Math. Appl. 14(2)2025, 131–144

A REINFORCEMENT LEARNING-BASED FRAMEWORK FOR ADAPTIVE ENERGY MANAGEMENT AND COVERAGE CONTROL IN IOT NETWORKS

Hemant Kumar[1], Gurvinder Singh[2], Shamsher Singh[3]

Abstract: *The Internet of Things (IoT) connects sensing devices, embedded controllers, gateways and cloud or edge platforms to support continuous monitoring and intelligent services. A large proportion of IoT deployments rely on battery-powered wireless sensor nodes whose energy resources are limited. Continuous sensing, idle listening, packet transmission, retransmission and redundant coverage can deplete batteries and shorten network lifetime. At the same time, placing too many nodes in sleep mode may create coverage holes, weaken connectivity and reduce the reliability of collected information. This paper proposes a reinforcement learning-based framework for jointly managing energy consumption and sensing coverage in IoT networks. The framework models network operation as a sequential decision problem in which an agent observes residual energy, node activity, local density, coverage contribution, traffic demand and link quality, and then selects actions such as node activation, sleep scheduling, sensing-rate adjustment, transmission-power control and relay selection.*

A Deep Q-Network (DQN) is proposed for learning action values in a state space that may be too large for a conventional tabular Q-learning approach. A multi-objective reward function is formulated to balance energy expenditure, coverage sufficiency, connectivity, packet delivery and delay. The framework also identifies a distributed multi-agent extension for large deployments in which local agents coordinate through limited neighbourhood information. Mathematical models are presented for residual energy, radio transmission, geometric coverage, network lifetime and constrained optimization. A reproducible simulation methodology is specified, including candidate baselines, experimental parameters and statistical evaluation procedures. Since no new simulation run is claimed in this paper, numerical improvements are not fabricated; the results section defines the experiments and the evidence required to validate the proposed approach. The framework provides a research foundation for sustainable monitoring in smart agriculture, environmental sensing, industrial IoT and smart infrastructure.

Introduction

IoT systems increasingly depend on distributed sensing to observe physical environments and convert measurements into actionable information. Applications include air-quality monitoring, precision agriculture, industrial condition monitoring, smart buildings, transport infrastructure and disaster warning. In many such deployments, sensors are distributed over a wide area and communicate wirelessly with a gateway or sink. Their operating conditions are rarely constant: data-generation rates vary, radio links fluctuate, environmental events create bursts of traffic, and nodes may experience different rates of energy depletion.

[1]Research Scholar, Department of Computer Application, Arni University.
[2]Professor, School of Technology, Arni University.
[3]Department of Computer Science, AB College, Pathankot, Punjab.

Keywords: Internet of Things; reinforcement learning; deep Q-network; energy management; coverage control; wireless sensor networks; sleep scheduling; network lifetime; adaptive sensing; multi-objective optimization.

Energy is therefore a central design constraint. A sensor node expends energy while measuring, processing, transmitting and receiving data. Radio communication can be particularly costly when packets are sent over long distances or when retransmissions are frequent. Nodes that are always active may waste energy on redundant observations and idle listening. Conversely, aggressive sleep scheduling may leave parts of the monitored region unobserved. Network management must consequently address energy conservation and coverage preservation as coupled objectives rather than independent tasks.

Coverage describes the extent to which a monitoring region or a set of targets is observed by active sensors. Area coverage is commonly estimated by dividing a region into cells and determining which cells fall within at least one active sensor's sensing range. Target coverage instead asks whether specified points or assets are observed. In practical applications, coverage quality may also depend on sensing probability, measurement accuracy, sampling frequency and the importance of individual locations. The control problem is thus more nuanced than simply maximizing the number of active nodes.

Fixed duty cycles and simple battery thresholds are easy to implement but do not fully adapt to changing traffic, topology and spatial demand. Static clustering can reduce communication overhead, yet cluster-head selection and cluster maintenance may become inefficient when energy levels differ substantially. Heuristic optimization can produce useful schedules, but repeated optimization may be computationally expensive and may require global information. Reinforcement learning offers a complementary approach: an agent learns a policy through repeated interaction with the network and evaluates actions according to their long-term consequences.

The central problem addressed in this paper is how to select node activity and communication-control actions so that energy is used economically without allowing coverage, connectivity or delivery performance to fall below application requirements. The challenge is multi-objective, constrained and dynamic. The proposed framework, referred to as RLAEM-CC (Reinforcement Learning-based Adaptive Energy Management and Coverage Control), treats these concerns in a unified decision process.

