Journal of Theoretical and Applied Information Technology
15th February 2026. Vol.104. No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ENERGY-EFFICIENT AND QOS-OPTIMIZED IOT ROUTING USING PROPOSED QEER-MFO HYBRID ALGORITHM

[1]S. BHARATHI, [2]DR. D. MARUTHANAYAGAM

[1]Research Scholar, Sri Vijay Vidyalaya College of Arts & Science, Dharmapuri, Tamilnadu, India
[2] Dean Cum Professor, PG and Research Department of Computer Science, Sri Vijay Vidyalaya College of Arts & Science, Dharmapuri, Tamilnadu, India.

E-Mail Id: [1]bharathimscmphil2018@gmail.com,[2]kowsimaruthu@gmail.com

ABSTRACT

The Internet of Things' (IoT) rapid expansion has transformed connectivity in a number of industries, including smart cities, healthcare, transportation, and agriculture. However, because of their dynamic topology, device heterogeneity, and limited power resources, IoT networks continue to confront difficulties with energy management, routing efficiency, and Quality of Service (QoS). Conventional routing protocols sometimes don't adjust to the changing needs of IoT environments because they were mostly created for static or homogeneous networks. In order to overcome these constraints, this study suggests a brand-new hybrid routing architecture called QoS and Energy-Efficient Routing utilizing Moth-LSTM Optimization (QEER-MFO), which combines Long Short-Term Memory (LSTM) networks with the Moth-Flame Optimization (MFO) method. While the LSTM model forecasts network characteristics like traffic congestion and node energy depletion to enable proactive routing decisions, the MFO component makes sure that the best path is chosen through adaptive exploration and exploitation, reducing energy consumption and transmission latency. The Trust-Aware IIoT Routing Dataset is used to assess the suggested method, and the ns3-gym interface is used to simulate it in NS-3 combined with Python. According to experimental data, QEER-MFO outperforms current bio-inspired and conventional routing protocols, resulting in increased packet delivery ratios, decreased latency, and extended network lifetime. For scalable and intelligent IoT ecosystems, this hybrid architecture provides a self-learning, predictive, and energy-efficient routing approach.

Keywords: *Internet of Things (IoT), Quality of Service (QoS), Energy Efficiency, Bio-Inspired Optimization, Moth-Flame Optimization (MFO), Long Short-Term Memory (LSTM), Hybrid Routing, Machine Learning, NS-3 Simulation and Predictive Network Modeling*.

1. INTRODUCTION

The Internet of Things (IoT) refers to a widespread network of interconnected physical objects sensors, actuators, smart appliances, wearables, and embedded systems that collect and exchange data over communication networks [1]. Because IoT deployments enable real-time monitoring, analytics, and autonomous decision making, they are revolutionizing sectors including healthcare, smart cities, agriculture, transportation, and industrial automation as they grow to billions of devices [2]. This transition has been sped up by the development of inexpensive sensors, edge computing, and ubiquitous connectivity, which has created both new opportunities and difficulties for network design, especially at the routing layer. IoT networks are dynamic and resource-constrained, making routing extremely difficult. Protocols created for static networks are rendered ineffective by frequent topology changes brought on by node mobility, malfunctions, or new additions [3]. Routing techniques that minimize energy consumption are necessary to increase network lifetime since many IoT devices are battery-powered and have limited energy, memory, and computing capabilities [4]. Routing design is made more difficult by the scalability and heterogeneity of IoT systems, which frequently comprise thousands of devices with different hardware and communication standards [5]. Route stability is very challenging to maintain in low-power and lossy networks

(LLNs) since wireless links are frequently unstable due to interference, high error rates, and asymmetry. The necessity for strong, flexible, and energy-efficient routing protocols that can maintain performance in changing IoT environments is highlighted by these coupled difficulties [6]. These elements make creating reliable, flexible routing protocols for the Internet of Things a significant research issue.

