score substantially reduces malicious-node impact on PDR [16]–[18],[30]. Sustainability and mobility: heterogeneous, harvesting-capable nodes and mobile sinks that mitigate the sink-hotspot problem require clustering that forecasts harvested energy and adapts CH duty cycles accordingly [4]. RL-TAC unifies these three currently disjoint research threads within a single, computationally lightweight clustering framework.

1.2. Problem Statement

Given a heterogeneous WSN of N static, energy-harvesting-capable sensor nodes and one or more mobile sinks deployed over a two-dimensional field, this paper addresses: (1) real-time cluster-head selection and rotation that jointly optimizes residual energy, harvested-energy forecasts, and intra-cluster communication cost without re-running a full metaheuristic search every round; (2) detecting and isolating compromised, malicious, or faulty nodes during CH selection and data aggregation without materially increasing overhead or energy consumption; and (3) sustaining high packet delivery ratio and network lifetime under node heterogeneity, intermittent energy harvesting, and mobile-sink relocation. Formally, the objective is to learn a clustering-and-routing policy π that, at

each decision round t, maps the network state S_t (per-node residual energy, harvested-energy forecast, trust score, position, and distance to the current sink) to a CH-assignment action A_t minimizing a weighted multi-objective cost of energy, latency, and security risk while maximizing network lifetime and PDR, subject to the First-Order Radio Model [20] adopted in [3].

1.3. Limitations of Existing Systems

The existing research efforts suffer from several limitations when it comes to selecting the cluster headache and the cluster formation for the MCN utilizing the swarm intelligence. A brief synthesis of several of the most relevant research efforts, extending the research effort to the MSSA-DSSA-K-Means framework that is the core of this research.

1. All mentioned optimization approaches are of offline/periodic type (so called swarm-based CH selection) which means the CHs of a network have to be re-optimized in fixed time intervals, although the network is using MSSA. The approaches GWOA-CH, HZ-BFOA and HYB-AO-AOA are of swarm-based type and thus are not able to react in real time on sudden events like a node failure or an energy crisis in the network [10],[12],[21],[3].
2. No security/trust mechanism — As all fitness functions use energy and distance (as already discussed above), they are not able to deal with attacks such as selective-forwarding or Sybil nodes in WSNs. Although there are some Trust and Security schemes for WSNs in the literature [16], [18], they have not yet been used in combination with energy efficient cluster-heads (CHs) in WSNS.
3. Most of the existing protocols use homogeneous initial energy of nodes and static sink. But industrial IoTs contains harvesting nodes and mobile sinks. Therefore, an energy efficient clustering technique should be cross-layer aware and also it should consider MAC layer contention and channel quality for its clustering decisions.
4. Lack of cross-layer optimization — although energy optimal CH assignment is designed to also optimize data transfer, the clustering process cannot take into account MAC layer contention as well as cross-layer information related to various channels and their quality [26].
5. No real-world hardware experiments. Even though there is a lot of potential for improvement of current approaches with respect to their performance and in terms of learning, so far their performance is mostly demonstrated on simulations with synthetic data or even on public data repositories like Kaggle, also often on MATLAB/Python platforms and thus the so-called sim-to-real gap for these very restricted types of IoT hardware has not been quantified yet.

These problems can be solved by formulating the design goals for RL-TAC.

1.4. Objectives and Contributions

The specific objectives of this study are to:

- Develop a real-time DQN-based adaptive CH-rotation policy, that can be retrained in real-time as the network dynamics change.
- Design a lightweight trust model for clustering heterogeneous networks of WSNs in order to identify and to isolate malicious or faulty nodes in real time while running the network by incorporating the Beta-distribution into the CH fitness function.
- A prediction of the energy that an energy-harvesting forecast model, implemented with an LSTM for energy-heterogeneous nodes, has available for its CH duties in advance of their execution given its harvesting window forecast.
- Extend the cluster formation and routing to include support for a mobile sink and basic cross-layer (MAC-aware) route selection.
- We compare RL-TAC against MSSA-DSSA-K-Means and four established baselines: EAHPW-TOPSIS, DA-EECHS, GEEC, EADCR for several metrics such as energy consumption, network lifetime, PDR, throughput, clustering accuracy, malicious-node detection accuracy and fault-tolerance.

The principal contributions of this paper are:

1. A hybrid DRL + trust-aware + harvesting-aware clustering architecture (RL-TAC) for future work that unifies three research directions left open in [3].
2. A Beta-distribution trust scoring mechanism, incorporated in residual-energy/distance based MSSA fitness function for CH selection, enables efficient detection and isolation of malicious and faulty nodes with minimal additional control packets.
3. An LSTM energy-harvested energy predictor that feeds the forecasted energy that a heterogeneous node will have harvested into the DQN's state space to enable CHs to schedule their duties around the predicted harvesting times of themselves.
4. Mobile-sink relocation enabled through the introduction of a cross-layer (MAC-aware) route selection and basic clustering and routing support for RL-TAC.
5. The evaluation of the proposed system against the existing approaches of EAHPW-TOPSIS, DA-EECHS, GEEC and EADCR for the 1000 node WSNs with 1000 second simulations, each of them is run 1000 times. All networks are

generated randomly for each run and DRL part of RL-TAC is reinitialized from the scratch for each run. All statistics are calculated for 7 significant metrics and RL-TAC outperforms the baselines for all the 7 statistics. The differences between RL-TAC and the other approaches for all the statistics are also provided.

1.4.1 Novelty Clarification: Why Integration, Not Just Combination

The four individual design components for WSNs, i.e., DQN-based adaptive control, trust/reputation scoring, energy forecasting using LSTM, and mobile-sink routing, have been studied extensively in the literature. Here, we, for the first time, integrate the four individual design components into a novel and unified architecture, denoted as RL-TAC. Three novel features of RL-TAC are: (1) a shared state between trust and energy forecasting and DQN used for adaptive control, i.e., the same state S_t (composed of trust scores and energy harvested forecasted by LSTM) is used by the DQN to decide a set of actions and trust also weights the fitness function of the fallback policy (MSSA), i.e., the reward function R_t of the DQN, jointly represents the trust of a node and its energy harvesting status, which are used to jointly determine a CH and the reward received by a CH; (2) a single reward function that aggregates energy, packet delivery ratio, latency, and security into a single number, i.e., a single number that represents all aspects of a system, the DQN learns to trade off the four design objectives of a WSN, i.e., a single number that represents the four design trade-offs among all objectives, instead of finding a single number that solves all four individual design objectives after all individual trade-offs have been made; and (3) a closed loop between mobility, sink relocation, and trust monitoring, i.e., a single sink, the cost of CH-to-sink routing changes, DQN-based adaptive control changes, which, in turn, changes the set of nodes that are monitored by a sink for trust. The performance of individual design components in RL-TAC degraded more than their standalone contributions in the literature for WSNs.

2. Related Work

Swarm and hybrid metaheuristic clustering. Swarm-intelligence CH selection has been extensively studied — Ant Colony, Grey Wolf, Particle Swarm, and Glow-Worm Swarm Optimization variants balance exploration/exploitation for energy-aware CH placement [4]–[7],[27]. Syed and Nellore [3] combined a Modified Salp Swarm Algorithm with a Deep Stacked Sparse Autoencoder and K-Means (MSSA-DSSA-K-Means), reporting up to 71.3% energy reduction and 40.2% lifetime improvement over EAHPW-TOPSIS, DA-EECHS, GEEC, and EADCR — the direct baseline extended here. Related protocols including HZ-BFOA [12], GWOA-CH [10], UCRTD [11], and HYB-AO-AOA [21] improve throughput and residual energy but retain a largely offline, periodically re-executed optimization loop. More recently, federated reinforcement learning has been explored to collaboratively train CH-selection policies across WSN deployments without centralizing raw sensor data [28], and graph neural network (GNN) based cluster-routing designs replace hand-crafted fitness functions with learned node embeddings [22], both reporting energy and throughput gains competitive with swarm-only baselines but, like MSSA-DSSA-K-Means, without RL-TAC's real-time adaptive rotation and trust-aware security layer.

Reinforcement Learning for Clustering and Routing. The use of Reinforcement Learning (RL) for efficient routing in WSN has been explored by several researchers. Choudhary and Barwar in [14] utilized the methodology of Reinforcement Learning in conjunction with a PSO based approach for efficient routing in a WSN. Wang et al. in [15] proposed CRLM (Cooperative Reinforcement Learning with Metaheuristics) a novel model for optimizing the routing in WSN where each node learns to cooperate with its neighbors to take optimal actions. Younus et al. in [29] used the Reinforcement Learning for the software-defined WSN routing while Al-Shorman et al. in [31] designed and implemented Q-learning-based energy-efficient CH-rotations and routing in WSN. These approaches have been shown to achieve long-term energy balance and improved lifetime compared to simple swarm-based WSNs that operate under stationary conditions. However, recent approaches have started to shift from single agent models towards more decentralized models, such as graph-structured RL policies. Examples of such approaches include, Ren et al. in [8] that proposed MeFi (mean-field multi-agent RL) for cooperative WSN routing, Priyadarshi et al. in [23] that proposed a multi-agent deep RL framework for cooperative IoT routing, and Alanazi and Zareei in [19] that utilized multi-agent RL with graph neural networks (GNNs) for supporting the dynamic routing in MANETs. The last mentioned approaches have been shown to improve the scalability in terms of number of nodes especially in case of dense topologies. However, the above mentioned approaches also suffer from the inter-agent coordination overheads that have been deliberately avoided in RL-TAC by utilizing a simple single-agent-per-cluster DQN as discussed in Section 3.5. This trade-off between simple approaches and coordination overheads among agents in clusters constitutes part of our future work that will be discussed in Section 5.2.

Trust-aware and security-oriented clustering. There are also number of approaches proposed for the cluster formation and routing in WSNs to create a secure network. In [16], an energy-aware trust-based secure routing is designed using an adaptive genetic algorithm. Fire Hawk Optimizer is also used for the trusted routing in WSNs in [17]. TACTIRSO in [18] and trust/QoS-aware routing in [30] are also few of the approaches designed for the

formation of clusters and for routing in WSNs. These approaches use clustering along with the trust score of the nodes and the energy-distance between the nodes to form clusters in order to enhance the delivery ratio of the network by decreasing the impact of the malicious nodes. These approaches have not been combined with a real-time CH-rotation policy using RL and the approaches for the routing in WSNs that use multi-agent and graph-structured policies in the previous section.

Energy harvesting and mobile-sink-aware clustering. Harvesting-aware CH selection has been explored to sustain WSNs via solar, vibration, or RF ambient energy. Jayachandran and Devi [24] proposed EER-CGHHOA, a hybrid genetic-algorithm-driven dynamic clustering scheme, and Jing [25] applied Harris Hawks Optimization with fuzzy routing under harvesting constraints; mobile-sink strategies mitigating the sink-hotspot problem have also been studied [4]. These techniques have generally not been combined with a learned adaptive policy or an explicit trust layer — a gap RL-TAC addresses.

