ISSN 2090-3359 (Print)
ISSN 2090-3367 (Online)

Advances in Decision Sciences

Volume 30
Issue 2
June 2026

亞洲大學
ASIA UNIVERSITY
SCIENTIFIC & BUSINESS WORLD

Published by Asia University, Taiwan and Scientific and Business World

Multi-Objective Constrained Reinforcement Learning for Joint Routing–MAC–Duty Cycling in Low-Power Wireless Sensor Networks

Ghaida Muttashar Abdulsahib

College of Computer Engineering, University of Technology, IRAQ

Email: Ghaida.M.Abdulsaheb@uotechnology.edu.iq

Mohammed Awad Mohammed Ataelfadiel

Applied College, King Faisal University, Saudi Arabia

Corresponding author Email: melfadiel@kfu.edu.sa

Received: March 13, 2026; First Revision: March 22, 2026;

Last Revision: April 3, 2026; Accepted: April 4, 2026;

Published: April 5, 2026

Abstract

Introduction: Wireless Sensor Networks (WSNs) face significant challenges in balancing energy efficiency, latency, and reliability while operating under severe resource constraints. Current methods either optimize network layers separately or use static cross-layer coordination, which doesn't work well when conditions change.

Purpose: The aim of this study is to introduce a Constrained Multi-Objective Reinforcement Learning Model (CMORLM) for optimizing Joint Routing, Medium Access Control (MAC), and Duty Cycling Optimization (DCO) in low-power WSNs.

Methods: In this paper, we have suggested the CMORLM approach as a constrained Markov Decision Process (MDP) with three competing goals: lowering Energy Consumption (EC), lowering End-to-End-Latency (EEL), and raising Packet Delivery Ratio (PDR). There are strict limits on the amount of residual energy, the buffer size, and the Quality of Service (QoS) requirements. Lagrangian Constraint Handling (LCH) and multi-objective policy gradients are combined within the primal-dual optimization method. For routing, MAC, and DCO, the policy network uses a shared encoder with factorized heads. Federated Gradient Aggregation (FGA) is used for distributed learning across Sensor Nodes (SN).

Results: Testing in NS-3 shows that EC is 34.2% lower, EEL is 41.3% higher, and PDR is 16.5% higher than Traditional Layered Protocols (TLP). Network Lifetime (NL) goes up by 38.4%. The constraint violation rate (CVR) is still below 1%. This is 23 times less than the CMORLM that was suggested. Ablation studies show that joint optimization increases the EC by 44.7% over single-layer control.

Conclusion: The suggested CMORLM works well on networks with 50 to 200 nodes and can handle changes in traffic, node failures, and mobile sinks. To enable operator control over performance trade-offs through weight configuration, Pareto frontier analysis is performed**.**

Keywords: WSNs, reinforcement learning, multi-objective optimization, constrained Markov decision processes, cross-layer optimization, energy efficiency

JEL Classifications: C44, C61, L96, C63

1 Introduction

WSNs are now a key technology for Internet of Things (IoT) apps. These uses include environmental monitoring, factory automation, and smart farming and healthcare systems (Trigka & Dritsas, 2025). Typically, these networks consist of 100 to 1,000 resource-constrained nodes powered by batteries with strictly limited operational lifespans. The nodes are used in situations where it is not possible or practical to change the batteries by hand. The main challenge in designing a WSN is finding the optimal balance between Network Lifetime (NL) and Quality of Service (QoS) for data collection applications (Aburukba & El Fakih, 2025; Khan, Mazhar et al., 2025). Energy efficiency directly affects how well a deployment will work (Behera et al., 2022). Network performance in WSN emerges from complex interactions across multiple protocol layers (Kumar et al., 2025). The routing decisions established and implemented in determine the multi-hop forwarding paths data packets take, from source to sink (Schlichter et al., 2025). As Bhutani et al. (2025) explain, Medium Access Control (MAC) protocols also control channel access, preventing collisions and retransmissions. Also, Rottleuthnerv et al. (2025) say that Duty Cycling Optimization (DCO) controls when radios go to sleep and wake up. At the same time, these three control aspects rely on each other in important ways. Conversely, achieving energy efficiency in routing via minimum-hop circuits sometimes requires low transmission power. Ekpenyong et al. (2022) and Salim (2023) say that this affects contention at the physical layer (MAC). For example, delays might get much worse if they coincide with long periods of sleep and many work changes. Another thing to consider is that routing that uses less energy may affect MAC-layer contention. In the past, when building WSN protocol stacks, each layer had to work as well as possible. Routing protocols often employ heuristics that are either manually adjusted or left unchanged.

These protocols consider many factors to determine forwarding pathways, such as the number of hops, the number of transmissions expected, and the amount of residual energy (RE). But these decisions don't account for contention levels at the MAC layer or the delays introduced by DCO (Islam & Lee, 2019). One distinguishing feature of MAC protocols is that they are configured using parameters that have already been specified. Because of this, it is harder for them to adapt to changes in the network or traffic flow. The current tiered structure does not allow it to fully leverage the potential benefits of interactions across different protocol levels. The research by Feng et al. (2025) shows that this makes it less likely that performance will become better. Researchers have proposed many cross-layer optimization strategies to address these limitations. Conversely, most currently available methodologies are either analytical models or metaheuristic algorithms. Mustafa et al. (2025) assert that all methodologies require fundamental knowledge of network architecture, channel conditions, and traffic flows. Because of this, these technologies aren't used sufficiently in the real world.

In rare cases, the way packages are delivered may change over time. Interference and fading are two factors that can negatively affect a network's performance. Unexpected changes in connections can occur when nodes fail. Recent developments in reinforcement learning have enabled networks to learn from mistakes and adapt to their environment. Terven's (2025) research shows that Deep Reinforcement Learning (DRL) is effective in complex decision-making situations. However, there are still significant challenges in using RL within Wireless Sensor Networks (WSNs). Real-world implementations are subject to stringent safety protocols. Priyadarshi (2024) and Luong et al. (2019) posit that safety nets (SNs) are essential for preventing buffer overflows, ensuring service quality, and

maintaining energy levels above a critical point. Therefore, methods in unconstrained multi-objective reinforcement learning (UMORL) could impose considerable restrictions on the actions allowed during the exploration phase.

Moreover, network operators need to understand the trade-offs in performance that arise when multiple goals conflict. These trade-offs encompass factors such as energy consumption, response time, and reliability. Consequently, operational effectiveness may be diminished by these inherent compromises. Network operators require a clear understanding of performance implications, transcending manual modifications and reward functions that lack automation and user-friendliness. Furthermore, to optimize constrained sensor node resources, the implementation of distributed learning methodologies and lightweight policy frameworks is essential to mitigate substantial computational requirements (El-Hajj, 2025).

This work employs a CMORLM to address these challenges within the joint routing, MAC, and DCO framework. The proposed CMORLM uses a constrained Markov Decision Process that incorporates operational safety limits alongside energy, delay, and reliability requirements to model the cross-layer control problem. The primal-dual optimization method combines policy gradient techniques with Lagrangian Constraint Handling (LCH) to change the weights of policy parameters and constraint penalties. Constrained-prediction networks ensure that actions are safe to take by checking whether they can be performed before they are carried out. The policy architecture uses a shared feature encoder for routing, MAC, and DCO. It also has factorized output heads to make decision making easier and reduce the number of actions available. Federated Gradient Aggregation (FGA) enables distributed learning by allowing nodes to update their own policies and periodically synchronize their parameters within clustered communication structures. We perform Pareto frontier analysis to enable operators to control performance trade-offs by adjusting weights. In this context, CMORLM is clearly defined as a decision-support framework. Operator preferences are represented by configurable objective weights, and Pareto frontier analysis provides a systematic method for rational decision-making regarding trade-offs among energy, latency, and reliability.

The proposed framework offers four primary contributions. First, it unifies routing, MAC, and duty-cycle optimization into a single constrained reinforcement learning model with clear safety requirements and conflicting performance goals. Second, it uses a primal-dual reinforcement learning algorithm that simultaneously optimizes policies and satisfies constraints by adjusting the Lagrange multiplier as needed. Third, it uses a factorized neural network architecture and a distributed training protocol to address communication and computational challenges in WSNs with limited resources. Fourth, many tests have shown that joint optimization, multi-objective formulations, and methods for handling constraints achieve much better results than traditional layered protocols and current learning-based approaches.

From a decision sciences perspective, managing a WSN is a complex Multi-Criteria Decision-Making (MCDM) problem. Network operators are frequently forced to make subjective trade-offs between conflicting Key Performance Indicators (KPIs), such as extending battery life versus ensuring ultra-low latency. Therefore, this study frames the proposed CMORLM not merely as an automated routing protocol, but as an intelligent decision-support framework. By utilizing configurable objective weights and Pareto frontier analysis, our model empowers network managers with explicit, rational control

over system performance, ensuring that operational policies align directly with strategic business or application requirements.

The rest of this work is set up as follows: Section 2 discusses related work; Section 3 discusses the system model and problem formulation; and Section 4 discusses the proposed CMORLM. Section 5 discusses how the experiment was set up, Section 6 presents the results and analysis, and Section 7 discusses the limitations and potential future developments.

2 Literature Review

2.1 RL-Driven Routing Adaptation in Low-Power IoT/WSN

Recent studies treat routing in Low-Power and Lossy Networks (LLNs/WSNs) as a sequential decision problem in which link quality, congestion, and node energy evolve with delay. Dey and Ghosh (2024) introduced multi-agent RL for adjusting IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) routing objectives under non-stationary conditions. Lei and Liu (2024) applied RL to routing with explicit load-balancing intent. In AMI-style mesh networks, Santos et al. (2024) used Q-learning to refine RPL-type routing decisions. Moving closer to explicit trade-off handling, Halloum et al. (2026) proposed multi-objective RL for adaptive routing with load balancing. However, these routing-centric works largely assume MAC behavior and duty cycling are fixed or externally tuned.

2.2 RL for MAC-Layer Scheduling and Access Control

At the MAC layer, RL has been applied for adapting scheduling and access parameters, particularly for IEEE 802.15.4e Time Slotted Channel Hopping (TSCH). Zerguine et al. (2025) developed an RL-based TSCH scheduling method to improve performance under varying interference and traffic conditions. Ben Yaala et al. (2025) similarly optimized TSCH scheduling with RL for Industrial Internet of Things (IIoT)/LLN environments. Further, Panda et al. (2025) advanced prioritized multi-agent RL TSCH schedulers. Most MAC and scheduling studies treat routing and duty cycling as given, and schedule optimization is not jointly aligned with routing-induced load shifts or duty-cycle-driven neighbor availability.

2.3 RL-Enabled Duty Cycling and Energy-Aware MAC Operation

Duty cycling remains central to low-power WSN operation because idle listening and overhearing frequently dominate Energy Consumption (EC) in sparse traffic regimes. Latif et al. (2025) implemented an RL-based intelligent duty-cycle MAC protocol that adapts sleep and wake behavior using experimental performance feedback. Khan, Ullah et al. (2025) applied RL to improve B-MAC-style operation. Further, these works demonstrate that RL can tune the duty cycle. However, routing and MAC scheduling are typically treated as external factors. Duty cycling directly alters neighbor availability, forwarding delay, and retransmission probability and contention patterns.

For system-level energy optimization, DRL has also been used by embedding energy cost directly into the learning objective. Yuan et al. (2024) verified DRL for EC optimization in WSN peer-to-peer

communication. However, such energy-centric studies often focus on a subset of protocol decisions or assume simplified MAC/routing settings.

2.4 Synthesis of Gaps Toward Joint Multi-Objective Constrained Control

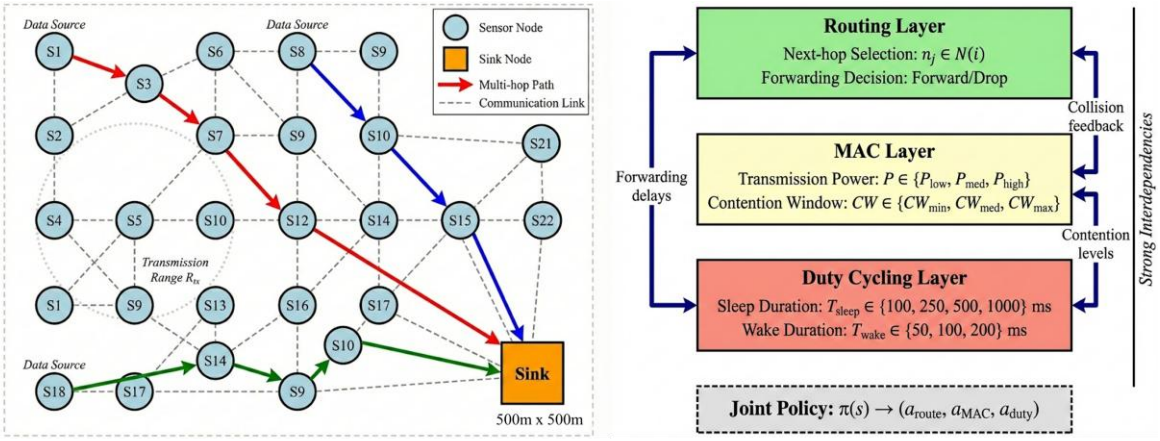
Across routing-focused (Dey & Ghosh, 2024; Halloum et al., 2026; Lei & Liu, 2024; Santos et al., 2024), MAC/scheduling-focused (Ben Yaala et al., 2025; Panda et al., 2025; Zerguine et al., 2025), and duty-cycle-focused (Khan, Ullah et al., 2025; Latif et al., 2025) studies, a consistent limitation is the predominantly *single-layer* learning design. Three gaps are most relevant to the present scope. First, cross-layer coupling is under-modelled: routing changes contention and load distribution; MAC scheduling changes service rates and collision rates in the network; and duty cycling changes connectivity and forwarding feasibility, yet these are rarely optimized under a single policy. Second, multi-objective trade-offs are handled inconsistently: even when routing is formulated as a multi-objective problem (Halloum et al., 2026), MAC and duty-cycle trade-offs are not jointly optimized. Third, constraint satisfaction is frequently evaluated after training rather than embedded into the learning formulation; duty-cycle bounds, deadline-miss limits, and reliability requirements are typically treated as post hoc checks.