Latency, throughput, packet delivery ratio (PDR), and reliability are examples of Quality of Service (QoS) measures that are essential for many Internet of Things applications (e.g., medical monitoring, industrial control). However, excessive connectivity can rapidly deplete batteries in IoT devices, which are usually energy-constrained [7]. Enhancing QoS frequently necessitates more transmissions, redundancy, or retransmissions, which runs counter to energy-saving objectives. As a result, a routing protocol must optimize both energy usage and QoS performance without severely impairing the other. In dynamic IoT environments, many current approaches consider them independently or make fixed trade-off assumptions, which is insufficient.
When used in contemporary IoT networks, traditional routing techniques like the shortest path, distance vector, link-state, and conventional ad hoc protocols have significant drawbacks. Because these methods were created for comparatively stable environments, they are not adaptive to frequent changes in topology or node mobility [8]. Real-time changes in traffic, congestion, or residual energy are not taken into consideration by their static cost functions, which are frequently based solely on hop count or link cost. Furthermore, proactive routing methods are inefficient for low-power Internet of Things devices because they produce extra control traffic and energy overhead. Conventional approaches often have trouble juggling several goals; they usually optimize one parameter, such path length, while ignoring others, like QoS or energy efficiency [9]. In the absence of heuristic adaptability or predictive intelligence, they are vulnerable to local optima, which eventually leads to decreased performance. The necessity for intelligent, adaptive, and hybrid routing frameworks like QEER-MFO is thus highlighted. Although fundamental, traditional legacy techniques are insufficient for dynamic, heterogeneous IoT contexts.

Because they are decentralized and adaptive, bio-inspired algorithms (BIAs), which imitate natural intelligence, are ideal for dynamic Internet of Things environments. ACO may converge slowly in vast networks, but it mimics ant foraging to discover the best routes [10]. FFA risks early convergence by using firefly attraction for load balancing and global search [11]. Grey wolf hunting behavior is modeled by GWO, which lacks adaptive learning but achieves good energy-efficient clustering [12]. Motivated by moth navigation, MFO [13] provides quick convergence and efficient global optimization, making it perfect for energy management and IoT routing. These heuristic approaches, however, are reactive and unable to forecast the future state of the network. Their adaptability is increased by integrating them with machine learning (ML) [14]. **LSTM networks, in particular, can forecast energy, delay, and congestion, enabling predictive and stable routing decisions**, while Deep Reinforcement Learning (DRL) adds self-optimization for real-time IoT adaptability [15].

While bio-inspired algorithms offer exploration and robustness, they lack foresight into future network conditions. Conversely, **ML models provide predictive capability, yet require optimization mechanisms to explore vast solution spaces efficiently**. Because of their complementing qualities, Moth-Flame Optimization (MFO) and LSTM prediction are integrated to create a hybrid system that combines the advantages of both paradigms. Self-learning IoT routing that anticipates connection deterioration, adjusts to topology changes, and proactively optimizes energy use can be made possible by the combination of heuristic search and data-driven learning [16]. In addition to improving Quality of Service (QoS) metrics like throughput, latency, and packet delivery ratio, this hybridization increases network lifetime and guarantees long-term IoT operations.

The primary objective of this research is to develop a hybrid, predictive, and energy-efficient IoT routing framework called QEER-MFO (QoS and Energy-Efficient Routing using Moth-LSTM Optimization). The framework intends to develop an LSTM-based prediction model to predict network conditions like congestion and node energy depletion (reduction), as well as an MFO-based optimization model for identifying

energy-conscious routing paths. These modules work together to optimize QoS and energy efficiency through proactive and adaptive route selection. The model is evaluated using metrics such as delay, PDR, throughput, and energy consumption, and its performance is compared with FFA, GWO and MFO, to demonstrate its superiority.

Despite extensive research on IoT routing, existing protocols and optimization techniques still suffer from critical limitations. Traditional routing protocols such as LEACH, RPL, and AODV are largely reactive and rely on static or instantaneous network conditions, making them unsuitable for highly dynamic and energy-constrained IoT environments. Similarly, standalone bio-inspired algorithms like FFA, GWO, and MFO provide effective global optimization but lack predictive intelligence, leading to suboptimal routing decisions under fluctuating traffic and energy conditions. On the other hand, machine learning–based routing methods offer predictive capabilities but often lack robust global search mechanisms and struggle with scalability. This research addresses this gap by proposing a novel hybrid routing framework, QEER-MFO, which tightly integrates LSTM-based temporal prediction with Moth-Flame Optimization. The proposed approach introduces predictive fitness adjustment, enabling proactive, energy-aware, and QoS-optimized routing. By anticipating congestion, link instability, and energy depletion, QEER-MFO significantly enhances network lifetime, reliability, and latency performance, making it highly suitable for large-scale and mission-critical IoT applications such as smart healthcare and industrial IoT.

This paper is organized as follows: The problem formulation and key limitations are presented in Section 3, the proposed QEER-MFO methodology and architecture is presented in Section 4, the experimental setup, datasets, metrics, and results are explained in Section 5, and the study is concluded with insights and future research directions in Section 6. Section 2 reviews related work on bio-inspired, ML-based, and hybrid IoT routing methods.