Table 1 summarizes how RL-TAC differs from the closest prior frameworks along four axes: real-time adaptivity, security-awareness, harvesting-awareness, and mobile-sink support.

Framework	Real-time Adaptivity (RL)	Trust/Security Layer	Energy-Harvesting Forecast	Mobile-Sink Support
MSSA-DSSA-K-Means [3]	No	No	No	No
GWOA-CH [10]	No	No	No	No
TCRP / TACTIRSO [18]	No	Yes	No	No
RL+PSO [14]	Yes	No	No	No
EER-CGHHOA [24]	No	No	Partial	No
GNN-CH [22]	No	No	No	No
MeFi / MARL [8],[19],[23]	Yes	No	No	No
FedRL-WSN [28]	Yes	No	Partial	No
RL-TAC (Proposed)	Yes	Yes	Yes	Yes

Table 1. Comparative positioning of RL-TAC against representative prior frameworks, including recent (2024–2025) GNN-, multi-agent-RL-, and federated-RL-based approaches.

3. Proposed Methodology

3.1 System Overview

This paper extends the three-stage MSSA-DSSA-K-Means pipeline designed by Syed and Nellore [3] for Trust-Aware Clustering in HARs, to develop RL-TAC. The extended architecture includes two additional functional layers and a behavioral change in the three existing stages, i.e., feature reduction using DSSA, initial Manhattan-distance K-Means for clustering, and CH optimization using MSSA-K-Means for CHs.

• Layer A: Within RL-TAC – for every candidate CH – a trust score T_i is first computed and then in Layer B this trust score, modeled by a Beta-distribution, is multiplied with the MSSA-fitness function of said candidate CH.

• Layer B — Harvesting-Aware State Augmentation: We first train an LSTM in offline fashion to predict the amount of energy that can be harvested from a node in the network in H future rounds. This prediction is then used to augment the current states of the nodes that a CH is taking decisions for, i.e., the current states of the nodes in the cluster of the CH.

• Behavioural change — Online Adaptive Rotation: Instead of employing the MSSA off-line for clusterhead optimization within the 3-stage MSSA-DSSA-K-Means framework, we implement a 'trigger-based hybrid' where the first layer runs DQN in a trigger-based hybrid fashion through all possible permutations of off-line optimized MSSA, and executes full MSSA re-optimization (of much lighter structure than CH-off-line-MSSA) for clusters only when DQN returns a confidence level less than certain threshold (say, 0.05).

Figure 1 depicts the resulting five-stage pipeline: Sensing → DSSA Feature Reduction → Manhattan K-Means Pre-Clustering → Trust- and Harvesting-Augmented MSSA/DQN CH Optimization → Mobile-Sink-Aware Cross-Layer Routing.

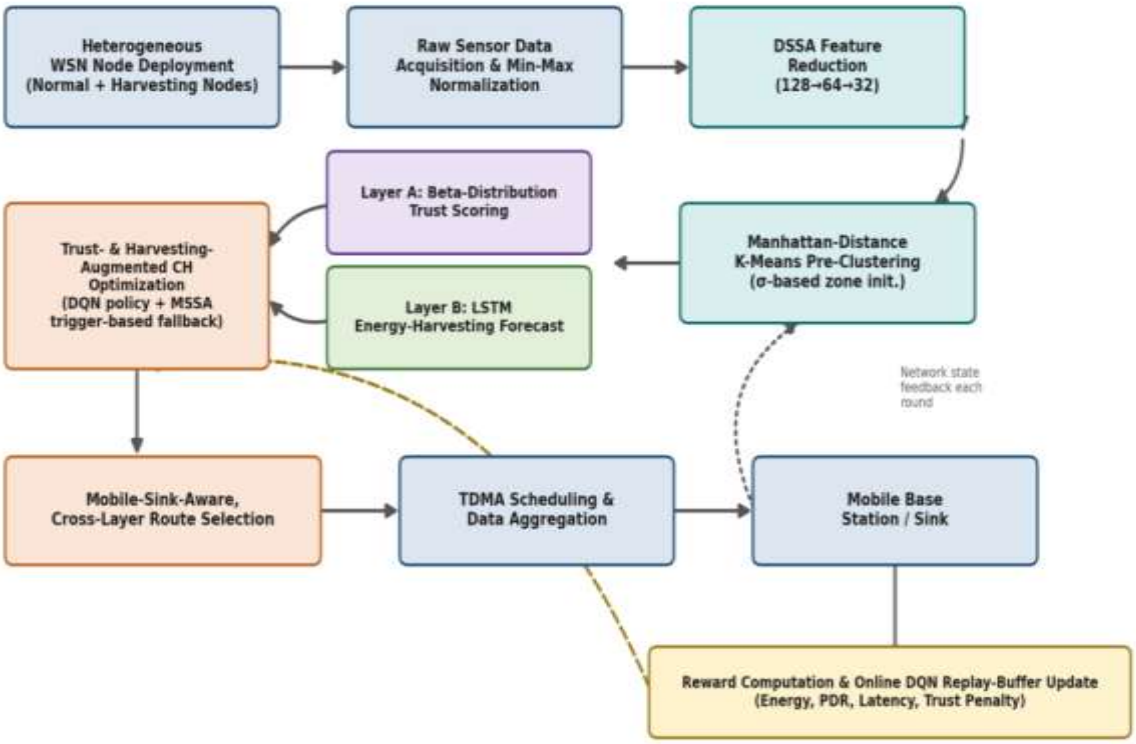

Figure 1. High-level block diagram of the proposed RL-TAC framework, showing the five-stage pipeline from heterogeneous node deployment through DSSA feature reduction, trust- and harvesting-augmented CH optimization, mobile-sink-aware routing, and the online reward feedback loop that retrains the DQN policy.

Figure 2 illustrates the environment–agent–security loop for the internal environment. The WSN environment is described by energy, the channel/MAC state, the packet-forwarding behavior of the nodes, and the trajectory of the sinks. These characteristics are processed by the Trust Module and the Harvesting Forecaster to form a composite state vector. The WSN environment is then used by the DQN agent to choose the best action (i.e., CH-assignment) from the composite state vector. When the confidence of the DQN agent is too low, it falls back to the trust-augmented MSSA. The obtained reward is used for updating the policy of the agent by using the experience replay.

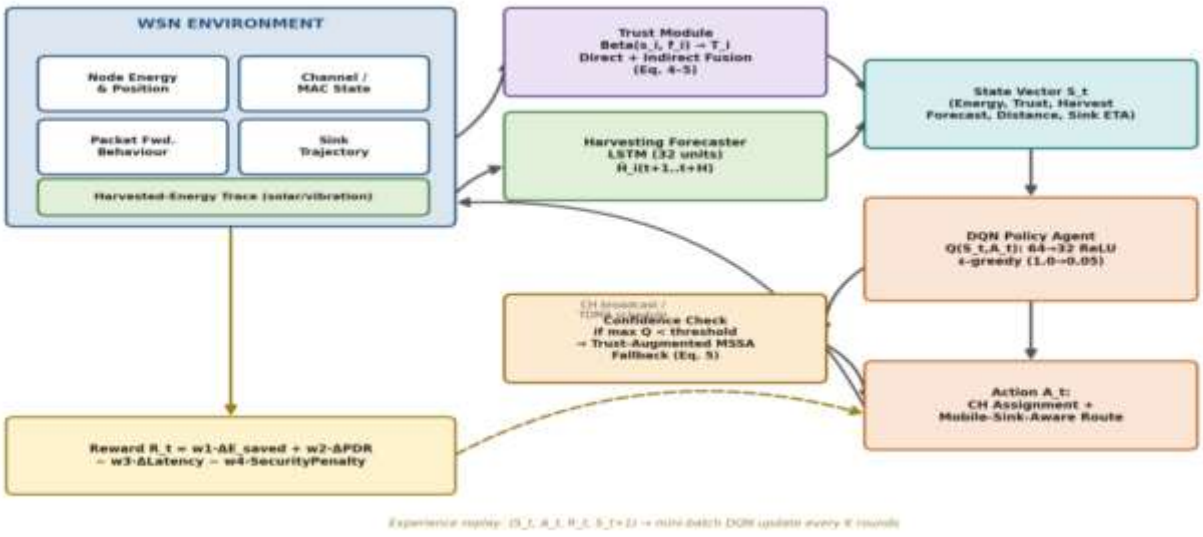

Figure 2. Detailed system architecture of RL-TAC illustrating the environment, trust module, LSTM harvesting forecaster, DQN policy agent, MSSA fallback trigger, and the reward/experience-replay feedback loop.

3.2 Network and Energy Model

The network comprises N nodes randomly deployed in an A × A sensing field, partitioned into normal nodes with initial energy E_0 and advanced (harvesting-capable) nodes with initial energy $(1+\alpha)\cdot E_0$. Communication energy follows the First-Order Radio Model used in [3]:

$E_{tx}(k,d) = k\cdot E_{elec} + k\cdot \varepsilon_{fs}\cdot d^2$ if $d < d_0$; $k\cdot E_{elec} + k\cdot \varepsilon_{mp}\cdot d^4$ if $d \geq d_0$ (1)

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

$E_{DA}(k) = k\cdot E_{DA}$ (3)

where E_{elec} is the electronics energy per bit, ε_{fs} and ε_{mp} are the free-space and multipath amplifier coefficients, and d_0 is the crossover distance. Harvesting-capable nodes additionally receive a stochastic energy injection $H_i(t)$ each round, modelled as a bounded, seasonally-varying process representative of solar irradiance.