These gaps motivate a unified **multi-objective constrained reinforcement learning** formulation for **joint routing–MAC–duty cycling** in low-power WSN, where energy–latency–reliability trade-offs are jointly optimized, and operational constraints are enforced during learning rather than offline tuning.

3 System Model and Problem Formulation

3.1 System Model

Figure 1: WSN and Cross-Layer Control Model



(a) (b)

Note: This figure illustrates the system architecture. (a) Depicts the multi-hop network topology where sensor nodes (blue) route data to the sink node (orange) via intermediate relays. (b) Details the proposed cross-layer control framework, highlighting the strong interdependencies between the Routing layer (next-hop selection), the MAC layer (transmission power and contention window), and the duty cycling layer (sleep/wake durations) to optimize network performance.

For modelling the WSN it is represented as directed graph $G = (V, E)$, where $'V'$ contains $'N'$ sensor nodes $'\{n_1, \dots, n_N\}'$, and sink node $'n_s'$. Directed link $(n_i, n_j) \in E$ exists when $'n_j'$ is within

transmission range of $'n_i'$. Each node $'n_i'$ has finite initial energy $'E_i^{init}'$ and buffer capacity $'B_i'$ and generates packets at a rate $'\lambda_i'$. Packets reach $'n_s'$ via multi-hop forwarding by intermediate relays (Figure 1a).

Traffic varies from periodic sensing with near-constant rates to event-driven, bursty arrivals. Network supports heterogeneous traffic classes with different QoS targets, including latency bounds. D_{max} for time-critical flows and minimum delivery requirements PDR_{min} for reliability-sensitive flows. Forwarding follows a store-and-forward operation where the queue at node $'n_i'$ evolves from local and upstream arrivals *vs.* successful downstream transmissions. Buffer overflow causes drops. Each node radio has four states—transmit, receive, idle listening, and sleep—and EC follows the first-order radio dissipation model.

Transmit energy for an *L* bit packet over distance *d* is defined in Equation 1:

$$E_{tx}(L, d) = E_{elec} \cdot L + \epsilon_{amp} \cdot L \cdot d^{\alpha}, \tag{1}$$

where E_{elec} is the per-bit energy cost for transmitter electronics; ϵ_{amp} is the amplifier energy coefficient, and α is the path loss exponent, ranging from 2 for free space propagation to 4 for multi-path fading environments. Receiving a packet consumes energy proportional to packet length, as shown in Equation 2:

$$E_{rx}(L) = E_{elec} \cdot L. \tag{2}$$

In idle listening, the radio monitors a channel without data exchange and consumes power as P_{idle} per unit time. In sleep mode, power drops to P_{sleep}, but transitions between sleep and active states incur switching energy E_{switch}. Residual Energy (RE) at node n_i evolves according to cumulative EC across all radio activities.

At a discrete time step t, the energy level updates as described by Equation 3:

$$E_i(t+1) = E_i(t) - \left(E_{tx}^i(t) + E_{rx}^i(t) + E_{idle}^i(t) + E_{sleep}^i(t) + E_{switch}^i(t)\right). \tag{3}$$

Node n_i becomes non-functional when RE falls below the minimum threshold E_{min}^i required for basic operation.

Channel access follows carrier-sense multiple access with collision avoidance principles. Interference occurs when multiple nodes within mutual range transmit simultaneously. Signal to interference plus noise ratio at receiver n_j from transmitter n_i is assumed by Equation 4:

$$SINR_{ij} = \frac{P_{tx} \cdot G_{ij}}{N_0 + \sum_{k \in I} P_k \cdot G_{kj}}, \tag{4}$$

where P_{tx} is transmission power; G_{ij} is channel gain; N_0 is noise power; *I* identifies a set of interfering transmitters.

Joint optimization spans three protocol layers (Figure 1b). At the routing layer, each node n_i uses a forwarding policy that maps buffered packets to the next hop in $A^i_{route} = N(i) \cup \{\emptyset\}$, where $\emptyset$ is drop action. At the MAC layer, carrier sense access is controlled by discrete actions A^i_{MAC} including transmit power levels $\{P_1, \dots, P_M\}$ and contention window options $\{CW_{min}, CW_{med}, CW_{max}\}$, and backoff methods.

Duty cycling control determines the temporal pattern of sleep and wake periods. Duty cycle action A^i_{duty} is sleep duration $T^i_{sleep}$and subsequent active listening window T^i_{listen} for node n_i. Duty cycle ratio is defined as the fraction of time the radio remains active and is computed using Equation 5:

$$DC_i = \frac{T^i_{listen}}{T^i_{sleep} + T^i_{listen}}. \tag{5}$$

Joint action at time t is $a_t = (a^t_{route}, a^t_{MAC}, a^t_{duty})$ and this captures simultaneous routing, MAC, and duty cycle decisions. These layers are coupled because routing affects transmission attempts and collision exposure, while duty cycling controls when nodes contend for the channel, and MAC parameters determine forwarding success.

To formulate the joint optimization problem, the following standard assumptions are made regarding the network environment:

Traffic Model: The network supports heterogeneous traffic. Periodic sensing follows a Constant Bit Rate (CBR), while event-driven traffic follows a Poisson arrival process, simulating unpredictable environmental bursts.

Channel and Interference Model: Wireless propagation is modeled using a log-distance path loss model with multi-path fading. Interference is calculated using the Signal-to-Interference-plus-Noise Ratio (SINR), assuming carrier-sense multiple access with collision avoidance (CSMA/CA).

Failure Model: Node failures occur either deterministically due to energy depletion (when residual energy falls below E_{min}) or stochastically due to hardware/environmental anomalies (modeled as random node deactivations).

3.2 Problem Formulation

The optimization problem balances 3 competing objectives that characterize network performance. Network average EC calculates energy efficiency per PDR in Equation 6:

$$J_1(\pi) = \frac{\sum_{i=1}^{N} \sum_{t=0}^{T} E_i^{consume}(t)}{\sum_{i=1}^{N} P_i^{delivered}}, \tag{6}$$

where $E_i^{consume}(t)$ denotes the total energy consumed by node i at time step t, $P_i^{delivered}$ represents the total number of packets successfully delivered to the sink that originated from node i, N is the total number of sensor nodes, and T is the total operational time horizon.

Subsequently, End-to-End Latency (EEL) is measured as the average delay experienced by packets from generation at source nodes to reception at the sink. The EEL objective is shown in Equation 7:

$$J_2(\pi) = \frac{1}{|P|} \sum_{p \in P} (T_p^{arrival} - T_p^{generation}), \quad (7)$$

where P is the set of all successfully delivered packets, $|P|$ is the total number of such packets, $T_p^{arrival}$ is the exact timestamp when packet p successfully arrives at the sink, and $T_p^{generation}$ is the timestamp when packet p was originally generated at the source node.

Reliability is computed using PDR, defined as the fraction of generated packets that reach the sink within the timeout period.

Reliability objective in Equation 8:

$$J_3(\pi) = \frac{\sum_{i=1}^{N} P_i^{delivered}}{\sum_{i=1}^{N} P_i^{generated}}. \quad (8)$$

where $P_i^{generated}$ is the total number of packets generated by node i during the operational period, and $P_i^{delivered}$ is as defined previously.

For each node i and time t the energy constraint in Equation 9:

$$C_1^i(t): E_i(t) \geq E_{min}^i. \quad (9)$$

Buffer constraints prevent packet drops due to overflow.

Buffer constraint for node i in Equation 10:

$$C_2^i(t): Q_i(t) \leq B_i. \quad (10)$$

QoS constraints ensure application requirements are met.

Latency-sensitive applications impose per-packet delay bounds in Equation 11:

$$C_3(p): T_p^{arrival} - T_p^{generation} \leq D_{max}, \quad (11)$$

for all PDR as p belonging to time-critical traffic classes.

Reliability-critical applications enforce minimum PDR requirements in Equation 12:

$$C_4: J_3(\pi) \geq PDR_{min}. \quad (12)$$

Feasible policy space $\Pi_{feasible}$ consists of all policies that satisfy these constraints in expectation over the policy-induced state distribution.

Joint routing–MAC–DCO problem is modelled as constrained multi-objective MDP $\langle S, A, P, r, C, \gamma \rangle$. State space S aggregates node results as $s_i = (E_i, Q_i, N_i, \lambda_i^{obs})$ that captures RE, queue occupancy, neighbor connectivity, and experimental traffic rate. Action space $A = A_{route} \times A_{MAC} \times A_{duty}$ is joint action $a = (a_{route}, a_{MAC}, a_{duty})$ for simultaneous cross-layer control. Transition kernel *$P :S \times$*

A \times S \to [0,1]$ models stochastic packet arrivals, channel variations, collisions, and energy depletion. Reward vector $r = (r_1, r_2, r_3)$ encodes three objectives using immediate rewards derived from Equations (6) to (8) and constraint vector $C = (C_1, C_2, C_3, C_4)$ quantifies constraint violations under each state–action pair. Discount factor $\gamma \in [0,1)$is used.

A constrained multi-objective optimization problem seeks a stochastic policy $\pi: \mathcal{S} \to \Delta(\mathcal{A})$ that maps states to probability distributions over actions.

Value functions for each objective under policy π are defined in Equation 13:

$$V_k^{\pi}(s) = E_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \, r_k(s_t, a_t) \mid s_0 = s \right], \tag{13}$$

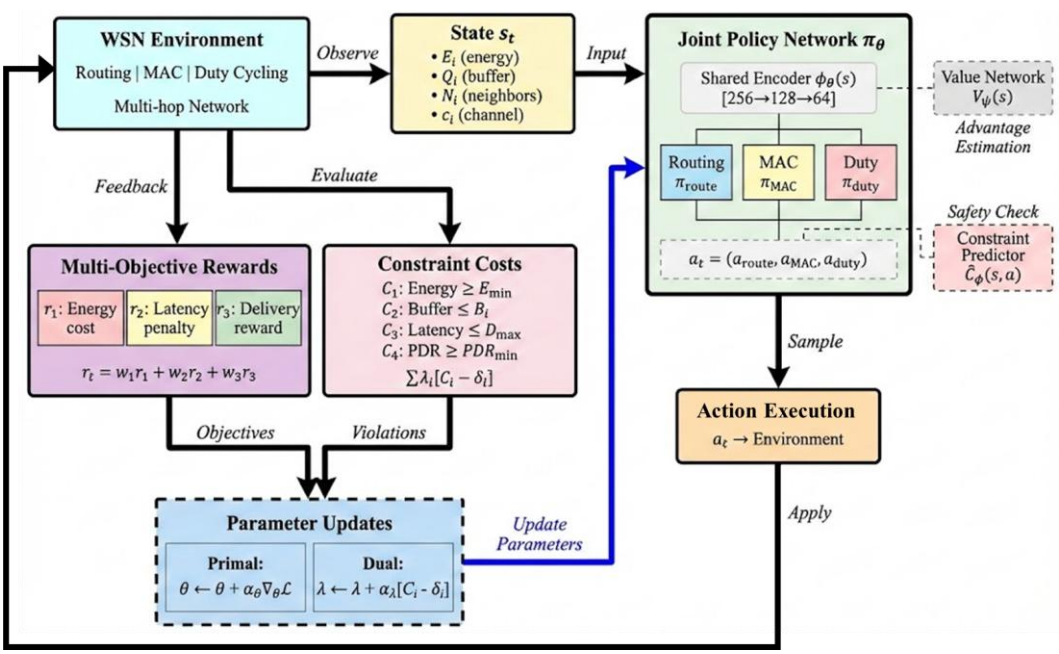
for objectives $k \in \{1,2,3\}$. Feasibility conditions require expected cumulative constraint costs to remain below specified thresholds δ_i over an infinite horizon, and this is formulated in Equation 14:

$$E_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \, C_i(s_t, a_t)\right] \leq \delta_i, \forall i \in 1,2,3,4. \tag{14}$$

The objective is to identify the Pareto frontier of policies $\pi^* \in \Pi_{feasible}$ such that no policy $\pi' \in \Pi_{feasible}$ exists satisfying $V_k^{\pi'}(s_0) \geq V_k^{\pi^*}(s_0)$ for all objectives $k$with strict inequality for at least one objective.