2. RELATED WORKS

Originating from stigmergic foraging, Ant Colony Optimization (ACO) readily translates to multi-path discovery and traffic splitting in routing by creating pathways through pheromone deposition and evaporation [17]. ACO's probabilistic exploration helps avoid transitory link failures in IoT low-power/loss-prone environments, but convergence may lag in high dynamics and sparse pheromone signals. The straightforward real-valued search of the Firefly Algorithm (FFA), which simulates attraction proportionate to perceived brightness, is appealing for lightweight route-cost optimization (delay/energy) on limited nodes. However, if parameter sensitivity (randomness, absorption coefficient) is not adjusted for traffic volatility, it may result in oscillations or premature convergence [18]. GWO stands for Grey Wolf Optimizer. Clustering and energy-aware next-hop selection in WSN/IoT research have benefited greatly from GWO's leader–pack hierarchy (α, β, δ, ω), which balances exploration/exploitation with few control parameters; yet, fixed update algorithms may lag when topology changes quickly [19]. Like other heuristics, Moth-Flame Optimization (MFO) lacks the ability to predict future traffic/energy trends. However, it has demonstrated strong performance on multimodal cost surfaces, such as joint delay energy routing objectives, thanks to its logarithmic spiral around "flames," which allows for quick global search and path refinement [20]. The appropriateness of swarm-based metaheuristics (ACO/FFA/GWO/MFO) for energy-efficient clustering and routing under LLN limitations is also supported by recent IoT-focused surveys.

In order to enable proactive congestion avoidance and energy-aware path selection in IoT/LLNs, temporal predictors like LSTM learn both short- and long-term dependencies in traffic and link-quality series [21]. In addition to traffic, PRR/LQI can be predicted via LSTM-style WLQE to guide next-hop decisions [22]. CNN and CNN-RNN hybrids help with QoS compliance by classifying congestion patterns and predicting hot spots by extracting spatial/temporal features from multivariate telemetry (RSSI/ETX/queue length/topology snapshots) [23]. Routing policies that jointly maximize latency, PDR, and energy under dynamics are learned by Deep Reinforcement Learning (DRL); experiments show improvements in load balancing and gains over RPL/AODV baselines [24].

Once trained, learning models provide millisecond-level inference, strong scalability, and prediction (LSTM/CNN) and closed-loop adaptation (DRL) with online policy improvement [25]. Sample inefficiency and exploration costs in DRL, computational/energy overhead on motes (requiring edge/offload), stability/generalization issues under sudden topology changes, and data scarcity/labeling burdens (for supervised CNN/LSTM) are some of the obstacles they encounter [26]. Although these limitations have been lessened by recent efforts using distributed DRL, lightweight LSTM variants, and attention, careful feature designing, transfer learning, and resource-aware deployment are still necessary for reliable performance [27].

Hybrid approaches combine metaheuristics (for global exploration over multi-objective cost surfaces) with ML/DL (for prediction or classification), e.g., CNN/LSTM models guided by swarm-based search for parameter tuning or path selection. Hybrids have been reported for IoT/WSN tasks such as energy-efficient clustering/routing and traffic/security analytics: CNN- or LSTM-based predictors fused with optimization to steer decisions [28]. In networking contexts, DRL-augmented routing and multi-optimality schemes illustrate the promise of mixing learning with search for better convergence and multi-metric QoS. Studies on link-quality/traffic prediction further show that learned forecasts can be embedded inside heuristic selection to reduce control overhead and extend lifetime [29].

This review follows a simulation-driven research methodology to design, implement, and evaluate a predictive and energy-efficient IoT routing framework. The proposed QEER-MFO model is developed by integrating bio-inspired optimization and deep learning prediction within a clustered IoT architecture. The methodology consists of four major phases: (i) network design and clustering, (ii) data collection and preprocessing, (iii) hybrid optimization and prediction-based routing execution, and (iv) performance evaluation using standard QoS and energy metrics. NS-3 integrated with Python via the ns3-gym interface is used to simulate realistic IoT communication behavior, while LSTM learning and MFO optimization guide adaptive routing decisions.

3. PROBLEM STATEMENTS

3.1. IoT Network Model and Architecture (Cluster-Based Design)

The proposed IoT network is **cluster-based and organized into multiple hierarchical layers to ensure scalability, energy efficiency, and intelligent routing**.