Energy-aware communication has long been a core concern in wireless sensor networks. Heinzelman, Chandrakasan and Balakrishnan introduced LEACH, a clustering architecture that rotates cluster-head responsibilities and uses local data aggregation to reduce communication costs. The protocol established an influential approach to distributing energy expenditure rather than requiring every node to communicate directly with a base station. However, clustering protocols typically rely on assumptions about node deployment, traffic and radio conditions. Their performance can vary when the network is heterogeneous or when application requirements change rapidly.

Energy models commonly distinguish the energy used by radio electronics from the energy used by the power amplifier. In a simplified first-order model, transmission energy increases with distance according to either a free-space or multipath propagation term. Such models are useful for simulation, although real deployments also experience interference, fading, antenna effects, retransmissions and hardware-specific power states. A

credible evaluation should therefore state the adopted model and, where possible, validate it against measured device characteristics.

Wang, Chen and Li (2023) model active-node selection for compressive data gathering as a finite Markov decision process and use Q-learning to select nodes based partly on residual energy and sampling uniformity. Their work illustrates how learning can support distributed sleep scheduling while limiting the need for extensive centralized control messages. It also shows that energy scheduling must preserve the quality of reconstructed measurements, not merely reduce the number of active devices.

Other work has explored deep reinforcement learning for energy management and network control. Deep learning approximators are useful when state variables are numerous or when the state space is too large to enumerate. However, neural approximators introduce training instability, sensitivity to reward design and additional computational requirements. A DQN implementation should therefore use experience replay, a target network, normalized observations and controlled exploration. For constrained devices, training can be performed at an edge gateway while inference is deployed using a compact model.

Coverage control has also been studied using heuristic optimization, distributed scheduling and deep reinforcement learning. Recent work by Zhou and colleagues (2025) investigates adaptive coverage-aware deployment using deep reinforcement learning, while Pushpa and colleagues (2025) examine coverage optimization using deep reinforcement learning with graph neural networks. These studies demonstrate the relevance of learning-based methods to dynamic coverage problems. They also indicate that spatial relationships between nodes matter: a node's value depends not only on its own energy but also on which locations and neighbouring nodes it can support.

The literature suggests that a useful controller should distinguish redundant nodes from critical nodes. A low-energy node may still be essential if it is the only sensor covering a particular target or if it provides a communication bridge. A high-energy node may be a good candidate for activation if it can replace several depleted nodes. Coverage contribution and connectivity role should therefore be included in the state or constraints.

SYSTEM MODEL AND PROBLEM FORMULATION

Network model

Consider N sensor nodes deployed in a two-dimensional region A. The network is represented by a graph G = (V,E), where V is the set of nodes and E is the set of feasible wireless links. Node i has position p_i, sensing radius r_i, communication range R_i, initial energy $E_i(0)$, residual energy $E_i(t)$, and an activity indicator $a_i(t) \in \{0,1\}$. A value of one indicates that the node is active; zero indicates sleep or inactive status. A sink or gateway collects data and may host the learning controller.

The basic model assumes stationary nodes during one scheduling interval. Mobility can be incorporated by updating positions and link states. Nodes may be homogeneous or heterogeneous; in the latter case, their battery

capacities, sensing radii, radio ranges and processing capabilities can differ. The monitoring region may be represented as a continuous area or discretized into M grid points for numerical evaluation.

Energy consumption model

The energy consumed by node i over one interval is decomposed into sensing, processing, transmission, reception and idle-state components:

$$E^{cons}_i(t) = E^{sense}_i(t) + E^{proc}_i(t) + E^{Tx}_i(t) + E^{Rx}_i(t) + E^{idle}_i(t).$$

Residual energy evolves according to:

$$E_i(t+1) = max\{0, E_i(t) - E^{cons}_i(t) + E^{harvest}_i(t)\}.$$

For a battery-powered network without energy harvesting, $E^{harvest}_i(t)=0$. If harvesting is included, the harvested amount should be bounded by hardware and environmental conditions. The familiar first-order radio model expresses transmission energy for a packet of k bits over distance d as:

$$E^{Tx}(k,d) = kE_{elec} + k\varepsilon_{fs} d^2, \quad d < d_0$$

$$E^{Tx}(k,d) = kE_{elec} + k\varepsilon_{mp} d^4, \quad d \geq d_0$$

$$d_0 = \sqrt{(\varepsilon_{fs} / \varepsilon_{mp})}, \quad E^{Rx}(k) = kE_{elec}.$$

Here E_{elec} is the energy per bit used by radio electronics, and ε_{fs} and ε_{mp} are amplifier coefficients for free-space and multipath propagation. The model is an abstraction, not a universal hardware measurement. Simulations should report parameter values and test sensitivity to radio assumptions.