4 Proposed CMORLM

Figure 2: The proposed CMORLM for Joint Cross-Layer Control



Note: The workflow of the Constrained Multi-Objective Reinforcement Learning Model (CMORLM). The agent observes the WSN state (energy, buffer, neighbors, channel) and utilizes a Joint Policy Network to output cross-layer actions. The environment returns multi-objective rewards (energy, latency, reliability) and constraint violation costs. A Primal-Dual parameter update mechanism is employed to optimize the policy while strictly enforcing operational constraints via Lagrange multipliers.

The proposed CMORLM uses a factorized neural policy for distributed learning across nodes. As summarized in Figure 2, the workflow includes a state simulation model, multi-objective reward construction, constraint-aware policy optimization, and distributed coordination. Learning proceeds

iteratively, and each node detects its local state and selects joint cross-layer actions via a learned policy. It receives objective feedback on energy, latency, and reliability, checks constraint satisfaction, and updates parameters via gradient-based primal-dual optimization. The primal step improves the weighted objective, and the dual step updates Lagrange multipliers using experiential constraint violations.

4.1 State, Action, and Observation Design

A. State Representation: The state space S is the Cartesian product of local node states. Each node n_i maintains the local state (Equation 15):

$$s_i = (E_i, Q_i, N_i, c_i) \in \mathbb{R}^{d_s}, \tag{15}$$

where $E_i \in [0,1]$ is normalized RE, $Q_i \in [0,1]$ is the normalized buffer occupancy Q_i/B_i, $N_i \in \mathbb{R}^{N_{max}\times 4}$ is a neighbor feature matrix, and $c_i \in \mathbb{R}^H$ captures channel statistics over a history window of length H. Each row of N_i corresponds to a neighbor $n_j \in N(i)$ and is (E_j, d_{ij}, Q_j, h_j), encoding neighbor RE, normalized distance, neighbor buffer occupancy, and hop count to the sink; N_i is zero-padded up to N_{max}. Channel features c_i include average collision rate ρ_i^{coll}, RSSI measurements, and observed interference. The global state is $s = [s_1, \dots, s_N] \in \mathbb{R}^{N\times d_s}$.

B. Action Decomposition: The joint action space is decomposed (Equation 16) as:

$$A = A_{route} \times A_{MAC} \times A_{duty}. \tag{16}$$

For node n_i, the routing action space is $A_{route}^i = N(i) \cup \{\text{drop}\}$ with $| A_{route}^i | = | N(i) | + 1$, where a_{route}^i selects the next hop (or drops). The MAC action is $a_{MAC}^i = (P^i, CW^i) \in P \times W$, where $P = \{P_{low}, P_{med}, P_{high}\}$ and $W = \{CW_{min}, CW_{med}, CW_{max}\}$, giving $| A_{MAC}^i | = 9$ combinations.

The duty-cycling action is $a_{duty}^i = (T_{sleep}^i, T_{wake}^i) \in T_{sleep} \times T_{wake}$, with $T_{sleep} = \{100, 250, 500, 1000\}$ ms and $T_{wake} = \{50, 100, 200\}$ ms, generating $| A_{duty}^i | = 12$ options; the duty ratio follows Equation 5. At timestep t, $a_t = (a_{route}^t, a_{MAC}^t, a_{duty}^t) \in A$, and the per-node joint action cardinality is $| A^i | = | A_{route}^i | \cdot | A_{MAC}^i | \cdot | A_{duty}^i |$.

C. Observation Aggregation for Distributed Learning: Under distributed learning, each node n_i forms observations without centralized coordination. As defined in Equation 17:

$$o_i = (s_i, \bar{s}_{global}) \in O, \tag{17}$$

where s_i is the local node state and $\bar{s}_{global} = (\bar{E}, \bar{Q}, \bar{\lambda})$ aggregates network statistics: average RE as $\bar{E}$, average buffer occupancy $\bar{Q}$, and total traffic load $\bar{\lambda}$. These global statistics are broadcast by the sink via periodic beacons every as T_{beacon}. Partial observability occurs due to stale global updates and hidden neighbor states, so nodes maintain neighbor belief states $\hat{s}_j$ using last-received information and predictive models.

D. Feature Engineering and Normalization: State features are pre-processed to improve training stability and convergence. Energy and buffer variables use *min–max* normalization as defined in Equation 18:

$$\tilde{x} = \frac{x - x_{min}}{x_{max} - x_{min}}, \tag{18}$$

where x_{min}, x_{max} are physical bounds. The transmission range scales distances is R_{tx} as $\tilde{d}_{ij} = d_{ij}/R_{tx}$. Channel features are standardized using running statistics for the collision rate as ρ_i^{coll}, $\tilde{\rho}_i = (\rho_i - \mu_\rho)/\sigma_\rho$, where μ_ρ, σ_ρ are exponentially weighted moving averages updated as defined in Equation 19:

$$\mu_t = \beta\mu_{t-1} + (1-\beta)x_t, \tag{19}$$

with $\beta = 0.99$. Temporal dependence is modeled by concatenating the current state with the previous $H-1$ states to form $[s_{t-H+1}, \dots, s_t]$, providing short-term historical data to project and network dynamics.

4.2 Multi-Objective Reward and Preference Modelling

A. Reward Decomposition: The multi-objective problem converts the three objectives in Equaitons 6 to 8 into timestep rewards for policy learning. The energy reward for node n_i at time t penalizes consumption as defined in Equation 20:

$$r_1^i(t) = -\frac{E_i^{consume}(t)}{E_{ref}}, \tag{20}$$

where E_{ref} is a reference transmission energy for normalization, and the network reward is $r_1(t) = \sum_{i=1}^{N} r_1^i(t)$. The EEL reward penalizes delivery delay using Equation 21:

$$r_2(t) = -\frac{1}{|P_t|}\sum_{p\in P_t}\frac{T_p^{arrival} - T_p^{generation}}{D_{ref}}, \tag{21}$$

where P_t is the set of packets delivered at t and D_{ref} is a reference delay; if $|P_t| = 0$, then $r_2(t) = 0$. The reliability reward follows Equation 22, rewarding deliveries and penalizing drops:

$$r_3(t) = \frac{|P_t^{delivered}| - |P_t^{dropped}|}{|P_t^{total}|}, \tag{22}$$

where $P_t^{delivered}$, $P_t^{dropped}$, and P_t^{total} are delivered, dropped, and total packets at timestep t.

B. Multi-Objective Scalarization: The three reward components are combined using weighted scalarization as defined in Equation 23:

$$r(s_t, a_t) = w_1 r_1(t) + w_2 r_2(t) + w_3 r_3(t), \tag{23}$$

where $w = (w_1, w_2, w_3)$, $w_i \geq 0$, and $\sum_{i=1}^{3} w_i = 1$. The weights control the policy trade-off and its position on the Pareto frontier.

C. Reward Normalization Method: Each reward component is standardized using Equation 24:

$$\tilde{r}_k(t) = \frac{r_k(t) - \mu_k(t)}{\sigma_k(t) + \epsilon}, \tag{24}$$

where $\mu_k(t)$, $\sigma_k(t)$ are running mean and standard deviation updated by Equation 19 with $\beta = 0.99$, and $\epsilon = 10^{-8}$. The final scalar reward uses normalized components: $r(s_t, a_t) = w_1\tilde{r}_1(t) + w_2\tilde{r}_2(t) + w_3\tilde{r}_3(t)$.

4.3 Constraint Handling via Primal-Dual Optimization

A. Lagrangian Formulation: The constrained multi-objective problem in Section 3.6 is solved using a primal–dual method that converts it into a saddle-point optimization via a Lagrangian penalty on constraint violations, Equation 25,

$$L(\theta, \lambda) = J(\theta) - \sum_{i=1}^{4} \lambda_i \, [C_i(\theta) - \delta_i], \tag{25}$$

where θ are policy parameters, $J(\theta) = \sum_{k=1}^{3} w_k \, V_k^{\pi_\theta}(s_0)$ is the weighted value (Equation 13), $C_i(\theta)$ is the expected cumulative cost for constraint i, δ_i is the threshold (Equation 14), and $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ are nonnegative multipliers enforcing constraints. The solution seeks a saddle point (θ^*, λ^*) satisfying Equation 26:

$$L(\theta^*, \lambda) \leq L(\theta^*, \lambda^*) \leq L(\theta, \lambda^*), \tag{26}$$

for all θ, $\lambda \geq 0$.

B. Primal-Dual Update Mechanism: The primal step performs gradient ascent on the Lagrangian to improve the weighted objective while accounting for constraint costs, Equation 27,

$$\nabla_\theta L(\theta, \lambda) = \nabla_\theta J(\theta) - \sum_{i=1}^{4} \lambda_i \, \nabla_\theta C_i(\theta). \tag{27}$$

The objective gradient uses a sample-based policy-gradient estimator in Equation 28:

$$\nabla_\theta J(\theta) \approx \frac{1}{M} \sum_{m=1}^{M} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta \left(a_t^m \mid s_t^m \right) \hat{A}_t^m, \tag{28}$$

where M is the number of trajectories, T is episode length; $\hat{A}_t^m$ is the GAE advantage (discount γ, parameter λ_{GAE}). Constraint gradients are estimated similarly using Equation 29:

$$\nabla_\theta C_i(\theta) \approx \frac{1}{M} \sum_{m=1}^{M} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta \left(a_t^m \mid s_t^m \right) \hat{C}_{i,t}^m, \tag{29}$$

where $\hat{C}_{i,t}^m$ is the discounted cumulative cost for constraint i from timestep t. The primal parameter update follows Equation 30:

$$\theta_{k+1} = \theta_k + \alpha_\theta \nabla_\theta L(\theta_k, \lambda_k), \tag{30}$$

with learning rate α_θ.

The dual-step updates the multipliers based on the constraint violation. The dual gradient is assumed in Equation 31:

$$\nabla_{\lambda_i} L(\theta, \lambda) = \delta_i - C_i(\theta), \tag{31}$$

and the projected dual update is defined in Equation 32:

$$\lambda_{i,k+1} = \max\left(0, \lambda_{i,k} + \alpha_\lambda \left[C_i(\theta_k) - \delta_i\right]\right), \tag{32}$$

where α_λ is the dual learning rate. Multipliers decrease when $C_i(\theta_k) < \delta_i$ and increase when $C_i(\theta_k) > \delta_i$, tightening enforcement on violations.

C. Safe Exploration via Constraint Prediction: A constraint prediction network $\hat{C}_\phi : S \times A \to \mathbb{R}^4$estimates expected constraint costs for candidate state–action pairs and is trained by supervised regression. For constraint i, the target is the Monte Carlo return defined in Equation 33:

$$y_i = \sum_{t'=t}^{T} \gamma^{(t'-t)} \, c_i(s_{t'}, a_{t'}), \tag{33}$$

where $c_i(s, a)$ is the immediate constraint cost. Training minimizes the MSE between $\hat{C}_\phi(s_t, a_t)$, y_i over-collected trajectories. At execution, actions are sampled from $\pi_\theta(a \mid s)$ are screened by $\hat{C}_\phi$; predicted violations beyond a safety margin are rejected and resampled; a conservative fallback is used. The safety rule is assumed in Equation 34:

$$a_t = \begin{cases} a \sim \pi_\theta(\cdot \mid s_t), & \text{If } \hat{C}_\phi(s_t, a) \le \delta_i + \epsilon_i, \forall i \\ a_{safe}, & \text{Otherwise} \end{cases}, \tag{34}$$

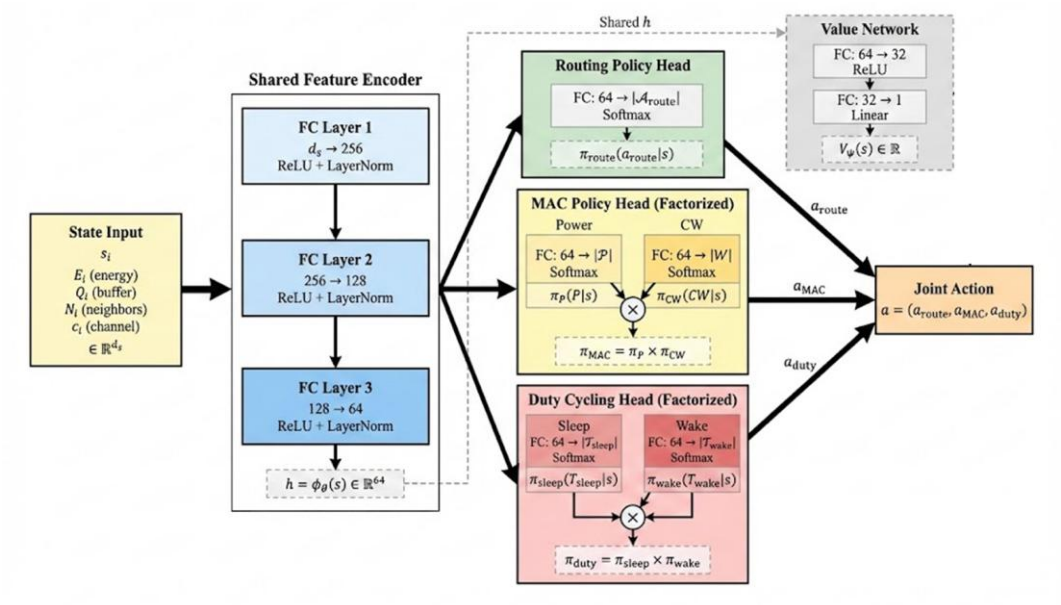
where $\epsilon_i > 0$ is a safety buffer and a_{safe} is a conservative action (*e.g.,* minimum power with maximum contention window).