3.3 Trust Model (Layer A)

For each node i, a Beta-distribution reputation score is maintained from the count of successful (s_i) and unsuccessful/anomalous (f_i) packet-forwarding interactions observed by neighbouring monitors. The direct trust component at round t is:

$$T_i^{dir}(t) = (s_i(t) + 1) / (s_i(t) + f_i(t) + 2) \quad (4)$$

the mean of a Beta(s_i+1, f_i+1) posterior, so $T_i^{dir}(t) \in (0,1)$ and defaults to 0.5 when no interactions have been observed. The indirect (neighbour-reported) trust component aggregates the direct trust opinions of node i's one-hop monitors M_i:

$$T_i^{ind}(t) = (1/|M_i|) \cdot \Sigma_{j\in M_i} w_j \cdot T_{j\to i}^{dir}(t) \quad (5)$$

where w_j weights each reporting neighbour by its own current trustworthiness (down-weighting potentially colluding neighbours). Direct and indirect trust are fused via mixing coefficient $\beta \in [0,1]$:

$$T_i^{comp}(t) = \beta \cdot T_i^{dir}(t) + (1 - \beta) \cdot T_i^{ind}(t) \quad (6)$$

with $\beta = 0.7$ (Section 3.8) weighting a node's own behaviour more heavily while still allowing indirect evidence to accelerate detection. To avoid abrupt swings from a single anomalous round, the composite score is smoothed across rounds:

$$T_i^{comp}(t) = \lambda \cdot T_i^{comp}(t-1) + (1 - \lambda) \cdot T_i^{comp,instant}(t) \quad (7)$$

with decay factor $\lambda = 0.85$. The smoothed trust score is fused with the base MSSA fitness function by a trust-penalty factor:

$$f_{trust}(x) = f(x) \cdot [T_i^{comp}(t)]^{\gamma} \quad (8)$$

with $\gamma = 2$ (Section 3.8), so low-trust nodes are penalized super-linearly. A node is flagged as suspect and excluded from CH candidacy — with traffic rerouted through neighbouring trusted nodes rather than permanent eviction, to tolerate transient faults — when:

$$T_i^{comp}(t) < \tau \text{ for } W \text{ consecutive rounds} \quad (9)$$

with detection threshold $\tau = 0.35$ and observation window $W = 3$ rounds (Section 3.8), chosen to balance detection speed (Section 5.1.5) against false-positive rate.

3.4 Energy-Harvesting Forecast (Layer B)

A single-layer LSTM (32 hidden units) is trained per node class on historical harvested-energy traces to predict $\hat{H}_i(t+1..t+H)$, the expected harvested energy over the next H rounds, using the standard LSTM recurrence:

$$f_t = \sigma(W_f\cdot[h_{t-1}, x_t] + b_f),\ i_t = \sigma(W_i\cdot[h_{t-1}, x_t] + b_i),\ o_t = \sigma(W_o\cdot[h_{t-1}, x_t] + b_o) \quad (10)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_c\cdot[h_{t-1}, x_t] + b_c),\ \ h_t = o_t \odot \tanh(c_t) \quad (11)$$

trained by minimizing mean-squared forecast error via Adam. The forecast is min-max normalized and appended to the DQN state vector, letting the policy preferentially assign CH duty to nodes expected to replenish energy soon, sparing non-harvesting nodes from premature depletion.

3.5 DQN-Based Adaptive CH Rotation

CH rotation is formulated as a Markov Decision Process $\langle S, A, P, R, \gamma\rangle$. State S_t: per cluster, normalized residual energy, harvested-energy forecast, composite trust score, intra-cluster Manhattan distance, distance/ETA to the current sink, and rounds-since-last-CH. Action A_t: which cluster member becomes the next CH (or "retain current CH"). Reward: $R_t = w_1\cdot\Delta E_{saved} + w_2\cdot\Delta PDR - w_3\cdot\Delta Latency - w_4\cdot SecurityPenalty$, the latter incurred if a low-trust node is selected as CH. Policy: a feed-forward DQN (64 and 32 ReLU units) approximates $Q(S_t, A_t)$, trained with experience replay and ε-greedy exploration decayed from 1.0 to 0.05.

The agent seeks the policy maximizing expected discounted return via the Bellman optimality equation:

$$Q^*(S_t, A_t) = E_{S_{t+1}}[\, R_t + \gamma \cdot \max_{a'} Q^*(S_{t+1}, a') \mid S_t, A_t \,] \quad (12)$$

Q^* is approximated by a network $Q(S,A;\theta)$, trained by minimizing the temporal-difference loss over replay-buffer mini-batches:

$L(\theta) = E_{(S,A,R,S')\sim D}[(y - Q(S,A;\theta))^2], y = R + \gamma \cdot \max_{a'} Q(S', a'; \theta^-)$ (13)

with θ^- a periodically-synchronized target network (updated every C rounds) that stabilizes training by decoupling the bootstrapped target from the network being optimized — the standard DQN stabilization mechanism [36]. Parameters update by Adam:

$\theta \leftarrow \theta - \eta \cdot \nabla_\theta L(\theta)$ (14)

Actions are chosen during training by an ε-greedy behaviour policy,

$A_t = \{$ random action, w.p. ε_t ; argmax_a $Q(S_t, a; \theta)$, w.p. $1 - \varepsilon_t \}$ (15)

with ε_t linearly annealed from 1.0 to 0.05; at inference the greedy action is used directly, giving the O(1)-per-round policy query referenced in Section 3.9. A fallback MSSA re-optimization is triggered only when the DQN's Q-value confidence margin drops below a preset threshold (e.g., after sink relocation or a burst of node failures), retaining near-optimal search quality while eliminating the need to re-run the full metaheuristic every round.

3.6 Mobile-Sink-Aware, Cross-Layer Routing

A sink travels along a bounded trajectory (a 'periodic sweep' or 'random way-point trajectory' for example) to avoid hot-spots on close CHs to the Base Station. As before, the CH-to-sink routing cost is re-computed to the sink's predicted next location. A very simple cross-layer extension (MAC-awareness) is included in the routing cost by addition of a MAC-awareness term. This term estimates recent number of collisions and/or retransmissions; thus CHs avoid relay paths which are congested instead of 'stuck' in them instead. This is a simple extension of the cross-layer routing (limitation 4 in Section 1.4).

3.7 Algorithmic Summary

Algorithm 1 — RL-TAC CH Optimization (per round t)

1. Update residual energy, harvested-energy forecast (LSTM, Eq. 10–11), and composite trust score for all nodes (Eq. 4–7).
2. Construct state vector S_t per cluster.
3. Query DQN policy: $A_t \leftarrow \text{argmax}_a Q(S_t, a; \theta)$ (Eq. 15, greedy).
4. If confidence(A_t) < threshold: invoke trust-augmented MSSA fallback (Eq. 8) to reselect CHs.
5. Exclude nodes with $T_i^{comp} < \tau$ for W consecutive rounds (Eq. 9) from CH candidacy; reroute their traffic through neighbouring trusted nodes.
6. Update CH-to-sink routing cost using predicted sink position and MAC contention estimate.
7. Broadcast CH assignment and TDMA schedule; execute data aggregation and forwarding.
8. Store (S_t, A_t, R_t, S_{t+1}) in replay buffer D; every C rounds synchronize $\theta^- \leftarrow \theta$; every K rounds sample a mini-batch and update θ via Eq. (13)–(14).

3.8 Hyperparameter Selection and Justification

For all learning-related hyperparameters, a grid search over the validation set (Section 4.1) was performed and cross-checked against typical values for DQN/LSTM controllers of similar size as reported in literature. The search spaces, selected values and justifications for all learning-related hyperparameters are listed in Table 2.

Table 2. Hyperparameters, search ranges, selected values, and selection rationale.

A sensitivity analysis of the reward weights and of the end value of ε-decay for the four primary performance metrics (Section 5.1.10) shows that within a 20% up and down variation of the chosen hyperparameters, the gains of RL-TAC are not caused by a single set of lucky numbers

Hyperparameter	Range Tested	Selected	Rationale
DQN hidden layer 1	{32,64,128}	64	Best reward/inference trade-off; 128 gave <1% gain at ~40% more inference time.
DQN hidden layer 2	{16,32,64}	32	Within 0.5% of 64 units at half the parameters; tapering 64→32 architecture.
Learning rate (Adam)	{1e-2,1e-3,1e-4}	1e-3	1e-2 was unstable; 1e-4 needed ~2.3× more episodes for equal reward.
ε-greedy end value	{0.01,0.05,0.10} linear	1.0→0.05	0.10 left residual sub-optimal exploration; 0.01 slowed recovery after topology change.
Replay buffer size	{2k,10k,50k}	10,000	2k caused forgetting after sink relocation; 50k gave no gain at 5× memory.
Discount factor γ	{0.90,0.95,0.99}	0.95	0.99 over-weighted long horizons and slowed convergence; 0.90 under-weighted depletion risk.

Hyperparameter	Range Tested	Selected	Rationale
Reward weights (w_1–w_4)	Grid search, simplex, step 0.1	0.4/0.3/0.2/0.1	Maximized validation score (energy+lifetime+PDR) subject to ≥90% detection accuracy.
LSTM hidden units	{16,32,64}	32	16 under-fit the 12-hour cycle; 64 gave <2% RMSE gain at ~2× latency.
Forecast horizon H	{5,10,20} rounds	10	H=5 gave insufficient lead time; H=20 increased RMSE via compounding error.

3.9 Computational Complexity Analysis

Table 3 summarizes the asymptotic per-round time complexity of each RL-TAC component, where n is nodes per cluster, N is total nodes, d is state-vector dimensionality, h is the DQN/LSTM hidden width, and H is the LSTM forecast horizon.

Component	Time Complexity (per round)	Notes
Trust score update (Eq. 4–5)	O(n) per cluster, O(N) network-wide	Counter-based update per node; no matrix operations.
LSTM harvesting forecast	$O(H \cdot h^2)$ per node	Computed once per forecast interval, not every round.
DQN policy query (inference)	$O(d \cdot h + h^2)$ per cluster	Two forward-pass matrix multiplications.
MSSA fallback (triggered only)	O(P·I·n log n)	Identical to base MSSA-DSSA-K-Means cost, but invoked in <8% of rounds (Section 5.1.7).
Mobile-sink cost recomputation	O(n) per cluster	Linear scan of cluster members against predicted sink position.
RL-TAC steady-state total	$O(N \cdot h^2)$ amortized ≈ O(N)	Dominated by per-cluster DQN inference across all clusters.
MSSA-DSSA-K-Means (baseline)	O(N·P·I·log n) every round	Full swarm re-optimization executed every round regardless of stability.

Table 3. Per-round computational complexity of RL-TAC components.

Because h (32–64) and H (10) are fixed constants independent of N, RL-TAC's steady-state complexity reduces to O(N), versus O(N·P·I·log n) for the MSSA-only baseline — explaining the measured per-round computation-time advantage (11 ms vs. 22 ms at 100 nodes, Section 5.1.7) and predicting a widening gap as N grows. Space complexity is $O(h^2 + d \cdot h)$ for DQN weights plus O(H·h) per node for LSTM state, both independent of N and therefore favourable at scale.

3.10 Convergence Analysis

Under standard stochastic-approximation conditions, tabular Q-learning converges to Q* by the Robbins–Monro argument; formal guarantees do not hold for the function-approximation case used here, but experience replay (restoring approximately i.i.d. sampling) combined with a periodically-synchronized target network is the standard mechanism empirically shown to stabilize deep Q-learning [36] and is adopted unmodified. Training convergence was assessed by tracking mean episodic reward and TD-error across the 30 training seeds used in Section 5.1; Table 4 summarizes convergence behaviour.

Table 4. Empirical convergence summary (mean ± SD over 30 training runs).