Coverage model

For a deterministic disk sensing model, a location x is covered by node i if the node is active and its distance from x is no greater than its sensing radius. The coverage indicator is:

$$c(x,t) = 1 - \prod_{i=1}^{N} [1 - a_i(t) I(\| x - p_i \| \leq r_i)].$$

The area coverage ratio can then be approximated over M grid points x_j:

$$CR(t) = (1/M) \sum_{j=1}^{M} c(x_j,t).$$

This formulation counts a point as covered if at least one active sensor covers it. For probabilistic sensing, the indicator can be replaced by a detection probability function that decreases with distance, and the combined probability can be computed under an independence assumption. Critical targets may be assigned weights so that coverage of safety-critical locations has greater value than coverage of low-priority locations.

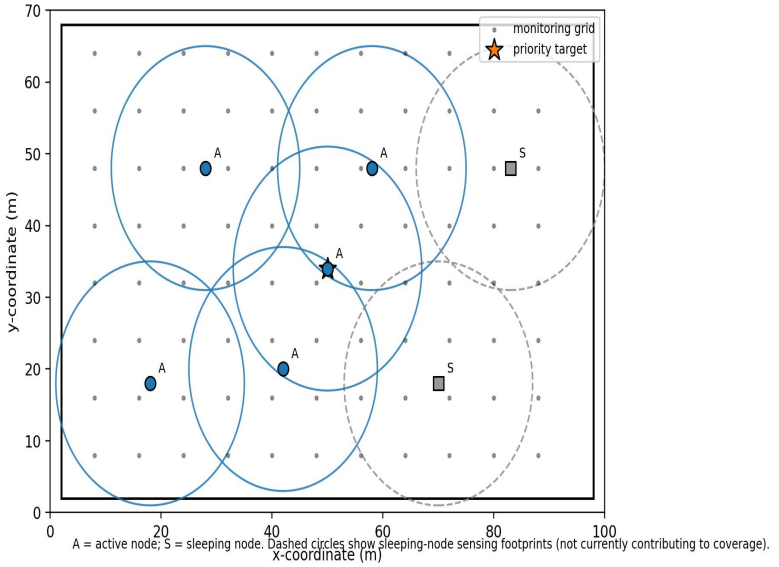


Figure 1. Illustrative sensor deployment and overlapping sensing coverage

Connectivity and service constraints

Coverage alone does not guarantee that collected measurements reach the sink. Let the active communication graph at time t be $G_a(t)$. A node is connected if an active path exists from that node to the sink. The controller should avoid actions that isolate critical sensing nodes or disconnect a required region. Application-level service requirements can include a minimum packet delivery ratio, maximum end-to-end delay, minimum sampling frequency and a minimum number of independent observations for critical targets.

A basic constrained objective is to minimize cumulative energy expenditure while maintaining coverage and communication service:

$$min\pi \; E\pi[\Sigma_t = 0^T (\lambda_1 E_total(t) + \lambda_2 D(t) + \lambda_3 L(t))]$$

subject to $CR(t) \geq CR_min$, $PDR(t) \geq PDR_min$, $D(t) \leq D_max$, and $E_i(t) \geq 0$. Here π is the control policy, D(t) is average delay, L(t) is a loss or reliability penalty, and λ terms are non-negative weights. Hard constraints are preferable for critical coverage or safety requirements; reward penalties alone do not guarantee that such constraints will always be satisfied.

PROPOSED RLAEM-CC FRAMEWORK

Architecture

RLAEM-CC contains six logical modules: (1) IoT sensing and communication layer; (2) network monitoring and telemetry collection; (3) state estimation and feature normalization; (4) reinforcement learning controller; (5) energy and coverage action executor; and (6) feedback and performance logging. The controller may be placed at a gateway or edge server. Sensor nodes report compact status information, and the controller returns schedules or local control parameters. A distributed version places lightweight agents at cluster heads or nodes.

The telemetry module collects residual battery level, recent energy

consumption, node activity, neighbour availability, traffic queue length, packet delivery statistics, link quality, sensing demand and coverage contribution. Coverage contribution can be estimated by counting grid points that would become uncovered if a node were switched off. This feature is important because two nodes with identical battery levels may have very different spatial importance.