4.4 Joint Policy Network for Routing–MAC–DCO

A. Neural Network: Joint policy uses a Deep Neural Network (DNN) with shared encoder and task-specific output heads (Figure 3). The shared encoder $\phi_\theta : S \to \mathbb{R}^{d_h}$ maps raw state inputs to latent features for routing, MAC, and DCO actions using fully connected layers with ReLU activations.

It has three hidden layers with dimensions $[d_s, 256, 128, 64]$ where d_s is a state dimension, and layer normalization is applied after each hidden layer. Encoder produces $h = \phi_\theta(s) \in \mathbb{R}^{64}$ and this feeds to policy heads.

Figure 3: Neural network for joint policy



Note: The architecture of the deep neural network used for the joint policy. It features a shared feature encoder that maps raw state inputs to a latent representation (h). This latent vector is then fed into three factorized, task-specific output heads: Routing (next-hop selection), MAC (power and contention window), and Duty Cycling (sleep and wake times), thereby reducing the overall action space complexity. A separate Value Network is used for advantage estimation

Three policy heads map the shared latent vector $h \in \mathbb{R}^{64}$to layer-specific action distributions. The routing head $\pi_{\text{route}}^{(\theta_r)}: \mathbb{R}^{64} \to \Delta(A_{\text{route}})$ outputs a categorical next-hop distribution using SoftMax as defined in Equation 35:

$$\pi_{\text{route}}(\, a_{\text{route}} \mid s; \theta\,) = \frac{\text{Exp}\big(f_{\text{route}}(h; \theta_r)_{a_{\text{route}}}\big)}{\sum_{a' \in A_{\text{route}}} \text{Exp}(f_{\text{route}}(h; \theta_r)_{a'})}, \tag{35}$$

where $f_{\text{route}}(h; \theta_r) \in \mathbb{R}^{|A_{\text{route}}|}$ is the routing logit vector.

The MAC head $\pi_{\text{MAC}}^{(\theta_m)}: \mathbb{R}^{64} \to \Delta(A_{\text{MAC}})$ factorizes the joint action (P, CW) into independent SoftMax distributions over power and contention window, Equation 36:

$$\pi_{\text{MAC}}(\, P, CW \mid s; \theta\,) = \pi_P(\, P \mid s; \theta_m^P\,) \cdot \pi_{CW}(\, CW \mid s; \theta_m^{CW}\,), \tag{36}$$

where π_P , π_{CW} are separate SoftMax distributions over power levels and contention windows, parameterized by disjoint subsets of MAC head weights θ_m^P and θ_m^{CW} . The duty-cycling head $\pi_{\text{duty}}^{(\theta_d)}: \mathbb{R}^{64} \to \Delta(A_{\text{duty}})$similarly factorizes $\big(T_{\text{sleep}}, T_{\text{wake}}\big)$as in Equation 37:

$$\pi_{\text{duty}}\big(\, T_{\text{sleep}}, T_{\text{wake}} \mid s; \theta\,\big) = \pi_{\text{sleep}}\big(\, T_{\text{sleep}} \mid s; \theta_d^{sleep}\,\big) \cdot \pi_{\text{wake}}\big(\, T_{\text{wake}} \mid s; \theta_d^{wake}\,\big). \tag{37}$$

The full joint policy is the product of head outputs, given by Equation 38:

$$\pi_\theta\left(a_{\text{route}}, a_{\text{MAC}}, a_{\text{duty}} \mid s\right) = \pi_{\text{route}}\left(a_{\text{route}} \mid s; \theta_r\right) \cdot \pi_{\text{MAC}}\left(a_{\text{MAC}} \mid s; \theta_m\right) \cdot \pi_{\text{duty}}\left(a_{\text{duty}} \mid s; \theta_d\right), \quad (38)$$

with $\theta = \{\theta_{enc}, \theta_r, \theta_m, \theta_d\}$. Actions are sampled independently from each head during exploration, and selected greedily by selecting each head's mode during deployment.

B. Policy Network Parameterization: Policy parameters $'\theta'$ are initialized with Xavier initialization to preserve gradient variance. The parameter count is nearly $256 \times 128 + 128 \times 64 + 64 \times (|A_{route}| + |P| + |W| + |T_{sleep}| + |T_{wake}|) \approx 45{,}000$ for typical cases with $|N(i)| \leq 8$. Entropy regularization is added to promote exploration via Equation 39:

$$J_{reg}(\theta) = J(\theta) + \beta_H \, \mathbb{E}_{s\sim d^{\pi_\theta}}\left[H\left(\pi_\theta(\cdot \mid s)\right)\right], \quad (39)$$

where $H(\pi_\theta(\cdot \mid s)) = -\sum_{a\in A} \pi_\theta(a \mid s) \log \pi_\theta(a \mid s)$ and β_H decays from 0.1 to 0.01 during training.

C. Value Network for Advantage Estimation: A value network $V_\psi(s)$ outputs a scalar estimate and is trained by minimizing the TD Bellman MSE in Equation 40:

$$L_V(\psi) = \mathbb{E}_{(s_t, r_t, s_{t+1})\sim D}\left[(V_\psi(s_t) - y_t)^2\right], \quad (40)$$

with $y_t = r_t + \gamma V_\psi(s_{t+1})$. Advantages of using GAE as defined in Equation 41:

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma \, \lambda_{GAE})^l \, \delta_{t+l}, \quad (41)$$

where $\delta_t = r_t + \gamma V_\psi(s_{t+1}) - V_\psi(s_t)$ and $\lambda_{GAE} = 0.95$.

D. Factorization for Action Space Reduction: The joint action space $|A| = |A_{route}| \times |A_{MAC}| \times |A_{duty}|$ is large (*e.g.,* $\approx 9 \times 9 \times 12 = 972$ for 8 neighbors), so factorization is used. The routing–MAC–duty product in Equation 38 reduces handling from $O(|A_{route}| \, |A_{MAC}| \, |A_{duty}|)$ to $O(|A_{route}| + |A_{MAC}| + |A_{duty}|)$, while cross-layer coupling is preserved via the shared encoder $h = \phi_\theta(s)$. Inside the MAC head, Equation 36 replaces $|P| \, |W|$ outputs with $|P| + |W|$, assuming a conditional near-independence of the assumed state. The policy-gradient term decomposes additively across heads as in Equation 42:

$$\begin{aligned}\nabla_\theta \log \pi_\theta(a \mid s) = \nabla_\theta \log \pi_{route}(a_{route} \mid s) + \nabla_\theta \log \pi_{MAC}(a_{MAC} \mid s) \\ + \nabla_\theta \log \pi_{duty}\left(a_{duty} \mid s\right),\end{aligned} \quad (42)$$

enabling efficient sampling and gradient computation with shared-encoder coupling retained.

4.5 Distributed Learning and Communication Overhead Control

A. Distributed Learning Paradigm: A hybrid distributed network is used: each SN runs a local policy for real-time decisions, while parameters are periodically synchronized via aggregation at the sink. The network is partitioned into K clusters; nodes in cluster k maintain local parameters θ_i, compute

updates from local trajectories and send them to a cluster head. Cluster heads aggregate updates and synchronize with the sink, which performs global updates and broadcasts refreshed parameters back to clusters.

B. Local Agent Design with Global Coordination: Each node n_i acts as an agent using π_{θ_i} and stores the experience in a buffer $B_i$up to B_{max}. When $B_i$is full or after T_{sync}, the node computes a local gradient as in Equation 43:

$$g_i = \frac{1}{|\ B_i\ |} \sum_{(s,a,r,s')\in B_i} \nabla_\theta \log \pi_\theta\ (\ a\ |\ s\ )\ \hat{A}(s,a), \tag{43}$$

where advantage estimates $\hat{A}(s,a)$ are computed using the local value network and stored trajectory rewards. Cluster heads aggregate node gradients with data-size weighting (Equation 44):

$$g_k = \sum_{i\in V_k} \frac{|\ B_i\ |}{\sum_{j\in V_k} |\ B_j\ |}\ g_i, \tag{44}$$

where the weighting reflects each node's relative contribution to the data. The sink aggregates across clusters (Equation 45):

$$g_{global} = \sum_{k=1}^{K} \frac{|\ V_k\ |}{N}\ g_k, \tag{45}$$

and updates $\theta \leftarrow \theta + \alpha_\theta g_{global}$, then broadcasts the parameters back to all nodes via the cluster heads.

C. Asynchronous Updates and Staleness Control: To avoid synchronization delays, nodes can transmit gradients asynchronously with version/timestamp metadata. Cluster heads accept updates only if $t - \tau_i \le \tau_{max}$ and down-weight stale gradients using Equation 46:

$$g_k = \sum\nolimits_{i\in\mathcal{V}_k} w_i\ (t-\tau_i)\cdot g_i, \tag{46}$$

where the staleness weight function is defined as $w_i(\Delta t) = \text{Exp}\ (-\lambda_{stale}\Delta t)$with decay rate $\lambda_{stale} = 0.1$.

D. Communication-Efficient Gradient Aggregation: To reduce bandwidth for $|\ \theta\ | \approx 45{,}000$, gradients are sparsified by sending only top-k magnitudes (Equation 47):

$$\mathcal{S}_i = \{j : |\ g_i[j]\ | \geq \text{threshold}(\{|\ g_i[m]\ |\}_{m=1}^{|\theta|}, k)\}, \tag{47}$$

where threshold$(\cdot, k)$ returns the k^{th} largest value in the set and quantized with b-bit stochastic rounding (Equation 48):

$$\hat{g}_i[j] = \text{Sign}(g_i[j]) \cdot \left\lceil \frac{|\ g_i[j]\ |}{s} \right\rceil s, \tag{48}$$

where $s = \mathrm{Max}_j \mid g_i[j] \mid/(2^b - 1)$ is the quantization scale factor transmitted alongside quantized values.

E. Bandwidth Overhead and Adaptive Synchronization: Per-cycle upload cost for sparse quantized gradients is assumed by Equation 49:

$$C_{upload}^{i} = L_{header} + k\ (b + Log_{\,2} \mid \theta \mid), \tag{49}$$

where L_{header} is the packet header size in bits, k is the number of transmitted gradient components, b is the quantization bit-width, and $\mathrm{Log}_{\,2} \mid \theta \mid$ bits encode each component index.

while the full-precision download cost is $C_{download}^{i} = L_{header} + \mid \theta \mid \cdot 32$ bits. The synchronization interval adapts to congestion using Equation 50:

$$T_{sync}(t+1) = \mathrm{Min}\ \left(T_{sync}^{max}, T_{sync}(t)\left(1+\eta_{adapt}\, I_{\bar{Q}(t)>Q_{thresh}}\right)\right), \tag{50}$$

where T_{sync}^{max} is the maximum allowed interval, $\eta_{adapt} = 0.1$ is the adaptation rate, $\bar{Q}(t)$ is the average buffer occupancy, $Q_{thresh} = 0.7$ is the congestion threshold, and $\mathbb{I}_{\{\,.\,\}}$ is the indicator function.

4.6 Training Procedure and Complexity Analysis

A. Training Algorithm: Training integrates constrained multi-objective policy optimization with distributed learning and safe exploration. Learning runs in episodes of duration $T_{episode}$, where nodes execute the current policy while the network processes traffic, and training continues until convergence.

Each iteration cycles through three phases: (a) trajectory collection via distributed policy execution, where nodes sample actions, observe transitions, and log rewards and constraint costs into local replay buffers; (b) local gradient computation, applying the primal–dual updates in Equations 30 and 32 using mini-batches, with advantage-based policy gradients and Lagrange multiplier updates from constraint violations, followed by gradient sparsification/quantization to cut communication; and (c) global synchronization, where the sink aggregates updates using Equations 44 and 45 update shared parameters and broadcast them back to nodes. Termination occurs when $\| \nabla_\theta \| < \epsilon_{grad} = 10^{-4}$and constraint violations satisfy $\max_i [C_i(\theta) - \delta_i]_+ < \epsilon_{const} = 0.01$over consecutive evaluation episodes.