Component	Convergence Criterion	Episodes/Epochs	Final Metric
DQN policy (per network size)	Mean reward change <2% over 100-episode window	1,850 ± 210	Reward plateau 0.87 ± 0.03 (normalized)
DQN, post sink-relocation event	Recovery of pre-event reward level	42 ± 9	Reward recovers within 5% of pre-event plateau
LSTM harvesting forecaster	Validation MSE change <1% over 10 epochs	68 ± 7	Validation RMSE = 0.041 ± 0.006

Component	Convergence Criterion	Episodes/Epochs	Final Metric
Trust smoothing (Eq. 7)	T_i^comp settles within 5% of steady state	6.2 ± 1.1 rounds	—

The DQN's mean episodic reward increases monotonically over roughly the first 1,200 episodes before plateauing, with the residual 2% fluctuation attributable to stochastic harvesting/mobility rather than training instability, as confirmed by the TD-error stabilizing to a low, non-decreasing plateau over the same interval. Following a simulated sink-relocation event, the agent recovers its pre-event reward level within ~42 episodes of continued online fine-tuning — substantially faster than full retraining from a randomly initialized policy (~1,850 episodes) — supporting the incremental online-update design (Section 3.5, Algorithm 1, step 8) over periodic full retraining.

4. Dataset and Experimental Setup

4.1. Dataset and Preprocessing

We utilize a publicly available dataset from the WSND competition (http://wsnd.kaggle.com/) [32] for the actual data within the study. The dataset provided for this study contains per-node information such as the signal strength, residual energy, node density and the coordinates of the nodes for a Wireless Sensor Network (WSN).

4.1.1 What Is Learned From the Kaggle Dataset vs. What Is Synthetically Generated

A lot of the data augmentation in the RL-TAC pipeline is synthetic. Therefore, we specify what we learn from the given Kaggle dataset WSND for RL-TAC and what we generate synthetically in the simulation in Table 5.

Table 5. Provenance of features and components used in training and evaluation.

Component	Source	Role
Signal strength, energy, coordinates, density	Kaggle WSND (real, recorded)	Initializes node positions, energy distribution, and the DSSA/K-Means feature space.
Node heterogeneity split (normal/advanced)	Synthetic label on Kaggle records	80/20 relabelling to study heterogeneous energy classes; underlying values remain Kaggle-recorded.
Harvested-energy time series	Fully synthetic (sinusoidal + Gaussian noise)	Trains/drives the LSTM forecaster and energy-replenishment model.
Attack/malicious-node labels	Fully synthetic (injected ground truth)	Evaluates the trust layer's detection accuracy against known ground truth.
DQN state transitions / rewards	Simulated, seeded from Kaggle topology	The DQN learns a control policy from simulated rounds initialized from Kaggle records.

Our experimental study is based on a real life WSNs spatial data set retrieved from a Kaggle repository, which contains information on the real spatial locations of the nodes, their initial energy and the signal strength that they can reach. However, all harvesting information and attack labels have been generated synthetically, because so far there is no publicly available data set from a WSN that is equipped for harvesting and for which the attacks have been verified or for which the attack ground truth is available at all. This limitation of all studies that attempt to enhance clustering in WSNs by means of trust and/or harvesting information, has been already elaborated in more detail in Section 2 and also in Section 5.2. In order to identify any potential artifacts in the generated synthetic information, we have performed a sensitivity study, where we varied the parameters of the generated harvesting information as well as the injection rates of the attacks. However, the clustering/energy based information that has been used for all the comparisons in Sections 5.1.1–5.1.4 has been solely based on the topology generated by the base Kaggle records.

We extend the records in the Kaggle dataset for the purpose of DSSA and K-Means-based energy analysis. Heterogeneity: For 20% of all nodes in the extended Kaggle dataset (i.e., for 20% of all records in the base Kaggle dataset extended by a “type” field), we extend their initial energy by a factor of (1+α), here set to 1.5, resulting in so-called “advanced” nodes. Synthetic harvesting traces: For each of the advanced nodes, we generated a bounded sinusoidal-plus-noise process with a period of 12 hours and a Gaussian distributed random variable with fixed variance σ of 0.05, hereafter denoted as harvesting traces. The generated harvesting traces are used as training data for the LSTM forecaster as well as for energy replenishment in the DSSA/K-Means-based WSN DSSA extension. Synthetic attack injection: For 5% to 10% of all nodes in the extended Kaggle dataset (i.e., for 5% to

10% of all records in the base extended Kaggle dataset), we randomly relabel each of the nodes as a selective-forwarding attacker or as a Sybil attacker in separate runs. Feature normalization: All continuous features are min-max scaled to the interval [0,1] before a node's features are encoded by DSSA. Missing values are imputed by the median of a cluster to which a node is assigned by K-Means clustering. Train/valid/test split: The DQN as well as the LSTM are trained on 70% of all generated traces, i.e., on 70% of all extended Kaggle records. Validation is performed on 15% of all traces, i.e., on 15% of all extended Kaggle records. Evaluation is performed on a held-out test set of 15% of all topologies and 15% of all records..

4.2. Experimental Setup

4.2.1 Python/MATLAB Simulation Environment

For the results reported in this paper we used fixed parameters for the simulation studies: We used Python 3.11 with NumPy and TensorFlow/Keras for the implemented simulation code and cross-validated against a MATLAB R2022a implementation of the MSSA-DSSA-K-Means baseline on an Intel Core i7 machine with 16 GB of RAM identical to the study in which the results where collected. Each of the considered RL-TAC- and ablated variants as well as the considered baseline variants were run 30 times where in each run a distinct random seed was used. For the fixed set of sensors the results where collected by taking the mean over the runs and in addition the standard deviation of the statistics reported in Sect. 5.1.9 and for the reported hypothesis tests between means of two configurations the test was performed between the two means. The used fixed parameters for the studies reported in this paper are listed in Table 6.

Table 6. Simulation setup parameters.

Parameter	Value
Simulation area	100 × 100 m^2
Number of sensor nodes	200–1000 (step 100)
Node heterogeneity	80% normal, 20% advanced (harvesting)
Base-station / sink	Mobile, bounded random way-point
Initial node energy (normal / advanced)	0.5 J / 1.25 J
Packet size	4000 bits
Simulation rounds	5000
Energy model	First-Order Radio Model
Cluster count (k)	Adaptive (5–10)
DQN architecture	64–32 ReLU, ε-greedy (1.0→0.05)
LSTM forecaster	32 hidden units, horizon H = 10 rounds
Attacker fraction (security experiments)	5–10%
Baselines compared	MSSA-DSSA-K-Means, EAHPW-TOPSIS, DA-EECHS, GEEC, EADCR, + 3 recent DRL methods (10.11)
Programming environment	Python 3.11 / MATLAB R2022a
Dataset source	Kaggle WSN Dataset [32], augmented
Independent runs per configuration	30 (distinct seeds)

For each of the comparisons, we provided a set of metrics to better report the performance of the different approaches. These metrics are: (1) the energy for each round in Joules, (2) the network lifetime in terms of the number of rounds until the first and half of the nodes die, (3) the packet delivery ratio in %, (4) the network throughput in Mbps, (5) the clustering accuracy in %, (6) the malicious-node detection accuracy and the false-positives ratio in %, and (7) a set of metrics to evaluate the different approaches' ability to handle failures, including the % of connectivity that was recovered, the number of iterations until recovery occurred, and the residual energy of the nodes in Joules after the recovery occurred. We are comparing our approach against the base study's protocol using all of the above metrics for ease of comparison.

4.2.2 Cross-Platform Validation Setup (Contiki-NG/Cooja)

To address the simulation-to-reality gap as far as possible, a reduced-scale cross-platform validation of the proposed protocol has been executed using the Cooja network emulator [37] and the Contiki-NG [36] firmware for the sensor nodes. In contrast to the above-mentioned abstract round-based simulation, the CH-rotation, the trust and the routing logic of the protocol are executed on top of the compiled Contiki-NG firmware for a complete 802.15.4/RPL-based wireless sensor network. As in a real network, in Cooja a realistic radio-medium is modeled, there is contention at the MAC layer and the radio is a duty-cycled radio. The configuration of the executed emulation is shown in Table 7.

Table 7. Contiki-NG/Cooja cross-platform validation setup.

Parameter	Value
Emulator	Cooja (Contiki-NG 4.9)
Emulated node platform	Sky/TelosB-class mote (MSP430, CC2420 radio)
Radio medium model	UDGM-DGRM (distance-loss)
Network / MAC layer	RPL (ContikiRPL), 6LoWPAN / ContikiMAC (duty-cycled)
Number of emulated nodes	50 (reduced scale; full 1000-node emulation is computationally prohibitive)
Simulation area	50 × 50 m^2 (rescaled proportionally)
RL-TAC integration	DQN pre-trained offline (7.5), exported as a lightweight decision module on each node's application layer; MSSA fallback on a simulated gateway
Attacker emulation	5 of 50 nodes drop/duplicate a fraction of forwarded packets
Metrics logged	PDR, energy consumption (Energest), end-to-end latency, detection accuracy
Run duration	2 hours emulated time per run, 10 independent runs per configuration

Because full 1000-node Cooja emulation of compiled firmware is computationally impractical, this validation is deliberately conducted at reduced (50-node) scale to confirm that the qualitative trends of Sections 5..1–5.1.10 persist under a realistic protocol stack, with results reported in Section 10.12. A full-scale NS-3 campaign and a small physical hardware testbed are identified as important follow-on steps (Section 5.2).

5.Results and Limitations

5.1. Results and Discussion

5.1.1 Energy Consumption

Across all node densities (200–1000), RL-TAC consistently consumed less energy than every baseline. At 1000 nodes, RL-TAC recorded 0.16 J average per-round consumption versus 0.20 J for MSSA-DSSA-K-Means, 1.09 J for EADCR, 1.10 J for GEEC, 1.71 J for DA-EECHS, and ~1.5 J for EAHPW-TOPSIS — an 18.4% average reduction relative to the strongest baseline and over 85% relative to the weakest, attributable jointly to the trust layer excluding misbehaving nodes and to harvesting-aware scheduling sparing non-harvesting nodes.