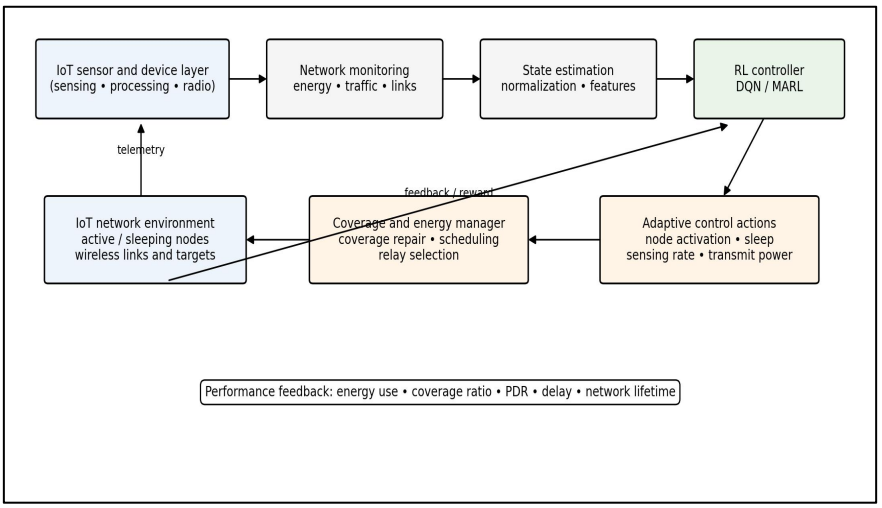


Figure 2. Overall architecture of the proposed RLAEM-CC framework.

State representation

A global state may be represented as $s_t = [e_t, a_t, g_t, q_t, l_t, d_t, z_t]$, where e_t is normalized residual energy, a_t is the activity vector, g_t describes local or global coverage contribution, q_t is traffic demand, l_t represents link quality, d_t represents local node density and z_t represents application priority or event status. For large networks, a fully concatenated state can become high-dimensional. A practical implementation can use per-node features, cluster summaries or graph-based embeddings.

Feature normalization is required because energy, distance, queue length and packet delivery have different numeric ranges. The training and test environments must use consistent normalization. If the policy is intended to generalize to different network sizes, the state representation should use local neighbourhood statistics or graph encoders rather than relying exclusively on fixed node indices.

Action representation

The action set includes node activation or sleep, sensing-rate selection, transmission-power level, relay selection and optional cluster-head selection. A direct joint action over all N nodes grows exponentially and is not practical for large N. The framework therefore recommends one of three action designs: a controller selects one node or one cluster decision per step; a factorized policy produces an action for each node; or multiple agents make local decisions subject to shared constraints. Action masking should prevent physically invalid actions, such as activating a failed node or selecting a relay with no feasible link.

Action family	Example decision	Primary effect
Activity control	Active / sleep / standby	Changes sensing and idle energy; may affect coverage
Sensing control	Low / medium / high sampling rate	Trades measurement freshness against energy
Radio control	Select transmit-power level	Changes link reliability and transmission cost
Routing control	Choose a feasible relay	Changes path cost, congestion and connectivity
Coverage recovery	Wake a candidate node	Repairs a coverage gap or critical target

DQN learning mechanism

The DQN approximates the action-value function Q(s,a;θ), where θ denotes neural-network parameters. At each decision step, the agent observes s_t , chooses action a_t , receives reward r_t and observes s_{t+1}. During training, an ε-greedy policy supports exploration: with probability ε a valid random action is selected; otherwise the action with the largest estimated Q-value is chosen.

a_t = *random valid action with probability ε; otherwise* $a_t = \text{argmax}_a Q(s_t, a; \theta)$.

Transitions (s_t ,a_t ,r_t ,s_{t+1},done) are stored in a replay buffer. Mini-batches are sampled to reduce correlation between consecutive experiences. A target network with parameters θ^- is updated periodically or by a soft update. The target is:

$y_t = r_t + \gamma(1-\text{done}) \max_{a'} Q(s_{t+1}, a'; \theta^-)$.

$L(\theta) = E[(y_t - Q(s_t, a_t; \theta))^2]$.

The discount factor γ determines the relative importance of future rewards. The learning rate, batch size, replay-buffer capacity, target update interval and exploration schedule should be selected using a validation environment rather than tuned on the final test scenarios. Double DQN may be considered to reduce overestimation of action values.