Algorithm 1: Constrained Multi-Objective RL for Joint Cross-Layer Control

Input: WSN environment, initial policy parameters θ₀, initial Lagrange multipliers λ₀

Output: Trained policy π*_θ satisfying constraints

1: Initialize policy network θ ← θ₀, value network ψ ← ψ₀, multipliers λ ← λ₀
2: Initialize replay buffers B_i ← ∅ for all nodes i ∈ V
3: For episode k = 1 to K_max Do
4: // Trajectory Collection Phase
5: For Each node n_i ∈ V in parallel Do
6: Reset local state s_i ← initial state

7: For timestep t = 1 to T_episode Do
8: Sample action: a_i,t ~ π_θ(·|s_i,t)
9: Apply constraint prediction safety check (Eq. 34)
10: Execute action, observe reward r_i,t, next state s_i,t+1, constraint costs c_i,t
11: Store transition (s_i,t, a_i,t, r_i,t, s_i,t+1, c_i,t) in B_i
12: End For
13: End For
14:
15: *// Local Gradient Computation Phase*
16: For Each node n_i ∈ V in parallel Do
17: Sample mini-batch M from B_i
18: Compute advantages Â using GAE (Equation 41)
19: Compute policy gradient: g_i^π ← ∇_θ L(θ, λ) (Equation 27)
20: Compute value loss gradient: g_i^V ← ∇_ψ L_V(ψ) (Equation 40)
21: Apply gradient sparsification (Equation 47) and quantization (Equation 48)
22: Transmit compressed gradient (g_i^π, g_i^V) to cluster head
23: End For
24:
25: *// Global Synchronization Phase*
26: For each cluster head c_k, do
27: Aggregate gradients from cluster: g_k ← Σ w_i g_i (Equation 44)
28: Transmit cluster gradient g_k to sink
29: End For
30:
31: At sink node:
32: Aggregate cluster gradients: g_global ← Σ (|V_k|/N) g_k (Equation 45)
33: Update policy: θ ← θ + α_θ g_global^π
34: Update value network: ψ ← ψ + α_ψ g_global^V
35:
36: *// Dual Variable Update*
37: Evaluate constraint costs: C_i(θ) for i ∈ {1,2,3,4}
38: Update multipliers: λ_i ← Max(0, λ_i + α_λ[C_i(θ) - δ_i]) (Equation 32)
39:
40: Broadcast updated parameters (θ, ψ, λ) to all cluster heads
41: Cluster heads disseminate to nodes in their clusters
42:
43: *// Convergence Check*
44: If ||g_global|| < ε_grad and max_i[C_i(θ) - δ_i]_+ < ε_const Then
45: Break
46: End If
47: End For
48: Return Trained policy π*_θ

B. Hyperparameter Configuration: The training procedure involves numerous hyperparameters that control learning dynamics, exploration behavior, and constraint enforcement. The hyperparameter settings are determined using a preliminary grid search over critical parameters and fixed based on empirical performance across diverse network scenarios. Table 1 summarizes the complete hyperparameter configuration employed throughout experimentation.

Table 1: Hyperparameter Configuration for Training

Category	Parameter	Symbol	Value
Network Architecture	Encoder hidden dimensions	$[h_1, h_2, h_3]$	$[256, 128, 64]$
	Value network dimensions	$[h_v^1, h_v^2]$	$[64, 32]$
	Activation function	-	ReLU
Learning Rates	Policy learning rate	α_θ	3×10^{-4}
	Value learning rate	α_ψ	1×10^{-3}
	Dual learning rate	α_λ	1.5×10^{-3}
	Dual-primal ratio	κ	5.0
Discount and GAE	Discount factor	γ	0.99
	GAE parameter	λ_{GAE}	0.95
Exploration	Initial entropy coefficient	β_H^{init}	0.1
	Final entropy coefficient	β_H^{final}	0.01
	Entropy decay rate	-	0.995
Training	Episode length	$T_{episode}$	1000
	Maximum episodes	K_{max}	5000
	Mini-batch size	$\| M \|$	128
	Replay buffer size	B_{max}	10000
Constraints	Energy threshold	δ_1	$0.2E_{init}$
	Buffer threshold	δ_2	$0.9B_i$
	Latency threshold	δ_3	500ms
	PDR threshold	δ_4	0.85
Distributed Learning	Synchronization interval	T_{sync}	50
	Clusters	K	4
	Gradient sparsity	k	4500
	Quantization bits	b	8
	Staleness decay	λ_{stale}	0.1
Normalization	Reward momentum	β	0.99
	Epsilon	ϵ	10^{-8}
Convergence	Gradient threshold	ϵ_{grad}	10^{-4}
	Constraint threshold	ϵ_{const}	0.01

Note: Summary of the hyperparameter settings used for training the CMORLM in the NS-3 simulation environment. The table details the neural network architecture dimensions, learning rates for the primal-dual updates, exploration coefficients, and specific thresholds for energy, buffer, latency, and reliability constraints. EC: Energy Consumption; EEL: End-to-End Latency; PDR: Packet Delivery Ratio; NL: Network Lifetime; CVR: Constraint Violation Rate; SN: Sensor Node; TLP: Traditional Layered Protocols; CMORLM: Constrained Multi-Objective Reinforcement Learning Model. All reported values are averaged over multiple simulation runs in NS-3.

The learning rate schedule uses constant rates throughout training rather than adaptive decay, since entropy regularization and diminishing exploration naturally stabilize learning in later episodes. The dual learning rate is set 5 times larger than the primal rate ($\kappa = 5$) to ensure Lagrange multipliers adapt quickly to constraint violations, maintaining feasibility throughout training.

C. Computational Complexity Analysis: Computational complexity is expressed using state dimension d_s, action sizes, network size N, and training horizon. Per timestep, each node performs policy/value forward evaluation, action sampling, and (during update rounds) gradient computation.

Forward evaluation through the shared encoder costs $O(d_s)$ (Three FC layers: $O(d_s \times 256 + 256 \times 128 + 128 \times 64)$). The three heads add $O\big(64 \mid A_{route} \mid +64(\mid P \mid +\mid W \mid) + 64(\mid T_{sleep} \mid +\mid T_{wake} \mid)\big)$, which is $O(1)$ for typical settings (*e.g.,* $\mid A_{route} \mid \approx 8, \mid P \mid = 3, \mid W \mid = 3, \mid T_{sleep} \mid = 4, \mid T_{wake} \mid = 3$). Thus, the total forward pass per node is $O(d_s)$.

Backpropagation matches the forward order, giving per-update cost $O(\square \mid \theta \mid d_s)$ with $\mid \theta \mid \approx 45{,}000$. Advantage estimation via GAE over trajectories of length $T_{episode} = 1000$ costs $O(T_{episode})$. Hence, per-episode training at each node is $O(T_{episode} d_s + \mid \theta \mid d_s)$.

Aggregation scales with sparse gradient size k after sparsification. A cluster head aggregating $\mid V_k \mid$ node updates costs $O(\mid V_k \mid k)$ with $k \approx 4500$; the sink aggregates $K = 4$ clusters in $O(Kk)$. Network-wide aggregation per synchronization is $O(Nk)$.

Sparse gradient uploads and parameter broadcasts dominate communication. Each node transmits $O(k(b + \log_2 \mid \theta \mid))$ bits per sync (quantization $b = 8$); amortized over $T_{sync} = 50$ steps this is $O(k/T_{sync}) \approx 90$ bits per step. Broadcasting parameters costs $O(\mid \theta \mid)$ bits per sync, amortized to $O(\mid \theta \mid /T_{sync}) \approx 900$ bits per step.

Across $K_{max} = 5000$ episodes of $T_{episode} = 1000$ steps, total compute is $O(K_{max} T_{episode} N d_s) \approx O(5 \times 10^9 d_s)$ for $N = 100$. The learned policy runs in $O(d_s)$ per node per timestep, supporting real-time inference with sub-10 *ms* forward execution on constrained nodes.

5 Experimental Set-up

5.1 Simulation Environment and Configuration

Experiments are conducted in NS-3 v3.35 using IEEE 802.15.4 PHY/MAC. Propagation follows a log-distance path loss model with indoor exponent $\alpha = 3.5$. Nodes emulate TelosB constraints (10 kB RAM, 48 *kB* flash, 2700 *mAh* at 3 V). Energy uses $E_{elec} = 50$ *nJ/bit*, $\epsilon_{amp} = 100$ *pJ/bit/m*^{2}, transmit power 0–10 *dBm*, $P_{idle} = 0.426$ *mW*, $P_{sleep} = 0.003$ *mW*, and $E_{switch} = 0.05$ *mJ*. Topologies include (i) a 10 × 10 grid with 50 *m* spacing, (ii) 100 random nodes in 500 × 500 *m*, and (iii) a clustered layout with 4 clusters of 25 nodes. The sink is placed at the perimeter for grid/random layouts and at the center for clustered layouts. The timestep is 10 *ms*, and each run lasts 5000 *Sec.*

5.2 Workloads and Network Scenarios

Periodic sensing uses CBR 100-byte packets every 5, 10, 20 *Sec.* (Heavy/Moderate/Light). Event-driven traffic follows Poisson arrivals with $\lambda_{event} = 0.1$ *events/s*, each triggering 5-packet bursts. Environmental monitoring combines a 20-*second.* periodic background with event bursts having exponential inter-arrival (Mean 60 *Sec.*). Network size varies from 50–200 nodes (baseline: 100,

average degree: 6.8), with 20% of nodes as anchors at maximum power. Robustness scenarios include 10% random node failures at $t = 2000$ *Sec*, a mobile sink moving at 1 *m/s*, and a load shift from light to heavy at $t = 1500$s. QoS uses $D_{max} = 500$ms (critical) or 2000 *ms* (tolerant) and $PDR_{min} = 0.9$ (sensitive) or 0.7 (best-effort), including two-class heterogeneous (D_{max}, PDR_{min})settings.

5.3 Baselines and Comparative Methods

A. Baseline 1: TLP: RPL(ETX) + ContikiMAC with fixed $CW = 16$, *max* power, and static 10% duty cycle.

B. Baseline 2: EAH Cross-Layer: Heuristic cross-layer scheme with energy-weighted routing metric $M = w_E(1 - E_i/E_{init}) + w_h h_i (w_E = 0.6, w_h = 0.4)$, link-quality-based power control, and buffer-threshold duty cycling (>50% increase, <20% decrease).

C. Baseline 3: Q-LRO: Tabular Q-learning for routing only ($\alpha = 0.1$, $\gamma = 0.95$, ϵ-greedy 0.3→0.05); MAC and duty fixed.

D. Baseline 4: DQN-JA for EC: DQN with joint routing–MAC–duty actions for single-objective EC (FC [128,64,32], buffer 10 *k*, target update 100, LR 10^{-3}).

E. Baseline 5: UMORL with the proposed rewards but without primal–dual constraint handling.

5.4 Evaluation Metrics and Statistical Protocol

Primary metrics are energy per PDR (Equation (6)), EEL (Equation 7), PDR within 5 *Sec.* (Equation 8), and NL (time until the first node falls below as E_{min}). Constraint satisfaction is reported as a normalized violation rate over energy ($E_i(t) < E_{min}$), buffer overflow drops, latency deadline misses ($> D_{max}$), and reliability violations ($PDR < PDR_{min}$). Learning efficiency includes convergence speed (episodes to 95% final), sample efficiency (interactions to converge), and stability (return variance over the last 100 episodes). Each setting uses 10 independent seeds with *mean* ± *std*; significance uses two-sample t-tests with Bonferroni correction ($p < 0.01$) and Cohen's d. Robustness is tested via ±50% sweeps on $'w'$, $'\delta'$, and learning rates.