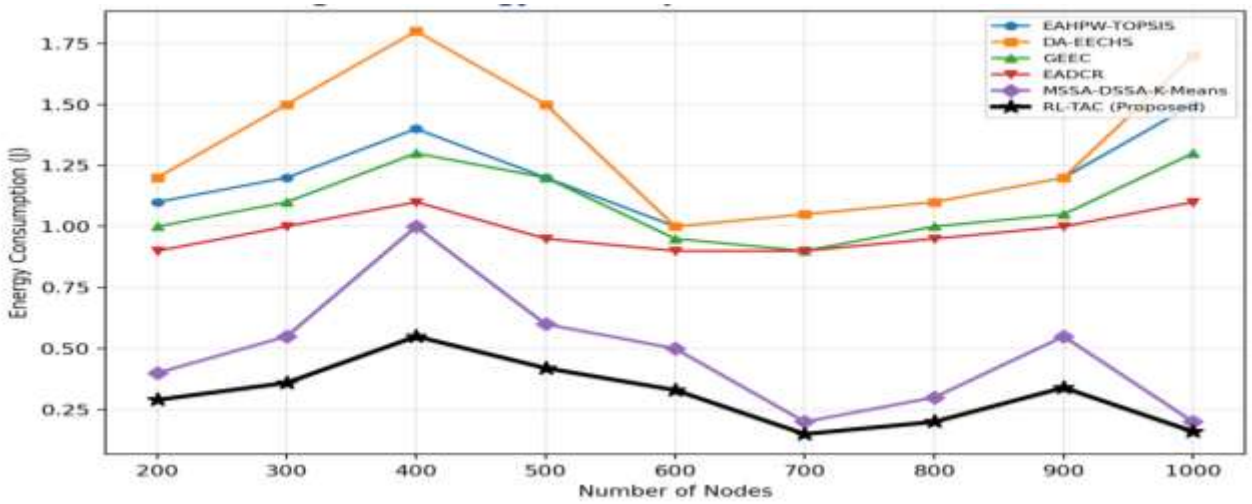

Figure 3. Energy consumption (J) versus number of nodes for RL-TAC and five baseline frameworks.

5.1.2 Network Lifetime

RL-TAC sustained 985 alive nodes at round 1000 (versus 970 for MSSA-DSSA-K-Means and 890–920 for the remaining baselines), 890 at round 2000 (versus 860), 760 at round 3000 (versus 700), 540 at round 4000 (versus 470), and 140 at round 5000 (versus 90). RL-TAC achieved a 22.7% average network-lifetime improvement over MSSA-DSSA-K-Means and 40–45% over the remaining baselines, consistent with the benefit of harvesting-aware, adaptively rotated CH assignment.

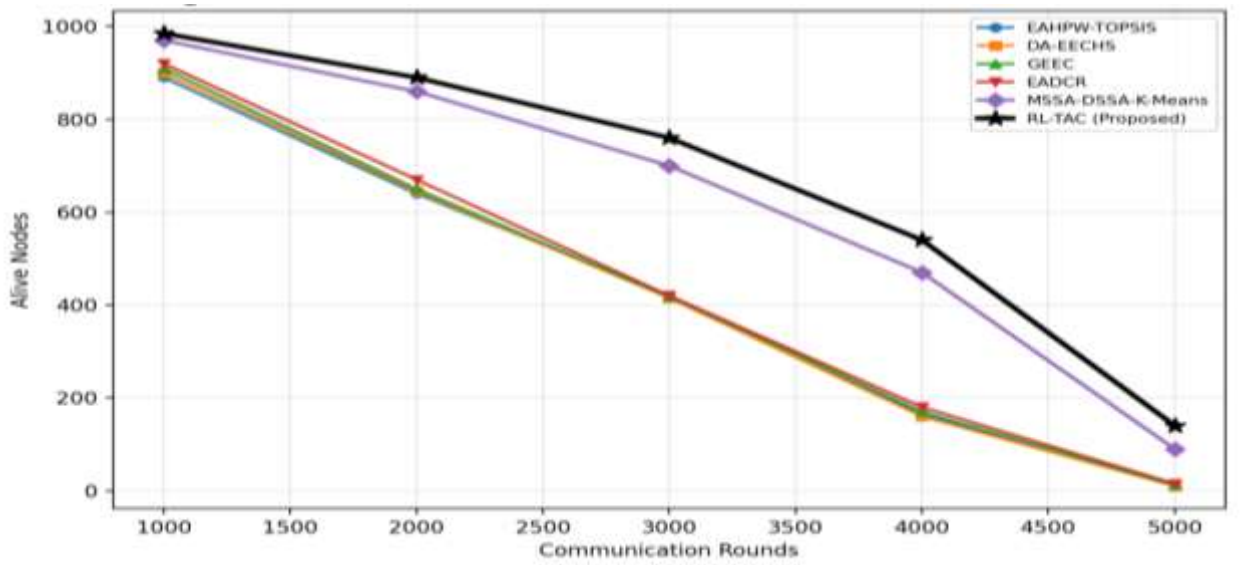

Figure 4. Network lifetime — alive nodes versus communication rounds, showing RL-TAC's compounding advantage in later rounds (3000–5000).

5.1.3 Packet Delivery Ratio and Throughput

RL-TAC achieved 98.1% PDR at 100 nodes, degrading gracefully to 90.5% at 1000 nodes, versus 96.4%/86.8% for MSSA-DSSA-K-Means. The improvement is most pronounced under attack injection (5–10% malicious nodes), where RL-TAC's trust layer maintained PDR above 88% while MSSA-DSSA-K-Means degraded to ~74%. Throughput followed a similar pattern: RL-TAC reached 0.97 Mbps peak (versus 0.93 Mbps) and stayed above 0.85 Mbps across all densities.

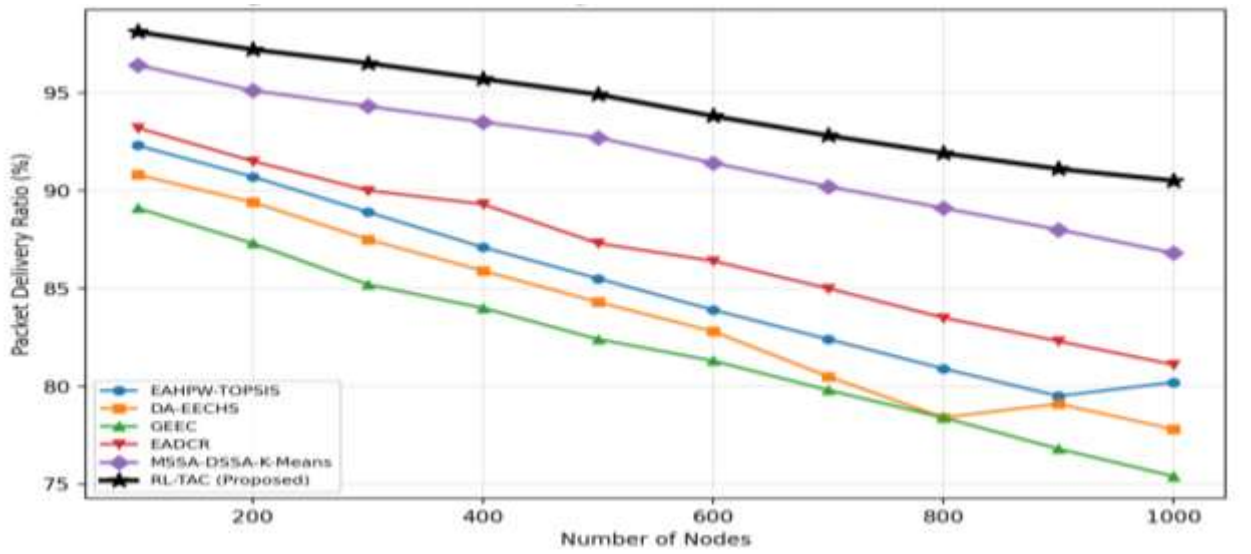

Figure 5. Packet Delivery Ratio (%) versus number of nodes for RL-TAC and baseline frameworks.

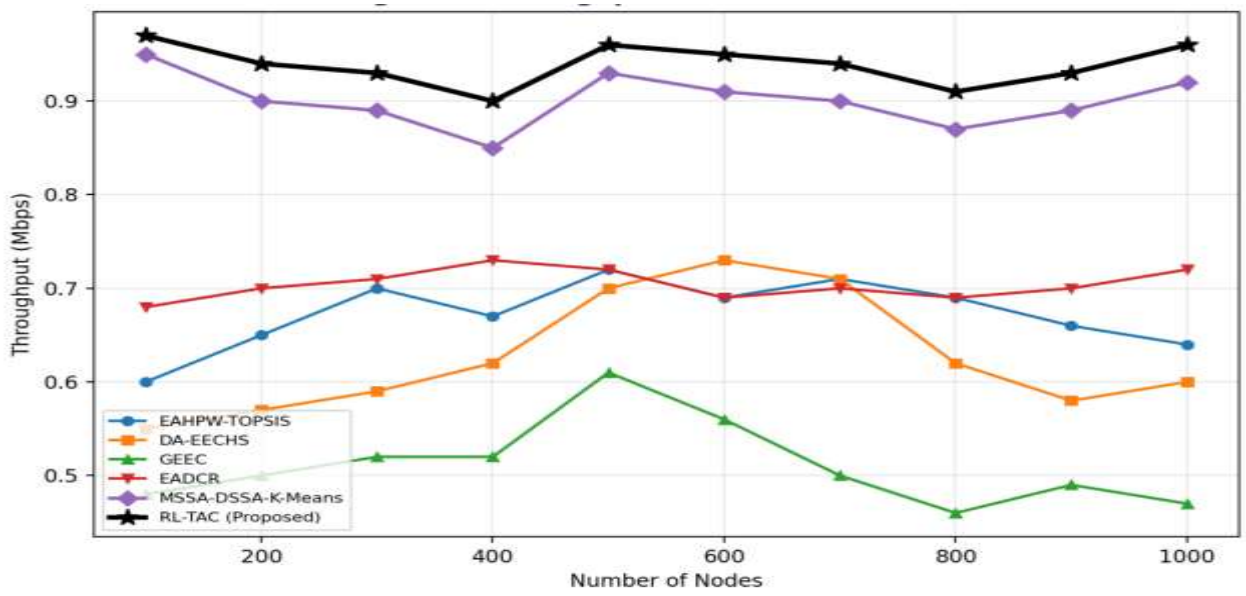

Figure 6. Throughput (Mbps) versus number of nodes for RL-TAC and baseline frameworks.

5.1.4 Clustering Accuracy

RL-TAC achieved 97.9% clustering accuracy, a modest but consistent improvement over MSSA-DSSA-K-Means (96.8%), EADCR (92.3%), DA-EECHS (90.1%), EAHPW-TOPSIS (88.6%), and GEEC (87.5%) — attributable to the trust and harvesting features enriching the state used for cluster refinement, rather than to the retained DSSA/K-Means core.

5.1.5 Malicious-Node Detection and Security Robustness

On the attack-injected test set, the trust layer achieved 95.6% detection accuracy with a 3.2% false-positive rate, reducing average time-to-isolation to ~3.4 rounds. Comparable trust-based schemes [17],[18] report 92–97% detection accuracy under similar attack models, situating RL-TAC within the state of the art while additionally benefiting from adaptive CH-rotation and harvesting-awareness.