The overall topology is illustrated Figure 1, conceptually as follows:

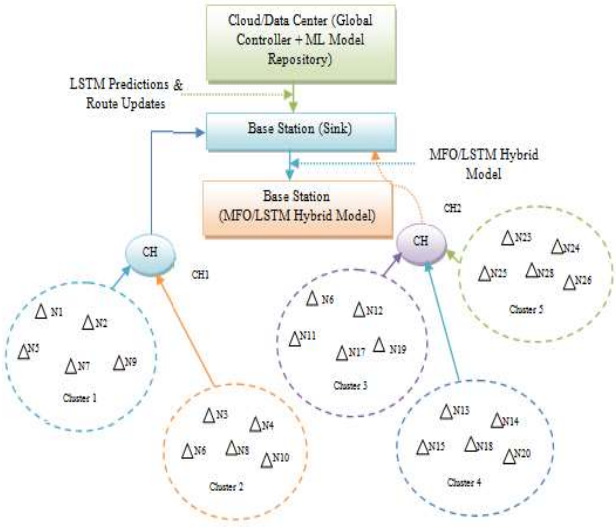

Figure 1: IoT Network Model

A. Sensor Layer (End Nodes)

The **sensor layer** consists of all IoT devices or sensor nodes (N_1–N_{20}) randomly deployed over a defined area (e.g., 500 m × 500 m). **Each node periodically collects environmental or contextual data such as temperature, pressure, or motion**. These nodes are **energy-constrained** and operate on batteries, transmitting their sensed data periodically to the assigned **Cluster Head (CH)**. To conserve energy, nodes switch to **idle or sleep mode** when not transmitting. If node *Ni* has residual energy$E_i(t)$, its transmission power at time *t* is determined by:

$$E_i(t+1) = E_i(t) - (E_{tx} + E_{amp} \times d_{i_{CH}}^2$$

Where - $d_{i_{CH}}^2$ is the distance between node N_i and its cluster head.

B. Cluster Layer (CHs)

The network is divided into five clusters, each managed by a Cluster Head (CH) responsible for

data aggregation, local routing, and energy monitoring. **Cluster Heads communicate with the base station using either direct or multi-hop links based on energy levels** and signal strength. Example cluster formation:

Cluster ID	Member Nodes	Cluster Head
C1	N1, N2, N5, N7	CH1
C2	N3, N4, N8, N9	CH2
C3	N6,N11, N12, N17, N19	CH3
C4	N13, N14, N15, N18,N20	CH4
C5	N23, N24, N25, N26,N28	CH5

Cluster Formation Algorithm, the **Moth-Flame Optimization (MFO)** algorithm determines the optimal CHs by minimizing:

$$F = a \times E_{residual} + \beta \times (1 - d_{CH} + \gamma \times Q_{link}$$

Where:

- $E_{residual}$: residual energy of node
- d_{CH-BS}: distance to base station
- Q_{link}: link quality or reliability
- α, β, γ: weighting coefficients

The node with the highest fitness score is selected as **Cluster Head (CH)**.

C. Communication Layer

The communication layer defines both intra-cluster and inter-cluster communication. In intra-cluster communication, sensor nodes send data to their respective CHs, while in inter-cluster communication, CHs forward aggregated data to the base station through optimized multi-hop routes. **These communication paths are determined by MFO and dynamically adjusted using LSTM-based predictions for link reliability and congestion**. Communication Phases include:

- **Setup Phase:** Cluster formation and CH selection using MFO.
- **Steady-State Phase:** Data transmission from nodes to CH and from CH to base station.
- **Maintenance Phase:** CH rotation and route updates based on LSTM predictions.

D. Base Station (Sink Node)

The base station (sink node) acts as the global network coordinator and hosts the QEER-MFO framework. **It integrates MFO for optimizing routing paths and cluster formation, and LSTM for predicting future network states such as node energy, delay, and congestion**. The base station collects aggregated data from CHs, updates the routing tables, and disseminates (distribute) adaptive control signals throughout the network.

E. Data Flow Explanation

The data flow in the IoT model proceeds through sequential and adaptive steps:

- **Data Sensing:** Each sensor node N_i senses environmental data and transmits it to its CH periodically.
- **Intra-Cluster Communication:** Nodes in the same cluster send data to their CH using a **single-hop link**. The CH aggregates redundant data to reduce transmission cost.
- **Cluster Head Optimization (MFO):** Using the MFO algorithm, CHs optimize routing paths toward the base station by evaluating fitness functions based on energy, distance, and QoS metrics.
- **Prediction (LSTM):** The LSTM model predicts next-round node energy, link quality, and delay using temporal features:

$$\hat{y}_{t+1} = f_{LSTM}(E_t, D_t, Q_t)$$

These predictions help proactively reroute data before congestion or failures occur.

- **Inter-Cluster Transmission:** Optimized CHs transmit aggregated data to the base station through **multi-hop paths** determined by MFO-LSTM cooperation.
- **Adaptive Learning Loop:** The system continuously monitors performance metrics and updates MFO parameters (flame count, fitness function weights) and LSTM weights to adapt to topology dynamics.