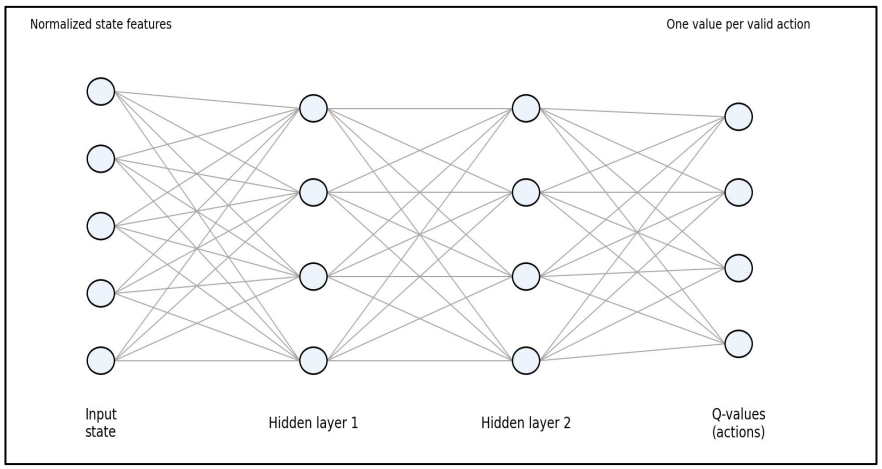


Figure 3. Conceptual Deep Q-Network architecture.

Multi-objective reward design

A reward function should express the application's priorities and avoid rewarding energy savings that are achieved by violating service requirements. One possible normalized reward is:

$r_t = w_E\ R_E + w_C\ R_C + w_Q\ R_Q + w_F\ R_F - w_D\ P_D - w_G\ P_G.$

R_E rewards lower energy expenditure relative to a reference budget; R_C rewards coverage; R_Q rewards packet delivery; R_F rewards balanced residual energy or fairness; P_D penalizes excessive delay; and P_G penalizes coverage below the required threshold. For example:

$R_C = CR(t), \quad P_G = max[0, CR_min - CR(t)].$

Energy reward can be defined as R_E = 1 − E_total(t)/E_ref, with clipping to a bounded interval to avoid extreme values. A packet-delivery reward may use PDR(t), while delay penalty can be normalized against D_max. Reward terms should be scaled to comparable ranges. A sensitivity analysis over weights is essential because arbitrary weights can make the policy prefer one objective at the expense of another.

Where coverage and connectivity are mandatory, the learning problem should be treated as constrained reinforcement learning or use a safety layer that rejects actions predicted to violate constraints. A penalty-based reward alone may permit temporary violations during exploration. In deployment, a rule-based safety shield can maintain a minimum active cover set and preserve at least one feasible route to the sink.

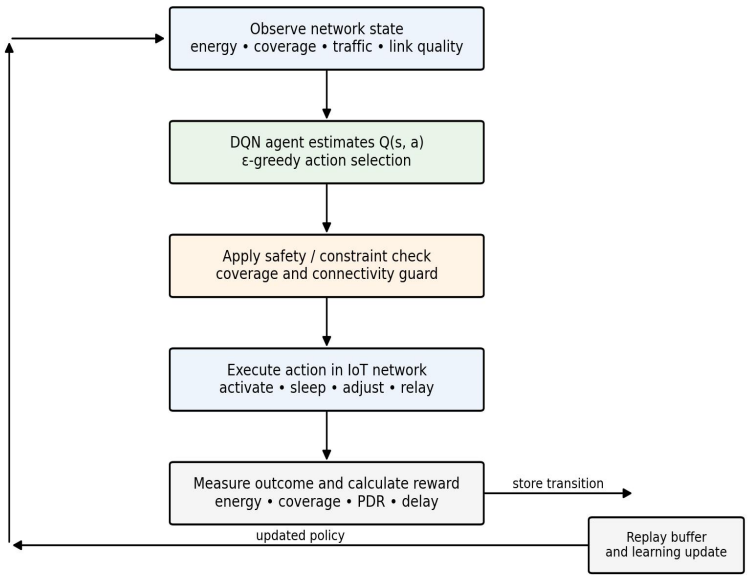


Figure 4. DQN-based adaptive decision and learning cycle

Pseudocode

Algorithm 1: DQN-based adaptive energy and coverage control
Input: network model, minimum coverage CR_min, service constraints, training episodes K.
Output: learned policy $\pi\theta$.

1. Initialize $Q(s,a;\theta)$, target $Q(s,a;\theta^{-})$, replay memory B.
2. For each episode k = 1,...,K: reset the network and observe s_0.
3. For each scheduling interval t: construct normalized state s_t and valid-action mask.
4. Select a_t using ε-greedy exploration; apply the safety shield.
5. Execute a_t; observe r_t, s_{t+1} and terminal flag.
6. Store (s_t, a_t, r_t, s_{t+1}, done) in B.
7. Sample a mini-batch and minimize the temporal-difference loss.
8. Update target-network parameters at the specified interval.
9. Reduce ε according to the validated exploration schedule.
10. Return the trained policy and evaluate it on independent test scenarios.