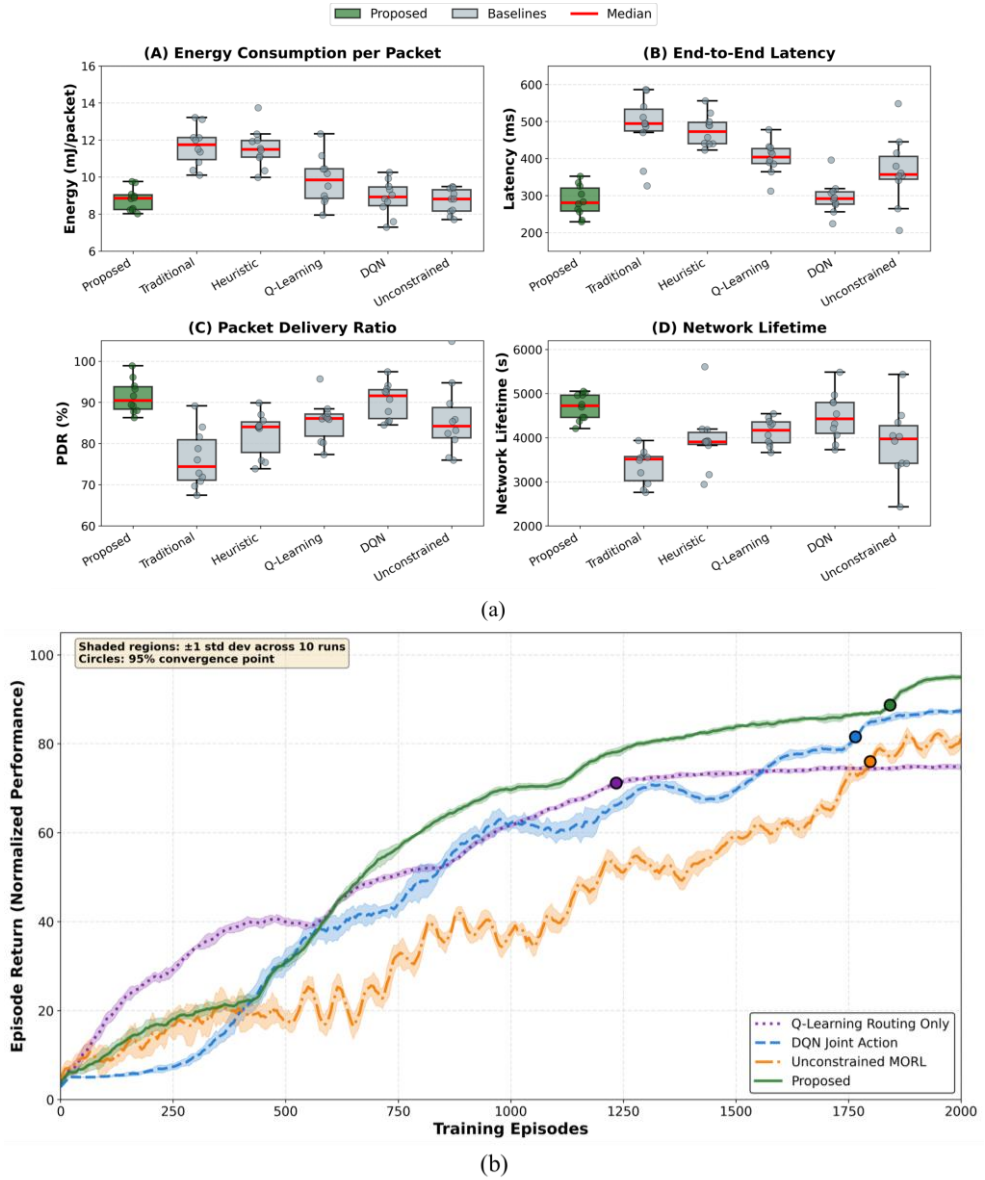
6 Results

6.1 Overall Performance Comparison

As shown in Table 2 and Figure 4 (a), the proposed CMORLM achieves EC of 8.42 ± 0.31 *mJ/packet* compared to 12.78 ± 0.52 *mJ/packet* for the TLP and 11.84 ± 0.48 *mJ/packet* for the EAH Cross-Layer, representing 34.2% and 28.7% reductions. Q-LRO achieves 10.21 ± 0.44 *mJ/packet*, DQN-JA consumes 9.15 ± 0.38 *mJ/packet*, and UMORL achieves 8.51 ± 0.34 *mJ/packet*.

EEL for proposed CMORLM is 287 ± 18 *ms vs.* 489 ± 24 *ms* for TLP (41.3% reduction), 470 ± 22 *ms* for Energy Aware Heuristic (EAH) (38.9% reduction), 412 ± 28 *ms* for Q-Learning Routing Only (Q-LRO), and 298 ± 19 *ms* for DQN Joint Action (DQN-JA) and 356 ± 31 *ms* for UMORL. The proposed CMORLM achieves 24.0% lower latency than UMORL ($p<0.001$). PDR reaches 91.2 ± 1.4% for proposed CMORLM compared to 78.3 ± 2.8% for TLP, 82.7 ± 2.1% for EAH, 84.5 ± 2.4% for Q-LRO, and 88.9 ± 1.7% for DQN-JA and 83.1 ± 3.2% for UMORL. Cohen's d effect size between the proposed CMORLM and UMORL is 3.12.

Figure 4: Performance Comparison and Convergence Analysis



Note: Comprehensive performance evaluation of the proposed CMORLM against five baselines (Traditional Layered Protocol [TLP], Energy-Aware Heuristic [EAH], Q-Learning Routing Only [Q-LRO], DQN Joint Action [DQN-JA], and Unconstrained MORL [UMORL]). (A) Energy consumption per packet, (B) End-to-End Latency, (C) Packet Delivery Ratio, and (D) Network Lifetime. Box plots represent the distribution over 10 independent simulation runs. The convergence curves (bottom) demonstrate the training progression of RL-based methods over 2000 episodes.

Table 2: Overall Performance Metrics

Method	EC (mJ/pkt)	EEL (ms)	PDR (%)	NL (s)	Convergence (episodes)
Proposed CMORLM	**8.42 ± 0.31***	**287 ± 18***	**91.2 ± 1.4***	**4782 ± 142***	1842 ± 127
TLP	12.78 ± 0.52	489 ± 24	78.3 ± 2.8	3456 ± 218	N/A
EAH	11.84 ± 0.48	470 ± 22	82.7 ± 2.1	3644 ± 196	N/A
Q-LRO	10.21 ± 0.44	412 ± 28	84.5 ± 2.4	4021 ± 174	1234 ± 98
DQN-JA	9.15 ± 0.38	298 ± 19	88.9 ± 1.7	4512 ± 158	1765 ± 134
UMORL	8.51 ± 0.34	356 ± 31	83.1 ± 3.2	3892 ± 287	1798 ± 119

Note: Quantitative comparison of the proposed CMORLM against baseline methods across key metrics: Energy Consumption (EC), End-to-End Latency (EEL), Packet Delivery Ratio (PDR), Network Lifetime (NL), and convergence speed. Values are presented as

Mean ± Standard Deviation over 10 independent runs. * indicates statistical significance ($p<0.01$) compared to all baselines using a two-sample t-test with Bonferroni correction.

NL extends to 4782 ± 142 *seconds* under the proposed CMORLM, vs. 3456 ± 218 *seconds* for TLP (38.4% improvement), 3644 ± 196 *seconds* for EAH (31.2% improvement), 4021 ± 174 *seconds* for Q-LRO, 4512 ± 158 *seconds* for DQN-JA, and 3892 ± 287 *seconds* for UMORL. The proposed CMORLM requires 1842 ± 127 episodes to reach 95% of final performance (Figure 4b), which is comparable to 1765 ± 134 episodes for DQN-JA and 1798 ± 119 episodes for UMORL. Q-LRO converges in 1234 ± 98 episodes. Sample efficiency is 2.84×10^8 timesteps, and this is within 8% of UMORL.

6.2 Multi-Objective Trade-off and Pareto Analysis

The proposed CMORLM generates distinct Pareto-optimal solutions across variable objective weight configurations (Table 3). Energy prioritized configuration with weights w = [0.7,0.15,0.15] achieves 7.21 ± 0.28 *mJ/packet* EC, 342 ± 21 *ms* EEL, and 88.4 ± 1.6% PDR. Balanced configuration with w = [0.33,0.34,0.33] achieves 8.42 ± 0.31 *mJ/packet* and 287 ± 18 *ms* EEL and 91.2 ± 1.4% PDR. Reliability prioritized configuration with w = [0.15,0.15,0.7] achieves 9.87 ± 0.36 *mJ/packet*, 251 ± 16 *ms,* and 94.6 ± 1.2% PDR.

Table 3: Pareto Solutions with Different Weight Configurations

Weight Configuration [w_1, w_2, w_3]	**EC (mJ/pkt)**	**EEL (ms)**	**PDR (%)**	**NL (s)**
Energy-prioritized [0.7, 0.15, 0.15]	7.21 ± 0.28	342 ± 21	88.4 ± 1.6	5124 ± 156
Balanced [0.33, 0.34, 0.33]	8.42 ± 0.31	287 ± 18	91.2 ± 1.4	4782 ± 142
Latency-prioritized [0.15, 0.7, 0.15]	10.34 ± 0.42	214 ± 14	89.7 ± 1.8	4156 ± 178
Reliability-prioritized [0.15, 0.15, 0.7]	9.87 ± 0.36	251 ± 16	94.6 ± 1.2	4298 ± 164
DQN-JA (Energy-Only)	9.15 ± 0.38	298 ± 19	88.9 ± 1.7	4512 ± 158
UMORL [0.33, 0.34, 0.33]	8.51 ± 0.34	356 ± 31	83.1 ± 3.2	3892 ± 287

Note: Performance metrics of the CMORLM under various Pareto-optimal objective weight configurations (w1,w2,w3). The weight vector components correspond to preferences for [Energy Efficiency, Latency Reduction, Reliability]. This demonstrates the framework's capability as a decision-support tool, allowing network operators to dynamically prioritize specific performance metrics based on application requirements. Values are Mean ± Standard Deviation over 10 runs.

EEL prioritized configuration with $w = [0.15,0.7,0.15]$ achieves minimum EED of 214 ± 14 *ms*. EC increases to 10.34 ± 0.42 *mJ/packet,* and PDR reduces to 89.7 ± 1.8%. The energy EEL trade-off shows a nearly 43% increase in energy for a 37.4% reduction in EEL when shifting from an energy-prioritized to an EEL-prioritized network. The energy reliability trade-off shows that a 36.9% increase in energy generated, @ 7.0% improvement in PDR relative to energy-to-reliability-prioritized settings. Energy prioritized configuration extends NL to 5124 ± 156 *seconds*. This surpasses the balanced configuration @ 7.1% and the EEL prioritized configuration by 23.3%. Reliability-prioritized configuration achieves 94.6 ± 1.2% PDR, representing a 6.5% improvement over DQN-JA at 88.9 ± 1.7% PDR. Balanced configuration provides intermediate performance across all metrics.

DQN-JA, which optimizes only EC, achieves 9.15 ± 0.38 *mJ/packet*. UMORL with balanced weights achieves 8.51 ± 0.34 *mJ/packet* and 356 ± 31 *ms* EEL. PDR degrades to 83.1 ± 3.2%. Proposed CMORLM's balanced configuration achieves 9.8% higher PDR than UMORL. Pareto frontier spans EC range [7.21, 10.34] *mJ/packet*, EEL range [214, 342] *ms,* and PDR range [88.4, 94.6]%. Weight adjustment enables navigation along the frontier. Energy-constrained deployments select $w = [0.7,0.15,0.15]$ and EEL critical applications select $w = [0.15,0.7,0.15]$ and reliability sensitive applications select $w = [0.15,0.15,0.7]$.

6.3 Constraint Satisfaction and Safety Verification

The proposed CMORLM maintains an aggregate constraint violation rate of 0.8 ± 0.3% compared to 18.4 ± 2.7% for UMORL and 12.6 ± 2.1% for DQN-JA (Table 4). Energy CVR in 0.2 ± 0.1% of node timestep pairs for proposed CMORLM compared to 8.7 ± 1.4% for UMORL and 5.3 ± 1.2% for DQN-JA. The primal dual optimization mechanism enforces a minimum RE threshold as $E_{min} = 0.2E_{init}$ across all nodes throughout the operation.

Table 4: Constraint Violation Rates (CVR)

Method	Energy Violations (%)	Buffer Violations (%)	Latency Violations (%)	Reliability Violations (%)	Aggregate Rate (%)
Proposed CMORLM	**0.2 ± 0.1**	**0.4 ± 0.2**	**1.1 ± 0.4**	**1.2 ± 0.5**	**0.8 ± 0.3**
TLP	4.2 ± 0.8	2.8 ± 0.6	32.4 ± 3.2	28.7 ± 2.9	17.0 ± 1.6
EAH	3.1 ± 0.7	2.1 ± 0.5	26.8 ± 2.8	22.3 ± 2.4	13.6 ± 1.4
Q-LRO	2.8 ± 0.6	3.4 ± 0.7	21.2 ± 2.5	18.9 ± 2.2	11.6 ± 1.3
DQN-JA	5.3 ± 1.2	7.2 ± 1.6	14.8 ± 2.1	13.1 ± 1.9	12.6 ± 2.1
UMORL	8.7 ± 1.4	11.6 ± 2.2	24.3 ± 2.6	28.9 ± 3.1	18.4 ± 2.7

Note: Comparison of Constraint Violation Rates (CVR) across different methods. Violations are tracked for Energy (residual energy dropping below minimum threshold), Buffer (packet drops due to overflow), Latency (exceeding maximum delay bounds), and Reliability (falling below required PDR). The proposed CMORLM strictly enforces these limits during training, resulting in the lowest aggregate violation rate. Values are Mean ± Standard Deviation over 10 runs.

Buffer overflow violations reach 0.4 ± 0.2% for proposed CMORLM compared to 11.6 ± 2.2% for UMORL and 7.2 ± 1.6% for DQN-JA. TLP exhibits 2.8 ± 0.6% buffer violations, EAH shows 2.1 ± 0.5%, and Q-LRO experiences 3.4 ± 0.7% violations. Latency CVR in 1.1 ± 0.4% of PDR for proposed CMORLM *vs.* 24.3 ± 2.6% for UMORL and 14.8 ± 2.1% for DQN-JA and 32.4 ± 3.2% for TLP. Maximum EEL bound $D_{max} = 500$ *ms* is maintained for 98.9% of PDR under the proposed CMORLM. Lagrange multiplier λ_3, EEL constraint stabilizes at 0.42 ± 0.08 during training.