3.2. Problem Definition and Research Gap

Billions of diverse devices are connected by the Internet of Things (IoT), which produces real-time data for intelligent control and monitoring. Routing optimization is still a significant problem, though, as it has an impact on network dependability, performance, and energy efficiency. IoT nodes are subject to stringent energy and processing constraints, and their dynamic topologies result in erratic connectivity

and deteriorated QoS metrics including packet delivery ratio, throughput, and latency. Due to their high energy consumption, frequent topological changes, lack of predictive intelligence, and difficulty combining QoS with energy efficiency, traditional routing protocols like LEACH, RPL, and AODV—which were created for static or homogeneous networks—struggle in the Internet of Things. The suggested QEER-MFO architecture combines LSTM-based prediction for proactive, adaptive routing with Moth-Flame Optimization (MFO) for effective route discovery in order to overcome these problems. In dynamic, expansive IoT contexts, this hybrid method improves network lifetime, QoS dependability, and routing stability.

4. PROPOSED METHODOLOGY: QEER-MFO FRAMEWORK

The suggested QoS and Energy-Efficient Routing using Moth-LSTM Optimization (QEER-MFO) framework combines machine learning prediction with bio-inspired optimization to produce an intelligent, flexible routing system for Internet of Things networks. The model makes use of the Long Short-Term Memory (LSTM) network's capacity for temporal learning as well as the exploration–exploitation balance of the Moth-Flame Optimization (MFO) algorithm. When combined, these elements allow for QoS-optimized, energy-conscious, and predictive routing across clustered IoT settings.

The proposed QEER-MFO framework is designed as a cluster-based IoT routing architecture to ensure scalability, energy efficiency, and adaptability. Sensor nodes are organized into clusters, each managed by a dynamically selected Cluster Head (CH) responsible for data aggregation and inter-cluster communication. The framework consists of two tightly coupled components: an MFO-based optimization module that performs global route and CH selection, and an LSTM-based prediction module that learns temporal patterns in network behavior. These components operate in a closed feedback loop, allowing routing decisions to be continuously refined based on both current and predicted network states.

Two synergistic layers underlie the QEER-MFO framework's operation:

- **Optimization Layer (MFO):** Handles global route discovery, cluster head (CH) selection, and multi-hop path optimization.
- **Prediction Layer (LSTM):** Learns from historical IoT data (node energy, delay, traffic load) to predict future network states and guide adaptive route updates.

A closed adaptive optimization loop is produced by combining MFO with LSTM. The LSTM model forecasts future congestion and possible energy depletion, while the MFO algorithm produces the best routing options based on the state of the network at each iteration. After being fed these predictions, MFO is able to adapt its search strategy dynamically and more effectively converge toward energy-conscious and sustainable routing alternatives. The QEER-MFO architecture's simplified conceptual workflow is shown below. Figure 2:

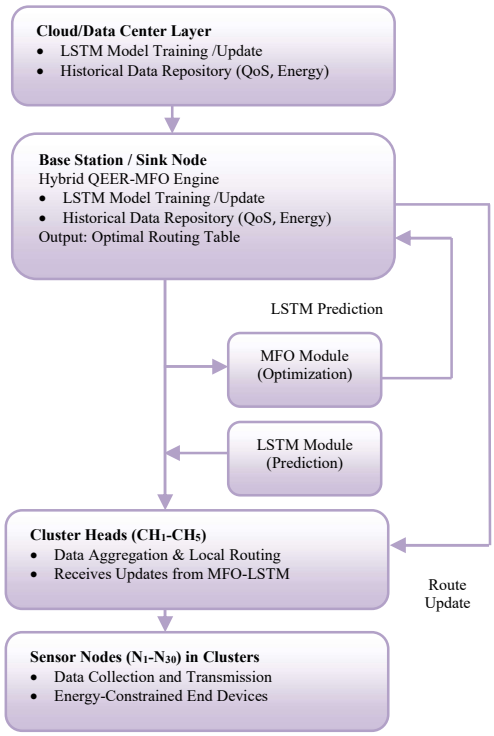

*Figure 2: Workflow of QEER-MFO **Architecture***

4.1. Workflow of QEER-MFO

The overall working of QEER-MFO follows **five systematic stages**, illustrated below.

Stage 1: Data Acquisition and Initialization

- The IoT network is deployed with nodes N1, N2,…,Nn.

- Each node periodically reports parameters: residual energy Ei, delay Di , throughput Ti, and packet success ratio Pi.
- Clusters are formed using a **distance–energy-based criterion**, and MFO initializes **candidate CH sets** randomly.
- The LSTM model loads pretrained weights from the cloud (based on previous routing datasets).