Distributed extension

For large deployments, the framework can be extended to cooperative multi-agent reinforcement learning. Each cluster or node agent observes local energy, neighbouring node status and local coverage contribution. Agents may share compact messages such as intended activity state, residual-energy category or coverage alert. Centralized training with decentralized execution is one possible design: a central critic can use broader information during training, while local actors make decisions using neighbourhood observations at run time.

Distributed learning reduces dependence on a single controller but introduces coordination challenges. Simultaneous sleep decisions can create coverage holes, while excessive message exchange can offset energy savings. A practical protocol should include periodic coordination, event-triggered alerts and a fallback schedule when communication with the controller is lost.

Research Methodology and Experimental Design

Research design

The study is designed as a quantitative simulation-based investigation. The experimental process comprises network generation, energy and coverage modelling, DQN training, baseline implementation, test evaluation and statistical comparison. The same topology, traffic traces, radio model and initial energy conditions should be used for each competing algorithm. Multiple random seeds are required to account for stochastic node placement, channel variation and exploration.

Proposed simulation configuration

Parameter	Candidate setting	Purpose
Network size	50, 100, 150, 200 nodes	Scalability assessment
Monitoring area	100 m × 100 m	Spatial deployment model
Initial energy	0.5–2.0 J per node (scenario dependent)	Battery sensitivity
Sensing radius	10–20 m	Coverage geometry
Communication range	25–40 m	Connectivity evaluation
Packet size	512 bytes	Traffic and radio energy
Scheduling interval	1–10 s (application dependent)	Control granularity
Minimum coverage	0.80 or application-specific	Coverage constraint
Learning algorithm	DQN; optional Double DQN	Adaptive control
Learning rate	0.001 initial candidate	Optimizer configuration
Discount factor	0.95–0.99	Long-term reward
Replay memory	10,000 transitions initial candidate	Experience replay
Batch size	32–128	Gradient update
Training episodes	Set after convergence study	Policy learning

These values are proposed starting points, not universal constants or experimental findings. The energy budget must be consistent with the selected radio hardware and simulation time scale. The final paper should report all simulator versions, random seeds, traffic models, antenna assumptions, channel parameters and DQN hyperparameters.

Baseline methods

- Fixed duty-cycle scheduling: nodes follow a predetermined active/sleep schedule.
- Residual-energy threshold: a node sleeps when its battery falls below a fixed threshold, subject to a basic coverage check.
- LEACH-style clustering: cluster-head rotation and aggregation reduce communication overhead.
- Conventional Q-learning: a tabular or compact-state RL baseline evaluates the contribution of deep function approximation.
- Coverage-aware heuristic: nodes are ranked by residual energy and marginal coverage contribution.
- Proposed DQN RLAEM-CC: a learned policy jointly selects energy and coverage actions.

Baseline implementations must be configured fairly. For example, each method should receive the same sensing and communication information where that information is available to the method. If a baseline is not designed to optimize routing, its packet-delivery performance should be interpreted accordingly rather than treated as a direct algorithmic failure.

Evaluation metrics

The following measures should be reported for each test run:

Metric	Definition / interpretation
Total energy consumed	Sum of node energy expenditure over the evaluation horizon.
Residual energy	Remaining network energy and its distribution across nodes.
First node death (FND)	Number of rounds until the first node exhausts its energy.
Half nodes death (HND)	Rounds until 50% of nodes are dead.
Last node death (LND)	Rounds until the final node exhausts its energy.
Area coverage ratio	Fraction of grid points covered by at least one active sensor.
Packet delivery ratio	Successfully received packets divided by generated or transmitted packets; denominator must be stated.
End-to-end delay	Mean time from packet generation to successful sink reception.
Energy fairness	Jain's index over node-level energy consumption or residual-energy allocation.
Control overhead	Bytes or energy used for scheduling and coordination messages.

$$\textit{Jain's fairness index: } J = (\Sigma_i x_i)^2 / [N \Sigma_i x_i^2],$$

where x_i is the selected non-negative quantity, such as residual energy or energy consumed, and the interpretation must be stated. A value closer to one indicates greater equality among the measured quantities; it does not by itself imply better network performance.