Reliability constraint violations occur in 1.2 ± 0.5% of evaluation windows for proposed CMORLM compared to 28.9 ± 3.1% for UMORL and 13.1 ± 1.9% for DQN-JA. EAH violates reliability requirements in 22.3 ± 2.4% of windows, and TLP experiences a 28.7 ± 2.9% violation rate. The proposed CMORLM maintains PDR above threshold across 98.8% of operation duration where PDR falls below $PDR_{min} = 0.85$. The aggregate violation rate for proposed CMORLM (0.8 ± 0.3%) is 23× lower than UMORL (18.4 ± 2.7%) and 15.8× lower than DQN-JA (12.6 ± 2.1%). TLP and EAH achieve 17.0 ± 1.6% and 13.6 ± 1.4% aggregate rates. All pairwise comparisons between proposed CMORLM *vs.* baselines have $p < 0.001$. Lagrange multiplier evolution during training shows λ_1 for energy constraint converges to 0.28 ± 0.05, and λ_2 for buffer constraint to 0.18 ± 0.04, and λ_3 for EEL

constraint to 0.42 ± 0.08, and λ_4 for the reliability constraint to 0.35 ± 0.06. Dual learning rate ratio $\kappa = 5.0$is used.

6.4 Ablation Study and Component Analysis

The proposed CMORLM proves consistent performance across diverse network configurations, traffic patterns, quality of service requirements, and dynamic operating conditions (Table 5). For scalability analysis, network sizes from 50 to 200 nodes are tested across three topology types (Figure 6). EC for the 100-node random topology baseline is 8.42 ± 0.31 mJ/packet, increasing to 9.21 ± 0.38 mJ/packet at 200 nodes (17.5% increase) and decreasing to 7.84 ± 0.29 *mJ/packet* at 50 nodes (6.9% reduction).

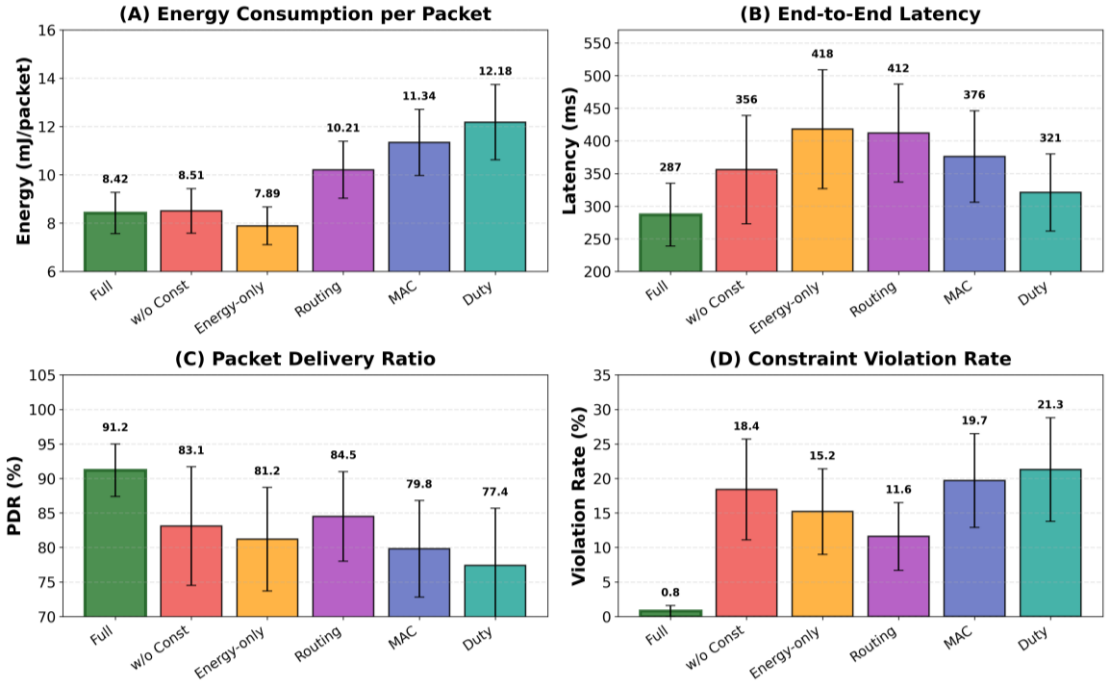
Table 5: Network Configuration and Traffic Pattern Analysis

Configuration	EC (mJ/pkt)	EEL (ms)	PDR (%)	NL (s)
Scalability				
50 nodes (random)	7.84 ± 0.29	264 ± 16	92.1 ± 1.2	5234 ± 168
100 nodes (random)	8.42 ± 0.31	287 ± 18	91.2 ± 1.4	4782 ± 142
150 nodes (random)	8.96 ± 0.35	302 ± 20	90.8 ± 1.6	4456 ± 178
200 nodes (random)	9.21 ± 0.38	318 ± 22	90.3 ± 1.7	4198 ± 186
Topology (100 nodes)				
Grid (10×10, 50m spacing)	8.12 ± 0.28	276 ± 17	91.8 ± 1.3	4921 ± 152
Random (500×500m)	8.42 ± 0.31	287 ± 18	91.2 ± 1.4	4782 ± 142
Clustered (4×25 nodes)	8.67 ± 0.34	294 ± 19	90.6 ± 1.6	4634 ± 164
Traffic Pattern				
Periodic (20s interval)	7.92 ± 0.27	268 ± 15	92.4 ± 1.2	5042 ± 158
Event-driven (Poisson λ=0.1)	8.76 ± 0.36	301 ± 21	90.1 ± 1.8	4512 ± 172
Environmental (periodic+events)	8.42 ± 0.31	287 ± 18	91.2 ± 1.4	4782 ± 142

Note: Robustness and sensitivity analysis of the proposed CMORLM under varying operational conditions. The model is evaluated across different network scales (50 to 200 nodes), spatial topologies (Grid, Random, Clustered), and traffic generation patterns (Periodic, Event-driven, Environmental). The baseline configuration uses 100 nodes with a random topology and environmental traffic. Values are Mean ± Standard Deviation over 10 runs.

To quantify the contribution of each protocol layer and objective, an ablation study was conducted, with the results summarized in Figure 5. For energy-only optimization, it achieves 7.89 ± 0.29 mJ/packet (6.3% reduction vs. full method) (see Figure 5A), 418 ± 34 ms EEL (45.6% increase) (Figure 5B), 81.2 ± 2.8% PDR (11.0% degradation) (Figure 5C), and 15.2 ± 2.3% CVR (19× increase) (Figure 5D). As further depicted in Figure 5, for routing only optimization with fixed MAC and duty cycling, it achieves 10.21 ± 0.44 *mJ/packet*, 412 ± 28 *ms* EEL, 84.5 ± 2.4% PDR, and 11.6 ± 1.8% CVR. MAC-only optimization achieves 11.34 ± 0.51 *mJ/packet*, 376 ± 26 *ms* EEL, 79.8 ± 2.6% PDR, and 19.7 ± 2.5% CVR. Duty only optimization achieves 12.18 ± 0.58 *mJ/packet*, 321 ± 22 *ms* EEL, 77.4 ± 3.1% PDR, and 21.3 ± 2.8% CVR.

Figure 5: Ablation Study Results



Note: Ablation study evaluating the contribution of individual components in the CMORLM framework. The x-axis represents different model variants: 'Full' (proposed joint method), 'w/o Const' (model without Lagrangian constraint handling), and single-layer optimizations ('Energy-only', 'Routing', 'MAC', 'Duty'). The metrics evaluated are (A) Energy Consumption, (B) End-to-End Latency, (C) Packet Delivery Ratio, and (D) Constraint Violation Rate. Results confirm that joint cross-layer optimization with constraint handling yields the best balanced performance.

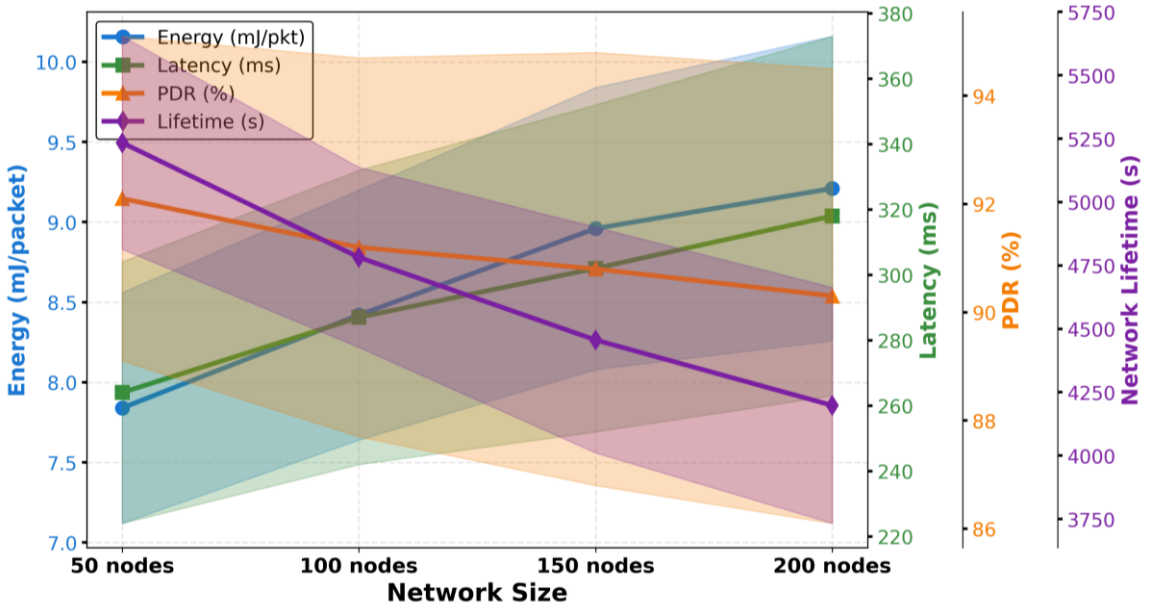
The full method achieves 21.3% lower energy than routing-only, 34.7% lower than MAC-only, and 44.7% lower than duty-only. EEL reductions are 43.6% *vs.* routing-only, 31.0% *vs.* MAC-only, and 11.9% *vs.* duty-only. PDR improvements are 7.9%, 14.3%, and 17.8% over routing-only, MAC-only, and duty-only. CVR for the full method at 0.8% compare to 11.6% for routing only, 19.7% for MAC only, and 21.3% for duty only. Crucially, as shown by the 'w/o Const' bars in Figure 5, the configuration without constraint handling achieves 18.4% CVR with 24.0% higher EEL and 9.7% lower PDR than the full method, despite a comparable EC of 8.51 *mJ/packet*. Statistical significance testing confirms all pairwise comparisons between the full method and ablated configurations across EC, EEL, PDR, and CVR metrics.

6.5 Scalability, Robustness, and Sensitivity Analysis

The proposed CMORLM proves consistent performance across diverse network configurations, traffic patterns, QoS requirements, and dynamic operating conditions (Table 6). For scalability analysis, network sizes from 50 to 200 nodes are evaluated across three topology types (Figure 6).

EC for the 100-node random topology baseline is 8.42 ± 0.31 *mJ/packet*, increasing to 9.21 ± 0.38 *mJ/packet* at 200 nodes (17.5% increase) and decreasing to 7.84 ± 0.29 *mJ/packet* at 50 nodes (6.9% reduction).

Figure 6. Scalability analysis



Note: Scalability analysis illustrating the impact of network size (ranging from 50 to 200 nodes) on four key performance indicators: Energy Consumption (blue line), End-to-End Latency (green line), Packet Delivery Ratio (orange line), and Network Lifetime (purple line). The shaded areas represent the standard deviation across 10 independent runs, demonstrating the model's robust performance under increasing network density.

For topology variations, grid deployment achieves 8.12 ± 0.28 *mJ/packet* with 91.8 ± 1.3% PDR, random topology achieves 8.42 ± 0.31 *mJ/packet* with 91.2 ± 1.4% PDR, and clustered deployment achieves 8.67 ± 0.34 *mJ/packet* with 90.6 ± 1.6% PDR. Grid topology has regular node spacing and predictable neighbor connectivity. Clustered topology exhibits 6.8% higher EC. For traffic pattern analysis, periodic sensing at 20-*second* intervals achieves 7.92 ± 0.27 *mJ/packet*, 268 ± 15 ms EEL, and 92.4 ± 1.2% PDR. Event-driven monitoring with Poisson arrivals increases energy to 8.76 ± 0.36 *mJ/packet* and EEL to 301 ± 21 ms, while reducing PDR to 90.1 ± 1.8%. Environmental monitoring combining periodic background with event bursts achieves intermediate performance at 8.42 ± 0.31 *mJ/packet*, 287 ± 18 *ms* EEL, and 91.2 ± 1.4% PDR.

Table 6: QoS Requirements and Robustness Analysis

Scenario	EC (mJ/pkt)	EEL (ms)	PDR (%)	CVR (%)
QoS Variations				
Latency-Critical (D_max = 500 *ms*)	8.42 ± 0.31	287 ± 18	91.2 ± 1.4	0.8 ± 0.3
Delay-Tolerant (D_max=2000 *ms*)	7.68 ± 0.26	412 ± 28	93.1 ± 1.2	0.4 ± 0.2
Reliability-Sensitive (PDR ≥ 0.9)	9.12 ± 0.34	274 ± 17	93.8 ± 1.1	0.6 ± 0.3
Best-Effort (PDR ≥ 0.7)	7.21 ± 0.28	321 ± 23	88.6 ± 2.1	1.2 ± 0.4
Robustness				
Node Failure (10% at t=2000s)	9.08 ± 0.42	324 ± 26	89.4 ± 2.1	2.4 ± 0.7
Mobile Sink (1 *m/s*)	8.67 ± 0.36	312 ± 23	90.6 ± 1.8	1.6 ± 0.5
Traffic Variation (Light→Heavy)	8.91 ± 0.34	298 ± 21	90.8 ± 1.5	1.4 ± 0.4

Note: Evaluation of the CMORLM's adaptability to strict Quality of Service (QoS) variations and dynamic network disruptions. Scenarios include varying latency and reliability constraints, as well as robustness tests involving sudden node failures (10% deactivation), a mobile sink, and abrupt traffic load shifts. Values are Mean ± Standard Deviation over 10 runs.