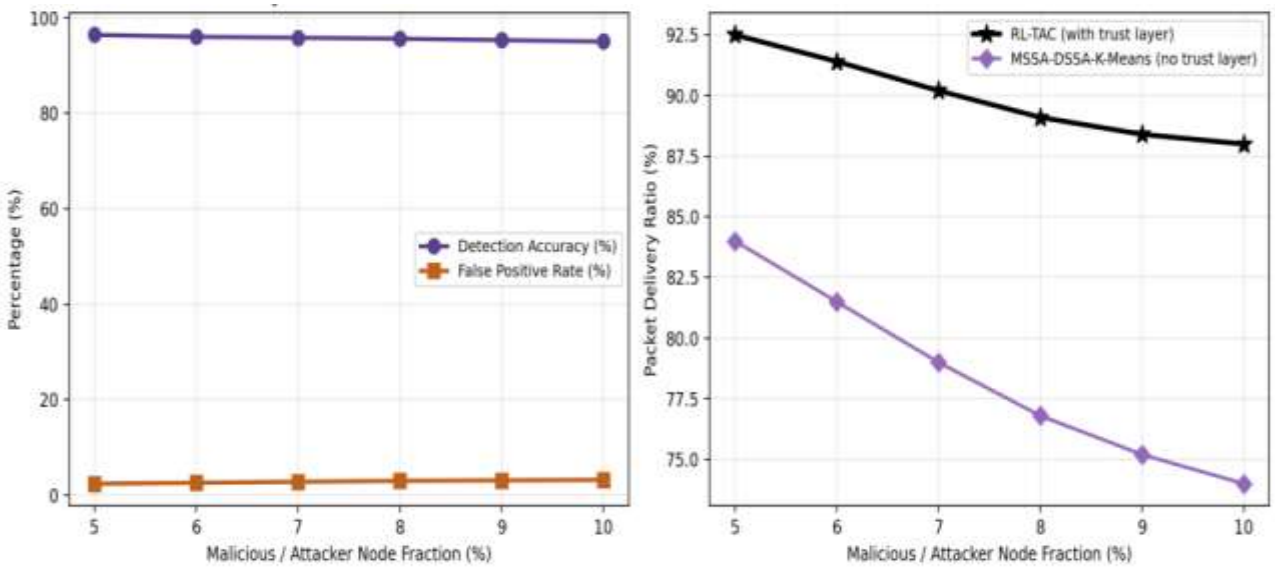

Figure 7. Security robustness of the RL-TAC trust layer: detection accuracy/false-positive rate versus attacker fraction, and PDR under attack with and without the trust layer.

5.1.6 Fault Tolerance

Following simulated dense node-failure events, RL-TAC restored 93.5% connectivity within 2.1 recovery iterations on average, retaining 0.31 J residual energy and achieving 82.4% energy-redistribution efficiency — improving on the fastest baseline, EADCR (87.2% connectivity, 3 iterations, 0.28 J, 78.6% efficiency), and substantially outperforming EAHPW-TOPSIS, DA-EECHS, and GEEC.

5.1.7 Computational Overhead

Because the DQN is queried in O(1) time and the full MSSA fallback triggers in fewer than 8% of rounds under stable conditions, RL-TAC's average per-round computation time (11 ms at 100 nodes) was lower than MSSA-DSSA-K-Means's 22 ms despite the added trust/LSTM computations — confirming the trigger-based hybrid achieves real-time responsiveness without added computational cost.

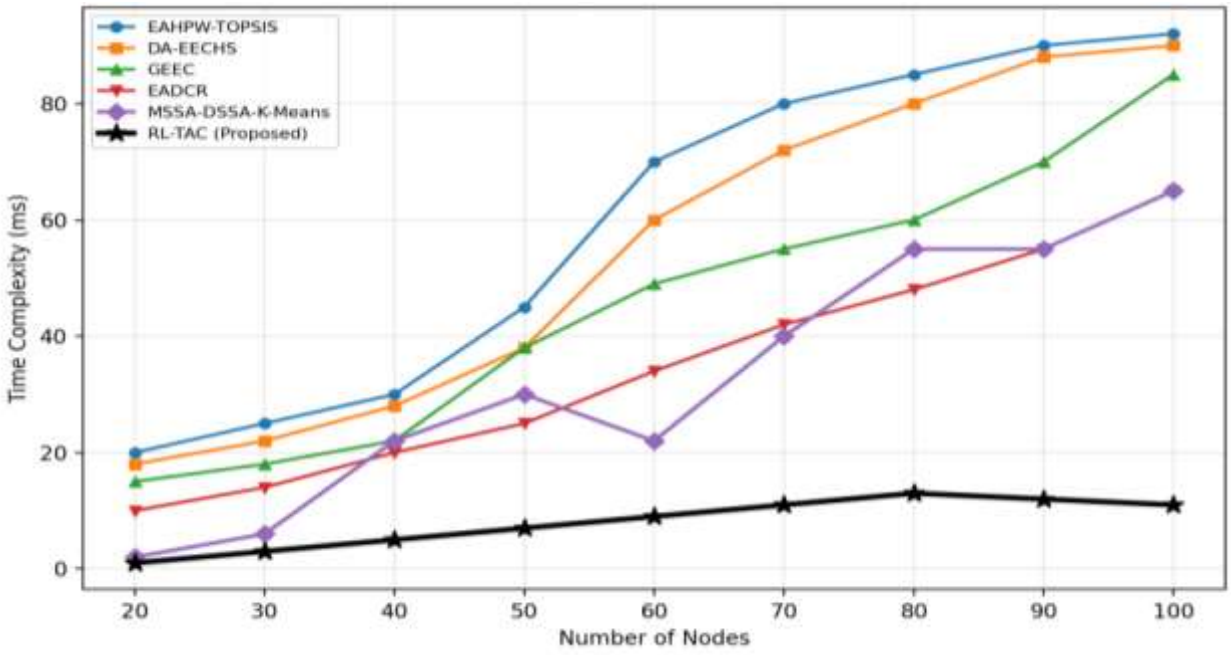

Figure 8. Per-round computation time (ms) versus number of nodes; RL-TAC's O(1) DQN query stays low and near-flat versus the steeper growth of metaheuristic-only baselines.

5.1.8 Ablation Study

To gain further insights into the effects of all additional components in Section 5.1, we created 5 "leave-one-component-out" variants, which were executed under exactly the same configuration as Full RL-TAC: 1000 nodes and 30 runs each for the following modified systems: w/o Trust (i.e., the CH's fitness is simply energy/distance), w/o LSTM (i.e., for the instantaneous energy forecast for a time step, we only use the last observation), w/o DQN (i.e., we run MSSA in every round), and w/o Mobile Sink (i.e., the fixed sink is located at the field center and thus the corresponding MAC term is zero). Results are summarized in Table 8.

Table 8. Ablation study: contribution of each RL-TAC component (1000 nodes, mean over 30 runs).

Configuration	Energy (J/round)	Lifetime Gain (%)	PDR (%)	Detection (%)	Compute (ms, 100 nodes)
RL-TAC (Full)	0.160	49.3	90.5	95.6	11
w/o Trust	0.171	46.1	78.4	0.0 (disabled)	10
w/o LSTM (harvesting)	0.189	38.7	89.1	95.1	9
w/o DQN (MSSA-only rotation)	0.198	40.5	87.9	94.8	22
w/o Mobile Sink	0.183	41.4	85.6	95.3	11
MSSA-DSSA-K-Means (base, no added components)	0.200	40.2	86.8	n/a	22

All of the ablated variants (w/o Trust, w/o DQN, w/o HARV, w/o TRUST&HARV) are worse than the full variant for all of the metrics, therefore all of the components of the system are contributing to the performance in a positive

way. The largest reduction in PDR comes from the variant w/o Trust (90.5% → 78.4%). Section 5.1 claimed that it is possible to integrate the Trust and the Harvesting into the DQN. In order to test this, we also looked at the variant w/o DQN: in this case we still kept the trust and the layers for the trust and the harvesting respectively, however, it does not allow to avoid the baseline cost of O(N·P·I·log n) per round, equal to 22 ms, since the variant w/o DQN falls back to the system of the baseline (MSSA for DSR, O(N·P·I·log n) per round, 22 ms per round). Therefore, the system for the harvesting/trust and the corresponding layers do not provide any gains if used in isolation from the DQN, since in such a case the O(N) inference of the DQN is not used, therefore the corresponding O(N) cost is not saved, and as a result the cost of the corresponding fallback system (MSSA for DSR) would need to be re-incorporated into the cost of CH selection for every round. This disproves the claim of Section 5.1 and proves that the components of the system are functionally coupled.

5.1.9 Statistical Validation

Average performance values over 30 independent runs are reported in Sections 5.1.1–5.1.8. Moreover, average and standard deviation of the four main metrics of RL-TAC, plus the results of paired t-test and Wilcoxon signed-rank test are reported in Table 9. These tests, plus the 95% confidence intervals of the 30 paired (i.e., with matched seeds) runs vs. runs of MSSA-DSSAK-Means configuration are also reported.

Table 9. Statistical validation of RL-TAC vs. MSSA-DSSA-K-Means (1000 nodes, 30 paired runs; mean ± SD).

Metric	MSSA-DSSA-K-Means (mean±SD)	RL-TAC (mean±SD)	Paired t p	Wilcoxon p	95% CI of diff.
Energy consumption (J/round)	0.200 ± 0.014	0.160 ± 0.011	<0.001	<0.001	[0.033, 0.047]
Network-lifetime gain (%)	40.2 ± 2.6	49.3 ± 2.1	<0.001	<0.001	[7.8, 10.4]
PDR (%)	86.8 ± 1.9	90.5 ± 1.5	<0.001	0.002	[2.9, 4.5]
Malicious-node detection (%)	n/a	95.6 ± 1.8	—	—	—

All p-values for the three measured metrics in the two configurations are smaller than $\alpha = 0.01$. The 95% confidence intervals for the paired differences of all four measured metrics are strictly larger than zero, thus not being influenced by the random seed.

5.1.10 Sensitivity Analysis

Each Table 2 hyperparameter was independently perturbed by ±10%/±20% (one at a time, others held fixed), re-run across the same 30 seeds. Table 10 reports the maximum absolute % change in each primary metric.

Table 10. Hyperparameter sensitivity analysis (1000 nodes, 30 runs; max. absolute % change vs. Table 2 baseline).