6.5 Statistical analysis

For each algorithm and scenario, report the mean, standard deviation and 95% confidence interval across independent runs. Paired comparisons should use the same random seeds and topology realizations where possible. If parametric assumptions are reasonable, repeated-measures ANOVA or paired t-tests may be applied with appropriate correction for multiple comparisons. Otherwise, use a non-parametric alternative such as the Wilcoxon signed-rank test for paired results or Kruskal-Wallis test for independent groups. Effect sizes should accompany p-values.
Ablation experiments should remove one reward component at a time, such as the coverage penalty, energy fairness term or delay penalty. Additional tests should vary network density, battery heterogeneity, traffic intensity, sensing range and minimum coverage requirement. This will reveal whether the framework's behaviour is robust or depends on a narrow parameter configuration.

RESULTS REPORTING FRAMEWORK AND DISCUSSION

Results status

The proposed framework has been mathematically specified, but numerical results are not reported here because a complete simulator run has not been conducted as part of this manuscript preparation. It would be scientifically inappropriate to insert invented energy savings, lifetime gains, coverage percentages or statistical significance values. The tables below are intended to be populated after implementation and repeated testing.

Comparative results template

Algorithm	Energy consumed (J)	FND (rounds)	Coverage (%)	PDR (%)	Delay (ms)
Fixed duty cycle	To be measured	To be measured	To be measured	To be measured	To be measured
Energy threshold	To be measured	To be measured	To be measured	To be measured	To be measured
LEACH-style	To be measured	To be measured	To be measured	To be measured	To be measured
Q-learning	To be measured	To be measured	To be measured	To be measured	To be measured
Coverage heuristic	To be measured	To be measured	To be measured	To be measured	To be measured
Proposed DQN	To be measured	To be measured	To be measured	To be measured	To be measured

Expected behaviour and interpretation criteria

If the reward and safety design operate as intended, the DQN controller should learn to avoid keeping every node active when redundant coverage exists. It may rotate active responsibility among nodes with comparable spatial contribution, preserving energy in nodes that are temporarily

unnecessary. When a coverage deficit is detected, the policy should activate a suitable sleeping node or adjust sensing rates. However, this expected behaviour must be verified by logs of the selected actions and by comparison with baseline policies.

A lower total energy figure is not sufficient evidence of success if it is achieved by allowing coverage or packet delivery to collapse. Results should therefore be interpreted jointly. A useful policy is one that reduces energy use while meeting the stated coverage, connectivity and delay constraints. If a method improves network lifetime but increases control overhead, the trade-off should be quantified. Similarly, high coverage with a very large active-node count may not be energy-efficient.

Threats to validity

Simulation conclusions depend on assumptions about node placement, radio propagation, traffic, battery behaviour and sensing geometry. A deterministic circular sensing model may overestimate real coverage in obstructed environments. Ideal links may overstate packet delivery. A centralized controller may underestimate signalling delay and controller failure risk. The study should test alternative channel models, imperfect sensing and communication failures. The policy should also be evaluated on topologies and random seeds not used during training to measure generalization.

The learned policy may exploit inaccuracies in the simulator. This simulation-to-reality gap can be reduced through hardware-in-the-loop tests, measured power profiles and controlled field trials. A lightweight model may be required for low-power nodes. Security should also be considered because falsified battery reports, manipulated coverage alerts or compromised agents could lead to unsafe scheduling decisions.

PRACTICAL IMPLICATIONS AND FUTURE RESEARCH

Application domains

In precision agriculture, sensor nodes monitor soil moisture, temperature and environmental conditions across fields. Adaptive sensing can increase sampling during irrigation or stress events and reduce activity when conditions are stable. In environmental monitoring, coverage constraints can prioritize river points, forest boundaries or pollution hotspots. In industrial IoT, critical machines may require high-frequency sensing while non-critical areas can tolerate lower sampling rates. In smart buildings, occupancy and air-quality sensors can adapt activity according to building use.

Edge deployment

Training a DQN directly on every sensor may be impractical because of limited memory, processing capacity and battery power. A more feasible design trains the policy at a gateway or edge computer and distributes a compressed inference model or schedule to nodes. The system should retain local safety rules so that nodes can continue essential sensing when the gateway is unavailable. Model updates should be authenticated and versioned.

Federated and multi-agent learning

Federated learning may allow multiple gateways to train local models and share model updates rather than raw measurements. This can reduce central data collection but does not automatically guarantee privacy or low energy cost; model exchange itself consumes bandwidth and computation. Multi-agent reinforcement learning can support local autonomy, but coordination protocols must prevent conflicting actions. Future work should compare centralized DQN, independent learners and centralized-training/decentralized-execution architectures.