Stage 2: Fitness Evaluation (MFO Optimization Phase)

Each candidate solution (moth) represents a **potential routing configuration**—a set of cluster heads and their communication paths. For each candidate, the **fitness function** evaluates both QoS and energy factors:

$$F = \omega_1 \times PDR + \omega_2 \times (1 - Delay) + \omega_3 \times (1 - Energy\ Consumption)$$

Where, w_1, w_2, w_3 are weighted coefficients adjusted by network conditions. The top solutions (flames) are retained for the next iteration.

Stage 3: Prediction Phase (LSTM Learning Module)

The **LSTM model** receives time-series inputs:

$$X_t = [E_t, D_t, T_t, P_t]$$

and predicts the next time step:

$$\hat{Y}_{t+1} = [\hat{E}_{t+1}, \hat{D}_{t+1}, \hat{T}_{t+1}, \hat{P}_{t+1}]$$

These predictions identify **nodes at risk** of congestion or energy depletion. LSTM outputs influence MFO's flame position updates by penalizing routes that traverse predicted weak links.

Stage 4: Hybrid Update and Adaptive Routing

MFO updates moth positions around flames using LSTM-guided adjustments:

$$M_i^{t+1} = S\left(M_i^t, F_j\right) + \delta(LSTM_{predict})$$

where S is the logarithmic spiral function and δ represents predictive correction from LSTM. This ensures that routes converge not only on current optimal paths but also on **future-stable routes**.

Stage 5: Route Deployment and Continuous Learning

- The optimized route table is deployed at the base station.
- Cluster Heads (CHs) and nodes route data accordingly.
- The system monitors feedback (energy, delay, PDR), which is fed back to both MFO and LSTM.
- The cloud periodically retrains the LSTM model with new datasets, improving prediction accuracy for subsequent iterations.

The QEER-MFO (QoS and Energy-Efficient Routing using Moth-LSTM Optimization) framework optimizes IoT routing by combining bio-inspired exploration with machine learning-based prediction. The algorithm begins by initializing a population of moths representing candidate routes and evaluating their fitness based on QoS and energy metrics. The best-performing solutions (flames) are identified, and the LSTM model predicts future node states including energy ($\hat{E}, \hat{D}$), and packet delivery ratio ($\widehat{P}$).These predictions guide the MFO optimizer to penalize weak or unstable links and adjust moth positions using the spiral update equation:

$$M_i^{t+1} = D_{ie^{bl}} \cos(2\pi l) + F_j$$

This process iterates until convergence or a maximum iteration count is reached, after which the optimal routing path is selected and deployed to the IoT base station. **The framework continuously monitors performance, updates the dataset, and periodically retrains the LSTM to adapt to changing network conditions.**

Predictive routing, where LSTM forecasts avoid congestion and energy loss; global optimization through MFO exploration; and dynamic adaptability through ongoing LSTM–MFO feedback are among QEER-MFO's main advantages. Its scalable modular design allows for new clusters or retraining without requiring system resets, and its multi-objective fitness guarantees balanced QoS and energy efficiency. In comparison to traditional algorithms, QEER-MFO achieves better performance and a longer network lifetime by integrating LSTM predictions into the MFO loop, converting routing from a reactive process into a self-learning, proactive, and sustainable optimization system.

4.1.1 Execution Protocol

The execution of the QEER-MFO framework follows a structured protocol. Initially, IoT nodes are deployed and clustered, and the MFO algorithm initializes candidate routing solutions. At each transmission round, current network metrics are collected and evaluated. The LSTM module predicts future energy levels, delay, and congestion based on historical data sequences. These predictions are incorporated into the MFO

fitness function to penalize unstable or energy-draining routes. The optimized routing paths are then deployed, and the system continuously monitors performance feedback. This iterative execution ensures proactive adaptation to dynamic network conditions while maintaining QoS and energy efficiency.

4.2 Moth-Flame Optimization (MFO) Module

The central optimization engine of the suggested QEER-MFO system is the Moth-Flame Optimization (MFO) module. This metaheuristic algorithm effectively explores and exploits the solution space to find energy-efficient and QoS-optimized routing solutions in dynamic IoT networks, drawing inspiration from the natural navigation behavior of moths. Cluster head (CH) selection, multi-hop route optimization, and fitness evaluation based on the network's Quality of Service (QoS) and energy consumption metrics are all handled by the MFO algorithm in the QEER-MFO model.