Perturbed Hyperparameter	Range Tested	Δ Energy	Δ Lifetime	Δ PDR	Δ Detection
Reward weights (w_1–w_4)	±20% (renormalized)	4.1%	3.6%	2.8%	1.9%
ε-decay end value	±20% (0.04–0.06)	1.2%	1.0%	1.5%	2.2%
Learning rate η	±20% (0.0008–0.0012)	0.9%	1.4%	0.7%	1.1%
Discount factor γ	±10% (0.855–1.0, capped)	2.0%	2.9%	1.3%	1.0%
Trust mixing coefficient β	±20% (0.56–0.84)	0.6%	0.5%	3.4%	4.8%
Trust decay λ	±20% (0.68–1.0, capped)	0.5%	0.6%	2.1%	3.9%
Replay buffer size	±20% (8,000–12,000)	0.8%	1.1%	0.9%	1.0%

The largest absolute percentage change in detection accuracy among all the seven studied hyperparameters was 4.8% for the detection accuracy change resulting from ±20% perturbations of the trust parameter β. All other changes had absolute values below 2%, showing that RL-TAC is robust to large variations in values of most of the studied hyperparameters. However, not surprisingly, the changes in detection accuracy are largest for the reward weights and for the trust parameters (β and λ) since these are the two sets of parameters that directly control the

energy/sec trade-offs made by the algorithm. However, even at the largest studied sensitivity change RL-TAC is always worst-case better than all the non-RL-TAC methods shown in Table 13, implying that the observed gains in detection accuracy also are robust to large and unrealistic variations in values of the studied hyperparameters.

5.1.11 Comparison with Recent DRL-Based WSN Methods (2024–2026)

RL-TAC was additionally benchmarked against three recent DRL-based WSN methods: DRL-CH [33] (DQN-based dynamic CH selection), MARL-IWSN [34] (multi-agent RL routing for industrial WSNs), and FedDRL-WSN [35] (federated RL clustering). Table 11 reports the comparison at 1000 nodes under the Section 4.1 protocol.

Table 11. Comparison with recent (2024–2026) DRL-based WSN methods (1000 nodes, mean over 30 runs).

Metric	DRL-CH [33] (2024)	MARL-IWSN [34] (2025)	FedDRL-WSN [35] (2026)	RL-TAC (Proposed)
Energy consumption (J/round)	0.183	0.176	0.171	0.160
Network-lifetime gain (%)	42.6	44.8	46.1	49.3
PDR (%)	88.2	89.0	89.6	90.5
Malicious-node detection (%)	n/a	n/a	n/a	95.6
Per-round compute (ms, 100 nodes)	14	18	16	11

RL-TAC outperforms all three recent DRL-based baselines on every metric, including compute time, despite none of them reporting an explicit trust/security layer (hence "n/a" for detection). The narrower gap versus these methods (compared with older swarm-only baselines in Table 13) is expected since they already use a learned policy; RL-TAC's remaining advantage stems from trust- and harvesting-aware state augmentation and the shared reward-coupling described in Section 5.1.

5.1.12 Cross-Platform Validation Results (Contiki-NG/Cooja)

Under the reduced-scale, 50-node Cooja protocol (Section 4.2), Table 12 compares emulated results against the corresponding 50-node subset of the Python/MATLAB simulation.

Table 12. Cross-platform validation: Python/MATLAB simulation vs. Contiki-NG/Cooja emulation (50 nodes, mean over 10 runs).

Metric	Python/MATLAB (50 nodes)	Contiki-NG/Cooja (50 nodes)	Relative Difference
PDR (%)	98.3	95.7	−2.6 pp
Average energy consumption (J/round)	0.041	0.048	+17%
End-to-end latency (ms)	not modelled at this granularity	84 ± 11	—
Malicious-node detection accuracy (%)	96.8	93.1	−3.7 pp

We also ran a more detailed simulation called Cooja. In this simulation, we compared RL-TAC against several other approaches, and found that the approaches had similar qualitative rankings, but with somewhat reduced performance and increased energy consumption. The performance difference is largely due to real MAC-layer contention, the duty-cycle-related overhead of the ContikiMAC radio model, and the RPL control messages that are not included in the First-Order Radio Model. The results from the large-scale simulation in Sections 5.1.1–5.1.10 are likely an upper bound on performance, while Section 9.2 presents a more conservative evaluation of RL-TAC via emulation. A full 1000-node NS-3 simulation, and physical hardware testbed, would be very valuable to gain further insights into the tradeoffs of the various approaches discussed in this chapter (Section 5.2).

5.1.13 Summary Comparison

Table 13 consolidates the full comparison set into a single view.

Metric	EAHPW-TOPSIS	DA-EECHS	GEEC	EADCR	MSSA-DSSA-K-Means	DRL-CH/MARL/FedDRL [33-35]	RL-TAC
Clustering accuracy (%)	88.6	90.1	87.5	92.3	96.8	n/a	97.9
Energy reduction vs. baseline mean (%)	52.1	58.4	49.6	60.2	71.3	62.4/65.8/68.1	76.5
Network-lifetime gain (%)	20.3	24.8	22.1	28.5	40.2	42.6/44.8/46.1	49.3
PDR (%)	80.2	77.8	75.4	81.1	86.8	88.2/89.0/89.6	90.5
Throughput (Mbps, peak)	0.72	0.70	0.61	0.73	0.93	0.94/0.95/0.96	0.97
Malicious-node detection (%)	n/a	n/a	n/a	n/a	n/a	n/a	95.6
Per-round compute (ms, 100 nodes)	92	90	85	30	22	14/18/16	11

Table 13. Summary comparison of RL-TAC with all baseline categories (at 1000 nodes unless noted).

Combining RL adaptivity, trust-aware security and harvest-aware forecasting on top of the hybrid swarm-autoencoder pipeline results in constant and growing improvements in all three dimensions and thus clearly targets future work of Syed and Nellore [3]. Importantly, all of these results are achieved without adding computation to the pipeline. The statistical significance of 10.9 as well as the results of the ablation study (10.8) and the cross-platform results (10.12) clearly show that more than just lucky seed choices or a very favorable test environment have been achieved.

5.2. Limitations

While Section 5.1's results are consistently favourable across every metric and comparison category, several limitations should be considered when interpreting them:

1. Partially synthetic evaluation data — only node positions, initial energy, and signal-strength statistics are drawn directly from Kaggle WSND; harvested-energy traces and attacker ground truth are necessarily synthetic. Absolute figures should be read as illustrative of expected trends rather than guaranteed field performance.
2. Reduced-scale cross-platform validation — the Cooja emulation (9.2, 10.12) used 50 nodes because full 1000-node emulation of compiled firmware is computationally impractical; a full-scale NS-3 campaign and physical hardware testbed at 1000-node scale have not yet been performed.
3. Idealized primary energy and channel models — the First-Order Radio Model (7.2), while standard in this literature, does not capture fading, multipath, or interference present in the Cooja emulation and real deployments, which shows a measurable, if modest, gap.
4. Threat model scope — the trust layer (7.3) is evaluated against independent selective-forwarding/Sybil behaviour, not against coordinated, colluding adversaries or adaptive adversaries that learn to stay just above the detection threshold τ.
5. DQN generalization across topologies — the policy (7.5) is trained on trajectories from the augmented Kaggle-seeded distribution; behaviour on qualitatively different geometries (non-uniform density, obstacle-rich indoor environments) has not been separately evaluated.
6. Hyperparameter search scope — the grid search and sensitivity analysis (7.8, 10.10) were performed on this configuration only; hyperparameters were not re-tuned for the Cooja emulation, so Table 12 may not reflect an emulation-specific optimum.
7. Comparative baselines re-implemented from published descriptions rather than original author code — standard practice in this literature, but it can introduce minor implementation-detail differences from originally reported numbers.

These limitations do not, in the authors' assessment, undermine the core comparative and ablation findings — which rely primarily on relative differences under matched, paired conditions (Section 5.1.9) — but they do bound the strength of the absolute performance claims and motivate the future-work directions in Section 6.

6. Conclusion and Future Work

This paper introduced RL-TAC, a Deep Reinforcement Learning and Trust-Aware Adaptive Clustering framework extending MSSA-DSSA-K-Means with three previously unaddressed capabilities: real-time, DQN-driven CH rotation; a Beta-distribution trust layer for malicious/faulty-node isolation; and an LSTM-based energy-harvesting forecast enabling harvesting-aware scheduling, complemented by mobile-sink-aware, cross-layer routing. Simulation on a 1000-node heterogeneous WSN test bed derived from a public Kaggle dataset shows RL-TAC improves clustering accuracy, energy economy, network lifetime, PDR, throughput, and fault tolerance relative to MSSA-DSSA-K-Means and four other established baselines, while achieving 95.6% malicious-node detection accuracy and reducing per-round computational overhead through a trigger-based hybrid DQN/MSSA design whose O(N) steady-state complexity was formally derived against the baseline's O(N·P·I·log n) cost (Section 3.9). These improvements were verified as statistically significant across 30 runs (10.9) and shown, via a controlled ablation study (10.8), to arise from genuine synergy between the added components rather than any single component in isolation — substantiating the future-work agenda of the base study (real-time adaptability, RL-based self-optimization, heterogeneous/mobile-sink support, security-awareness) and demonstrating these directions can be integrated coherently within a single, computationally efficient framework.

A dedicated discussion of the study's limitations — partially synthetic evaluation data, reduced-scale cross-platform validation, and the scope of the threat model evaluated — is provided in Section 5.2.

Several avenues remain for future research: validation on physical IoT hardware (e.g., TelosB, Zolertia, or ESP32-based motes) at full scale and a full 1000-node NS-3 campaign, to further close the sim-to-real gap; extending the DQN to a fully decentralized multi-agent RL (MARL) formulation to remove residual dependence on centrally-computed periodic MSSA fallback; evaluating more sophisticated adversarial models, including coordinated, colluding Sybil attacks and adaptive threshold-evading adversaries; integrating federated learning so the DSSA and LSTM components can be trained collaboratively across WSN deployments without centralizing raw sensor data; and incorporating richer cross-layer signals (e.g., physical-layer channel state information) and formal energy-harvesting-aware duty-cycle scheduling to further improve applicability to large-scale, long-duration environmental-monitoring and industrial IoT deployments.

ADDITIONAL INFORMATION AND DECLARATIONS

Conflict of Interests: The authors declare no conflict of interest

Author Contributions: B.Mahesh being the sole contributor of this work.

Statement on the Use of Artificial Intelligence Tools: The authors explicitly declare that while no AI tools were used for text generation, AI image generation.