Safe and explainable control

Future research should investigate constrained RL, safe exploration and explainable policies. A safety shield can block actions that would disconnect a critical target or reduce coverage below a hard threshold. Explainability methods can identify which state features influenced an activation decision, helping operators audit network behaviour. Robustness to sensor faults, malicious reports and sudden traffic changes should be evaluated before deployment in critical infrastructure.

Hardware validation

A practical validation programme may use ESP32 or other low-power sensor boards, a Raspberry Pi or industrial gateway, and calibrated current measurement equipment. Tests should record sleep current, sensing current, transmit current, receive current, packet loss and inference latency. Field trials should compare predicted energy consumption with measured battery depletion and should assess whether the learned policy remains effective under radio interference and environmental obstruction.

CONCLUSION

This paper has proposed RLAEM-CC, a reinforcement learning-based framework for adaptive energy management and coverage control in IoT networks. The central premise is that energy scheduling and sensing coverage must be optimized together because every activation or sleep decision changes both battery consumption and the spatial availability of observations. The framework represents network management as a constrained sequential decision problem and defines state variables for residual energy, node activity, coverage contribution, traffic, link quality and application priorities.

A DQN-based controller is specified to learn action values for node activation, sleep scheduling, sensing-rate adjustment, transmission-power selection and relay choice. A multi-objective reward combines energy efficiency, coverage, packet delivery and fairness while penalizing delay and coverage deficits. The mathematical model includes a first-order radio energy abstraction, geometric coverage estimation and service constraints. The experimental methodology defines simulation parameters, baselines, performance measures, statistical tests and ablation studies.

The proposed framework is a research design rather than a completed empirical validation. Its effectiveness must be established through reproducible simulation and, ultimately, hardware experiments. Future work should implement the complete controller, compare it against strong energy-aware and coverage-aware baselines, evaluate generalization to

unseen topologies, and quantify the overhead of centralized and distributed learning. With these validations, the framework may contribute to more sustainable and reliable IoT deployments in environmental monitoring, agriculture, industrial automation and smart infrastructure.

References

[1] Heinzelman, W. R., Chandrakasan, A., & Balakrishnan, H. (2000). Energy-efficient communication protocol for wireless microsensor networks. Proceedings of the 33rd Annual Hawaii International Conference on System Sciences. https://doi.org/10.1109/HICSS.2000.926982

[2] Heinzelman, W. B., Chandrakasan, A. P., & Balakrishnan, H. (2002). An application-specific protocol architecture for wireless microsensor networks. IEEE Transactions on Wireless Communications, 1(4), 660–670. https://doi.org/10.1109/TWC.2002.804190

[3] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., et al. (2015). Human-level control through deep reinforcement learning. Nature, 518, 529–533. https://doi.org/10.1038/nature14236

[4] Pushpa, G., Anand Babu, R., Subashree, S., et al. (2025). Optimizing coverage in wireless sensor networks using deep reinforcement learning with graph neural networks. Scientific Reports, 15, Article 16681. https://doi.org/10.1038/s41598-025-01841-2

[5] Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). MIT Press.

[6] Wang, X., Chen, H., & Li, S. (2023). A reinforcement learning-based sleep scheduling algorithm for compressive data gathering in wireless sensor networks. EURASIP Journal on Wireless Communications and Networking, 2023, Article 28. https://doi.org/10.1186/s13638-023-02237-4

[7] Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. Machine Learning, 8, 279–292. https://doi.org/10.1007/BF00992698

[8] Zhou, P., Kan, M., Chen, W., Wang, Y., et al. (2025). An adaptive coverage method for dynamic wireless sensor network deployment using deep reinforcement learning. Scientific Reports, 15, Article 30304. https://doi.org/10.1038/s41598-025-16031-3

[9] Yuan, J., Peng, J., Yan, Q., He, G., Xiang, H., & Liu, Z. (2024). Deep reinforcement learning-based energy consumption optimization for peer-to-peer communication in wireless sensor networks. Sensors, 24(5), 1632. https://doi.org/10.3390/s24051632

[10] Hu, L., Han, C., Wang, X., Zhu, H., & Ouyang, J. (2024). Security enhancement for deep reinforcement learning-based strategy in energy-efficient wireless sensor networks. Sensors, 24(6), 1993. https://doi.org/10.3390/s24061993

[11] Vergin Raja Sarobin, M., Akil, S., Shalu, S. B., Sriyaa, S., Tamizhnambi, Y., et al. (2026). Energy optimized scheduling in wireless sensor networks (WSNs) using hybrid bio-inspired reinforcement learning approach. Scientific Reports, 16, Article 13109.