A. Biological Inspiration and Principle

Moths use a technique known as transverse orientation to navigate at night by keeping their angle steady with respect to the moonlight. Nevertheless, moths spiral around artificial lights indefinitely, which gave rise to the idea of a logarithmic spiral movement. The MFO method is mathematically based on this natural phenomenon [14]. Each moth, which is made up of certain cluster heads (CHs) and their communication linkages, represents a potential routing solution throughout the optimization process. Each flame, which represents the top-performing routes with the highest fitness values, relates to an elite or ideal solution that has been found thus far. In order to attain optimal network performance, the moths efficiently balance exploration (looking for new routing paths) and exploitation (improving the best current solutions) by updating their positions by spiraling around the flames during each iteration.

B. Model of MFO

Representation of Solutions:

Let $M=\{M_1,M_2,...,M_n\}$ denote a population of n moths, where each moth Mi encodes a candidate routing path in the IoT network:

$$M_i = [CH_1, CH_2, \dots., CH_k]$$

Where, CH_krepresents the chosen cluster heads for routing at iteration *t*.

Flame Matrix

Similarly, the flame population $F=\{F_1,F_2,...,F_n\}$ stores the best solutions found in each iteration. The number of flames decreases linearly with iterations to enhance exploitation as the algorithm converges:

$$Flame_{count} = round\ (N - \frac{t}{T}(N-1))$$

Where , N = total number of moths, t = current iteration, T = maximum iterations.

Spiral Movement (Position Update)

The **logarithmic spiral** guides each moth towards its corresponding flame:

$$M_i^{t+1} = D_{i.}\ e^{b.l}\cos(2\pi l) + F_j$$

Where , $M_i^{(t+1)}$: new position of the *i-th* moth, $D_i=|F_j-M_i(t)|$: distance between moth and flame, b: constant defining spiral tightness (commonly set to 1), l: a random number in [−1,1] controlling spiral shape, F_j : *j-th* flame (elite solution). The **spiral function** allows controlled exploration (large l values) and exploitation (small l values).

Fitness Function

The fitness function integrates **QoS and energy parameters** to evaluate each candidate route:

$$Fitness = \omega 1 \times PDR + \omega 2 \times (1 - Delay) + \omega 3 \times (1 - E_{cons})$$

Where, PDR: Packet Delivery Ratio, Delay: average end-to-end delay, Econs: normalized energy consumption, w1, w2, w3: weighting coefficients satisfying w1+w2+w3=1. This multi-objective fitness ensures that the selected routes balance reliability, latency, and energy efficiency.

Integration with LSTM Predictions:

Receive predictive feedback from the **LSTM module** on expected link congestion or node energy depletion. Adjust flame weights dynamically:

$$F_j' = F_j \times (1 - \delta \times R_{pred})$$

Where, R_{pred} is the predicted risk factor and δ is a learning constant.

The QEER-MFO framework's MFO Module acts as the optimization engine, utilizing a bio-inspired search method to evolve and improve routing paths. The initialization step is where node properties like energy, distance, PDR, and delay are specified and random candidate routes (moths) are created. The best-performing routes are ranked as flames throughout the fitness evaluation process, which uses QoS and energy metrics to evaluate each moth's quality. The logarithmic spiral equation is then used in the spiral update phase to modify moth placements around flames, progressively reducing the number of flames to improve exploitation. The MFO can dynamically improve its search by using the LSTM module's predictive feedback on

future connection congestion and node energy depletion. The best communication path is determined by choosing the route with the highest fitness when this iterative process is completed and the fitness function converges.

The hybrid QEER-MFO framework ensures energy-stable cluster head selection and QoS-resilient routing under a range of traffic and mobility scenarios by using MFO as the global optimizer and LSTM as the predictive intelligence. The MFO module is effective for large-scale IoT simulations or edge-based deployments due to its high exploration capabilities, quick convergence, and low computing cost. It concurrently balances throughput, energy efficiency, and delay through multi-objective optimization. MFO transforms into a self-adaptive, predictive optimizer by integrating LSTM-driven predictions, guaranteeing reliable, energy-conscious, and high-performance routing across dynamic IoT networks.

4.3 LSTM-Based Prediction Module

The temporal learning part of the suggested QEER-MFO system is the Long Short-Term Memory (LSTM)-Based Prediction Module. Its main purpose is to estimate future network states by analyzing past IoT network metrics including packet delivery ratio (PDR), throughput, latency, and node energy. By giving the MFO optimizer predictive direction, these projections allow for proactive route modifications before to traffic jams, node failures, or energy depletion.