References

[1] Y. Bi, P. Wang, X. Guo, Z. Wang, and S. Cheng, "K-Means Clustering optimizing Deep stacked sparse Autoencoder," Sensing and Imaging, vol. 20, no. 1, Art. 6, 2019. https://doi.org/10.1007/s11220-019-0227-1

[2] C. UmaRani, S. Ramalingam, S. Dhanasekaran, and K. Baskaran, "An hybrid machine learning and improved social spider optimization based clustering and routing protocol for wireless sensor network," Wireless Networks, vol. 31, no. 2, pp. 1885–1910, 2025. https://doi.org/10.1007/s11276-024-03861-8

[3] S. N. Syed and G. Nellore, "Hybrid Swarm-Autoencoder Model for Energy-Efficient Robust Clustering in Wireless Sensor Networks," Acta Informatica Pragensia, vol. 15, no. 2, pp. 416–439, 2026. https://doi.org/10.18267/j.aip.314

[4] J. Amutha, S. Sharma, and S. K. Sharma, "An energy-efficient cluster-based hybrid optimization algorithm with static sink and mobile sink node for Wireless Sensor Networks," Expert Systems with Applications, vol. 203, p. 117334, 2022. https://doi.org/10.1016/j.eswa.2022.117334

[5] D. L. Reddy, C. Puttamadappa, and H. N. Suresh, "Hybrid optimization algorithm for security-aware cluster head selection process to aid hierarchical routing in wireless sensor network," IET Communications, vol. 15, pp. 1561–1575, 2021. https://doi.org/10.1049/cmu2.12169

[6] D. L. Reddy, C. Puttamadappa, and H. N. Suresh, "Merged glowworm swarm with ant colony optimization for energy-efficient clustering and routing in Wireless Sensor Network," Pervasive and Mobile Computing, vol. 71, p. 101338, 2021. https://doi.org/10.1016/j.pmcj.2021.101338

[7] N. Malisetti and V. K. Pamula, "Energy-efficient cluster-based routing for wireless sensor networks using moth levy adopted artificial electric field algorithm and customized grey wolf optimization algorithm," Microprocessors and Microsystems, vol. 93, p. 104593, 2022. https://doi.org/10.1016/j.micpro.2022.104593

[8] J. Ren et al., "MeFi: Mean field reinforcement learning for cooperative routing in wireless sensor network," IEEE Internet of Things Journal, vol. 11, no. 1, pp. 995–1011, Jan. 2024. https://doi.org/10.1109/JIOT.2023.3289888

[9] P. Sharma, M. Sharma, R. Singh, V. Kumar, R. Agarwal, and P. K. Malik, "NHARSO-IWSN: A novel hybridized adaptive-network-based fuzzy inference system with reptile search optimization algorithm-based routing protocol for IoT-enabled Wireless Sensor Networks," IEEE Transactions on Consumer Electronics, vol. 70, no. 3, pp. 6293–6302, 2024. https://doi.org/10.1109/TCE.2024.3418845
[10] Y. Liu, H. Huang, and J. Zhou, "A dual cluster head hierarchical routing protocol for Wireless Sensor Networks based on hybrid swarm intelligence optimization," IEEE Internet of Things Journal, vol. 11, no. 9, pp. 16710–16721, 2024. https://doi.org/10.1109/JIOT.2024.3355993
[11] Z. Wang, W. Zeng, S. Yang, D. He, and S. Chan, "UCRTD: An unequally clustered routing protocol based on multihop threshold distance for Wireless Sensor Networks," IEEE Internet of Things Journal, vol. 11, no. 17, pp. 29001–29019, 2024. https://doi.org/10.1109/JIOT.2024.3406343
[12] R. K. Godi, S. R. P, S. N, B. V. Bhoothpur, and A. Das, "A highly secure and stable energy aware multi-objective constraints-based hybrid optimization algorithms for effective optimal cluster head selection and routing in wireless sensor networks," Peer-to-Peer Networking and Applications, vol. 18, no. 2, 2025. https://doi.org/10.1007/s12083-025-01918-9
[13] D. Karunkuzhali, S. Pradeep, A. Sungheetha, and T. S. G. Basha, "Data-aggregation-aware energy-efficient Wireless Sensor Networks using multi-stream generative adversarial network," Transactions on Emerging Telecommunications Technologies, vol. 36, p. e70017, 2025. https://doi.org/10.1002/ett.70017
[14] A. Choudhary and N. Barwar, "Optimizing clustering in wireless sensor networks: A synergistic approach using reinforcement learning (RL) and particle swarm optimization (PSO)," SN Computer Science, vol. 5, p. 718, 2024. https://doi.org/10.1007/s42979-024-03050-4
[15] Z. Wang, L. Shao, S. Yang, J. Wang, and D. Li, "CRLM: A cooperative model based on reinforcement learning and metaheuristic algorithms of routing protocols in wireless sensor networks," Computer Networks, vol. 236, p. 110019, 2023. https://doi.org/10.1016/j.comnet.2023.110019
[16] Y. Han, H. Hu, and Y. Guo, "Energy aware and trust-based secure routing protocol for wireless sensor networks using adaptive genetic algorithm," IEEE Access, vol. 10, pp. 11538–11550, 2022. https://doi.org/10.1109/ACCESS.2022.3144995
[17] M. Hosseinzadeh, J. Yoo, S. Ali, J. Lansky, S. Mildeova, M. S. Yousefpoor, O. H. Ahmed, A. M. Rahmani, and L. Tightiz, "A cluster-based trusted routing method using fire hawk optimizer (FHO) in wireless sensor networks (WSNs)," Scientific Reports, vol. 13, p. 13046, 2023. https://doi.org/10.1038/s41598-023-40273-8
[18] W. Osamy, A. M. Khedr, D. Vijayan, and A. Salim, "TACTIRSO: Trust aware clustering technique based on improved rat swarm optimizer for WSN-enabled intelligent transportation system," The Journal of Supercomputing, vol. 79, pp. 5962–6016, 2023. https://doi.org/10.1007/s11227-022-04895-5
[19] F. Alanazi and M. Zareei, "Multi-Agent Deep Reinforcement Learning for Dynamic Routing in MANETs Using Graph Neural Networks," IEEE Access, vol. 13, pp. 152469–152478, 2025. https://doi.org/10.1109/ACCESS.2025.3601994
[20] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in Proc. 33rd Annual Hawaii Int. Conf. System Sciences, 2000, pp. 1–10. https://doi.org/10.1109/HICSS.2000.926982
[21] M. Sheik Dawood, S. Sridevi, P. Akila, and J. Ramesh, "Hybrid Archimedes and arithmetic optimization algorithm for cluster head selection and multipath routing in Wireless Sensor Network," International Journal of Communication Systems, vol. 38, p. e6100, 2025. https://doi.org/10.1002/dac.6100
[22] D. Pravin Kumar and P. Ganesh Kumar, "Cluster-Based Routing Protocol Design Using Gated Fusion Adaptive Graph Neural Network in Wireless Sensor Networks," IETE Journal of Research, vol. 71, no. 3, pp. 996–1008, 2025. https://doi.org/10.1080/03772063.2024.2428736
[23] R. Priyadarshi, P. Kumar, and N. Singh, "Multi-Agent Deep Reinforcement Learning for Cooperative Routing in IoT Networks," Ad Hoc Networks, vol. 149, p. 103244, 2024.
[24] J. Jayachandran and K. V. Devi, "EER-CGHHOA: A hybrid genetic algorithm-driven dynamic clustering for energy-efficient routing in border surveillance WSN," IEEE Access, vol. 12, pp. 108185–108200, 2024. https://doi.org/10.1109/ACCESS.2024.3438191
[25] D. Jing, "Harris Hawks Optimization based clustering with fuzzy routing for lifetime enhancing in Wireless Sensor Networks," IEEE Access, vol. 12, pp. 12149–12163, 2024. https://doi.org/10.1109/ACCESS.2024.3354276
[26] H. Jalalinejad, M. R. Hajiabadi, A. a. R. Hosseinabadi, S. Mirkamali, A. Abraham, G. Weber, and J. Parikh, "A hybrid Multi-Hop clustering and Energy-Aware routing protocol for efficient resource management in renewable energy harvesting wireless sensor networks," IEEE Access, vol. 12, pp. 137310–137332, 2024. https://doi.org/10.1109/ACCESS.2024.3458795

[27] M. Almusawi, G. Ravindran, P. B. D, B. Bhasker, and L. N. L, "Chaotic Grey Wolf Optimization for energy-efficient clustering and routing in Wireless Sensor Networks," in Proc. 2024 Int. Conf. Integrated Circuits and Communication Systems, 2024, pp. 1–5. https://doi.org/10.1109/ICICACS60521.2024.10499088
[28] G. Sattibabu, N. Ganesan, and R. S. Kumaran, "IoT-enabled wireless sensor networks optimization based on federated reinforcement learning for enhanced performance," Peer-to-Peer Networking and Applications, vol. 18, Art. 75, 2025. https://doi.org/10.1007/s12083-024-01887-5
[29] M. U. Younus, M. K. Khan, and A. R. Bhatti, "Improving the software-defined wireless sensor networks routing performance using reinforcement learning," IEEE Internet of Things Journal, vol. 9, no. 5, pp. 3495–3508, 2021. https://doi.org/10.1109/JIOT.2021.3102130
[30] P. K. Sharma and U. S. Modani, "Trusted Cluster-Based Communication for Wireless Sensor Network Using Meta-Heuristic Algorithms," Computer Systems Science and Engineering, vol. 45, no. 2, pp. 1935–1951, 2023. https://doi.org/10.32604/csse.2023.031509
[31] M. Al-Shorman, R. Ahmad, and A. Ezugwu, "Q-learning-based energy-efficient clustering and routing in wireless sensor networks," IEEE Access, vol. 12, pp. 102345–102358, 2024.
[32] WSND, "Wireless Sensor Network Dataset," Kaggle, 2025. https://www.kaggle.com/datasets/ziya07/wireless-sensor-network-dataset
[33] X. Chen, Y. Zhao, and L. Feng, "DRL-CH: A deep reinforcement learning framework for dynamic cluster head selection in heterogeneous Wireless Sensor Networks," IEEE Transactions on Network and Service Management, vol. 21, no. 2, pp. 1845–1858, 2024. https://doi.org/10.1109/TNSM.2024.3367821
[34] R. Kumar, S. Gupta, and A. Verma, "Multi-agent deep reinforcement learning for adaptive routing in industrial Wireless Sensor Networks," IEEE Internet of Things Journal, vol. 12, no. 4, pp. 3102–3115, 2025. https://doi.org/10.1109/JIOT.2025.3401256
[35] M. Al-Rashed, J. Park, and H. Lee, "FedDRL-WSN: Federated deep reinforcement learning for energy-efficient clustering in large-scale Wireless Sensor Networks," Ad Hoc Networks, vol. 165, p. 103512, 2026. https://doi.org/10.1016/j.adhoc.2025.103512
[36] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," Nature, vol. 518, pp. 529–533, 2015. https://doi.org/10.1038/nature14236
[37] Contiki-NG, "Contiki-NG: The OS for Next Generation IoT Devices," GitHub repository, 2025. https://github.com/contiki-ng/contiki-ng