For variations in QoS requirements, EEL-critical applications can achieve an average EEL of 287 ± 18 ms, a 0.8 ± 0.3% violation rate, and an EC of 8.42 ± 0.31 *mJ/packet*. EEL-tolerant applications relaxing the constraint of EC to 7.68 ± 0.26 *mJ/packet*. EEL is 412 ± 28 *ms,* and PDR is 93.1 ± 1.2%. Reliability-sensitive scenarios that enforce and achieve 93.8 ± 1.1% PDR. EC is 9.12 ± 0.34 *mJ/packet*. Best effort scenarios with relaxed minimize EC to 7.21 ± 0.28 *mJ/packet* while PDR is 88.6 ± 2.1%.

For robustness evaluation under dynamic network conditions, node failure with 10% random deactivation at seconds produces 9.08 ± 0.42 *mJ/packet*, 324 ± 26 *ms* EEL, 89.4 ± 2.1% PDR, and 2.4 ± 0.7% CVR. Policy adapts routing paths around failed nodes within 156 ± 34 *seconds*. A mobile sink scenario with a 1 *m/s* velocity achieves 8.67 ± 0.36 *mJ/packet*, 312 ± 23 *ms* EEL, 90.6 ± 1.8% PDR, and 1.6 ± 0.5% CVR. Traffic variation from light to heavy load results in 8.91 ± 0.34 *mJ/packet*, 298 ± 21 *ms* EEL, 90.8 ± 1.5% PDR, and 1.4 ± 0.4% CVR.

For hyperparameter sensitivity analysis, multi-objective weights are adjusted. Energy latency reliability ranges span [7.89, 10.87] *mJ/packet*, [214, 342] *ms*, and [88.4%, 94.6%]. For CVR, tighter thresholds reduce violations, while EC or EEL increases them. Learning rate variations affect convergence speed by up to 30% (1624 to 2187 episodes). Final performance metrics vary by less than 4%.

Table 7: Hyperparameter Sensitivity Analysis

Parameter	Baseline	Variation Range	Impact of EC	Impact of EEL	Convergence Impact
Energy Weight (w_1)	0.33	0.165 - 0.495	7.89 - 9.21 *mJ/pkt*	298 - 342 *ms*	±5%
EEL Weight (w_2)	0.34	0.17 - 0.51	8.42 - 10.87 *mJ/pkt*	214 - 318 *ms*	±6%
Reliability Weight (w_3)	0.33	0.165 - 0.495	8.42 - 10.12 *mJ/pkt*	PDR: 88.4 - 94.6%	±4%
Energy Threshold (δ_1)	0.2E_init	0.1 - 0.3 E_init	±2%	±3%	±8%
EEL Threshold (δ_3)	500 *ms*	250 - 750 *ms*	±4%	234 - 298 *ms*	±7%
Primal LR (α_θ)	3×10^{-4}	1.5×10^{-4} - 4.5×10^{-4}	±3.2%	±2.8%	1624 - 2187 ep

Note: *Impact measured as deviation from baseline. Convergence in episodes to 95% final performance.

The robustness of the CMORLM is further evidenced by its resilience to environmental volatility. As shown in Table 6, when subjected to a 10% random node failure, the policy dynamically reroutes traffic, maintaining an 89.4% PDR with only a marginal increase in latency. Furthermore, the hyperparameter sensitivity analysis (Table 7) demonstrates that the model is not overly brittle; variations in learning rates and objective weights (±10%) result in less than a 5% fluctuation in overall energy consumption and latency. This stability is critical for practical decision-making, as it ensures that operators do not need to perfectly fine-tune hyperparameters to achieve near-optimal, safe network performance.

7 Conclusion and Future Work

This work presents CMORLM for joint MAC-DCO routing in low-power WSNs. The CMORLM integrates primal dual optimization with multi-objective policy learning. Hard constraints are enforced on RE, buffer capacity, and QoS requirements. The proposed CMORLM achieves 34.2% reductions

in EC and EEL, and a 16.5% improvement in PDR, compared to TLP stacks. NL extends by 38.4%, and CVR is below 1%. This is a 23× reduction compared to UMORL. In ablation studies, joint cross-layer optimization achieves 44.7% improvement in EC over single-layer control. Multi-objective formulation provides balanced performance across competing objectives. LCH ensures safe operation without catastrophic failures during learning. The CMORLM proves robust performance across network scales from 50 to 200 nodes and adapts to dynamic conditions, including node failures, mobile sinks, and traffic variations. Pareto frontier analysis shows that operator control over energy latency-reliability trade-offs is achieved through weight configuration.

Despite its advantages, this study has several limitations that present avenues for future research. First, regarding deployment, there remains a 'sim-to-real' gap; while the model accounts for TelosB hardware constraints, deploying deep neural networks on microcontrollers with only 10 kB of RAM requires further model quantization and pruning. Second, regarding scalability, the reliance on periodic global state synchronization and centralized gradient aggregation at the sink introduces communication overhead that scales linearly with the network size, potentially creating a bottleneck for networks exceeding 200 nodes. Finally, the current framework assumes relatively stationary traffic statistics during training episodes. In highly non-stationary environments—such as rapid, unpredictable mobility or extreme, sudden interference—the RL agent may experience performance degradation before the policy has sufficient time to adapt.

Future work includes Federated Learning (FL) methods to eliminate centralized parameter aggregation and integration with energy-harvesting mechanisms for perpetual network operation—an extension to mobile sensor networks with dynamic topology. Transfer Learning (TL) across different deployment environments to reduce training time. Formal verification of safety constraints using Lyapunov stability analysis. Investigation of model-based RL could improve sample efficiency. Hierarchical multi-agent formulations may address scalability beyond 200 nodes through decentralized coordination.

Managerial Implications: For industry practitioners and network managers, the proposed framework offers a systematic tool to navigate the inherent trade-offs in IoT deployments. Instead of relying on trial-and-error heuristics, operators can use the CMORLM to guarantee strict Quality of Service (QoS) and safety constraints (e.g., preventing premature battery depletion). This translates to reduced maintenance costs, fewer network outages, and more predictable performance in resource-constrained environments.

Funding

This work was supported by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia [Grant No. **KFU261265**]

References

Aburukba, R., & El Fakih, K. (2025). Wireless sensor networks for urban development: A study of applications, challenges, and performance metrics. *Smart Cities*, 8(3), 89.

Behera, T. M., Samal, U. C., Mohapatra, S. K., Khan, M. S., Appasani, B., Bizon, N., & Thounthong, P. (2022). Energy-efficient routing protocols for wireless sensor networks: Architectures, strategies, and performance. *Electronics*, 11(15), 2282.

Ben Yaala, S., Ben Yaala, S., & Bouallegue, R. (2025). Optimizing TSCH scheduling for IIoT networks using reinforcement learning. *Technologies*, 13(9), 400. https://doi.org/10.3390/technologies13090400

Bhutani, M., Oruganti, S. K., Gupta, S. K., Alsekait, D. M., AbdElminaam, D. S., & Albeshri, M. Y. (2025). Redesigning MAC superframes for adaptive priority handling in IEEE 802.15.7-based optical wireless sensor networks. *IEEE Access*, 13, 189607–189628.

Dey, K., & Ghosh, S. (2024). iTRPL: Multi-agent reinforcement learning-based objective function for RPL in IoT. *Ad Hoc Networks*, 163, 103586. https://doi.org/10.1016/j.adhoc.2024.103586

Ekpenyong, M. E., Asuquo, D. E., Udo, I. J., Robinson, S. A., & Ijebu, F. F. (2022). IPv6 routing protocol enhancements over low-power and lossy networks for IoT applications: A systematic review. *New Review of Information Networking*, 27(1), 30–68.

El-Hajj, M. (2025). Enhancing communication networks in the new era with artificial intelligence: Techniques, applications, and future directions. *Network*, 5(1), 1. https://doi.org/10.3390/network5010001

Feng, C., Zhang, A., Min, G., Huang, Y., Quek, T. Q. S., & You, X. (2025). Towards 6G native-AI edge networks: A semantic-aware and agentic intelligence paradigm. *arXiv Preprint*. https://arxiv.org/abs/2512.04405

Halloum, N., Ahmadi, A., & Darmani, Y. (2026). Aris-RPL: A multi-objective reinforcement learning framework for adaptive and load-balanced routing in IoT networks. *Future Internet*, 18(2), 72. https://doi.org/10.3390/fi18020072

Islam, T., & Lee, Y. K. (2019). A comprehensive survey of recent routing protocols for underwater acoustic sensor networks. *Sensors*, 19(19), 4256. https://doi.org/10.3390/s19194256

Khan, O., Ullah, S., Khan, M., & Chao, H.-C. (2025). RL-BMAC: An RL-based MAC protocol for performance optimization in wireless sensor networks. *Information*, 16(5), 369. https://doi.org/10.3390/info16050369

Khan, S., Mazhar, T., Shahzad, T., Ghadi, Y. Y., & Hamam, H. (2025). Integrating IoT and WSN: Enhancing quality of service through energy efficiency, scalability, and secure communication in smart systems. *Peer-to-Peer Networking and Applications*, 18(5), 249.

Kumar, S., Chinthaginjala, R., Ahmad, S., & Kim, T. (2025). Energy-efficient unequal multi-level clustering for underwater wireless sensor networks. *Alexandria Engineering Journal*, 111, 33–46.

Latif, S. A., Drieberg, M., Sarang, S., Abd Aziz, A., Ahmad, R., & Stojanovic, G. M. (2025). A reinforcement learning-based intelligent duty cycle MAC protocol for Internet of Things. *IEEE Access*, 13, 156170–156187. https://doi.org/10.1109/ACCESS.2025.3606053

Lei, J., & Liu, D. (2024). RARL: Reinforcement learning aided routing for load balancing in WSN. *Pervasive and Mobile Computing*, 99, 101891. https://doi.org/10.1016/j.pmcj.2024.101891

Luong, N. C., Hoang, D. T., Gong, S., Niyato, D., Wang, P., Liang, Y.-C., & Kim, D. I. (2019). Applications of deep reinforcement learning in communications and networking: A survey. *IEEE Communications Surveys & Tutorials*, 21(4), 3133–3174.

Mustafa, R., Sarkar, N. I., Mohaghegh, M., Pervez, S., & Vohra, O. (2025). Cross-layer analysis of machine learning models for secure and energy-efficient IoT networks. *Sensors*, 25(12), 3720. https://doi.org/10.3390/s25123720

Panda, N., Supriya, M., & Elsts, A. (2025). Prioritized and multi-agent reinforcement learning-based TSCH schedulers. *IEEE Open Journal of the Computer Society*, 6, 1763–1774. https://doi.org/10.1109/OJCS.2025.3624137

Priyadarshi, R. (2024). Exploring machine learning solutions for overcoming challenges in IoT-based wireless sensor network routing: A comprehensive review. *Wireless Networks*, 30(4), 2647–2673.

Rottleuthner, M., Schmidt, T. C., & Wählisch, M. (2025). Duty-cycling is not enough in constrained IoT networking: Revealing the energy savings of dynamic clock scaling. *arXiv Preprint*. https://arxiv.org/abs/2508.09620

Salim, A. (2023). An approach for data routing in wireless body area network. *Wireless Personal Communications*, 130(1), 377–399.

Santos, C. L. D., Mezher, A. M., León, J. P. A., Cárdenas-Barrera, J. L., Guerra, E. C., & Meng, J. (2024). Q-RPL: Q-learning-based routing protocol for advanced metering infrastructure in smart grids. *Sensors*, 24(15), 4818. https://doi.org/10.3390/s24154818

Schlichter, J., Schwarz, M., & Wolf, L. (2025). Evaluating the effects of different layer multi-connectivity on reliable multi-hop industrial WSNs. *IEEE Internet of Things Journal*. https://doi.org/10.1109/JIOT.2025.3549254

Terven, J. (2025). Deep reinforcement learning: A chronological overview and methods. AI, 6(3), 46. https://doi.org/10.3390/ai6030046

Trigka, M., & Dritsas, E. (2025). Wireless sensor networks: From fundamentals and applications to innovations and future trends. *IEEE Access*, 13, 98504–98529. https://doi.org/10.1109/ACCESS.2025.3574660

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

Zerguine, N., Aliouat, Z., & Kharchi, S. (2025). A Reinforcement Learning-Based Scheduling Scheme for the IEEE 802.15.4e TSCH Network. *Engineering, Technology & Applied Science Research*, *15*(5), 27060–27068. https://doi.org/10.48084/etasr.12033