A. Role and Motivation

Traditional bio-inspired optimizers (ACO, GWO, FFA, and MFO) only use current network measurements to carry out reactive optimization. IoT networks do, however, show temporal dependencies, meaning that historical patterns—such as energy depletion or packet delays—have a significant impact on present and future conditions. By using feedback connections inside its memory cells to learn sequential dependencies, the LSTM network gets over this restriction and is hence well-suited for time-series prediction in dynamic Internet of Things contexts. For QEER-MFO, LSTM forecasts:

- **Residual node energy** in upcoming transmission rounds.
- **Expected end-to-end delay and congestion levels.**
- **Link quality and packet loss trends.**

These predictions guide MFO in penalizing unstable routes and prioritizing energy-stable and low-latency paths**.**

B. LSTM Architecture for IoT Routing Prediction

The proposed LSTM network follows a **sequential model architecture** composed of:

Layer	Description
Input Layer	Accepts normalized network parameters: residual energy (Et), delay (Dt), PDR (Pt), throughput (Tt), and control overhead (Ot).
LSTM Hidden Layers (2–3 stacked)	Capture long-term dependencies and nonlinear temporal patterns in routing metrics. Each layer contains 64–128 memory cells.
Dropout Layer	Prevents overfitting during training by randomly deactivating neurons (dropout = 0.2).
Fully Connected (Dense) Layer	Combines features from hidden layers and prepares outputs for regression.
Output Layer	Predicts future values $\hat{E}_{t+1}, \hat{D}_{t+1}, \hat{P}_{t+1}, \hat{T}_{t+1}$

At time step *t*, the LSTM cell receives input vector x_t and the previous hidden state h_{t-1}. It updates its states using:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad \text{(Forget gate)}$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad \text{(Input gate)}$$

$$\hat{C}_t = tanh(W_c[h_{t-1}, x_t] + b_C) \quad \text{(Candidate State)}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \hat{C}_t \quad \text{(Cell State update)}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad \text{(Output gate)}$$

$$h_t = o_t \odot \tanh(C_t) \quad \text{(Hidden Output)}$$

where σ is the sigmoid function, and ⊙ denotes element-wise multiplication. The final hidden state h_t at the last time step represents the learned temporal features, which are fed into a dense layer to produce multi-dimensional predictions.

C. Input and Output Specification

- **Input features:** $X_t = [E_t, D_t, P_t, T_{t,} O_t]$ (normalized using Min-Max scaling)
- **Output predictions:** $Y_{t+1} = [\hat{E}_{t+1}, \hat{D}_{t+1}, \hat{P}_{t+1}, \hat{T}_{t+1}]$
- **Dataset source:** Trust-Aware IIoT Routing Dataset (Kaggle) augmented with NS-3-generated QoS logs.
- **Training parameters:** Optimizer = Adam, learning rate = 0.001, batch size = 64, epochs = 100.
- **Loss function:** Mean Squared Error (MSE)

$$L = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

D. Prediction Workflow

- **Data Collection:** Each cluster head and node periodically send state vectors X_t to the base station.
- **Preprocessing:** Data is normalized and **divided into training and testing sequences**.
- **Model Training:** The cloud layer trains the LSTM using historical IoT data. Model weights are periodically transmitted to the base station.
- **Real-Time Prediction:** During operation, the trained LSTM predicts future QoS and energy metrics for the next time interval t+1.
- **Feedback to MFO:**
  - If $\hat{E}_{t+1}$for a node falls below threshold E_{th}, the MFO penalizes routes through that node.
  - If predicted $\hat{D}_{t+1}$ *or* $\hat{O}_{t+1}$ exceeds QoS limits, alternative low-delay paths are prioritized.
- **Continuous Learning:** The predicted values are validated with actual measurements; discrepancies are logged for future model retraining.

E. Integration of LSTM with MFO

The integration between LSTM and MFO forms the **core intelligence loop** of QEER-MFO:

- **MFO performs global search to identify high-fitness routing candidates**.
- LSTM predicts future network metrics for these candidates.
- MFO updates its fitness scores dynamically using prediction-based penalty or reward terms:

$$F' = F - \lambda_1(1 - \hat{E}_{avg}) - \lambda_2 \hat{D}_{pred} + \lambda_3 \hat{P}_{pred}$$

Where, λi are adaptive weighting factors.

- The adjusted fitness guides the next spiral update, leading to **predictive convergence** toward sustainable routes.

The QEER-MFO framework is converted from a static optimizer into a proactive, self-learning system via the LSTM-based prediction module. In order to reduce latency, balance energy consumption, and increase overall network lifetime, the LSTM network forecasts important QoS and energy parameters, enabling the MFO optimizer to make smart, proactive routing decisions. A resilient, adaptive, and predictive next-generation intelligent IoT routing paradigm is produced by combining deep learning with nature-inspired search.

4.4. Hybrid Integration Strategy

The suggested QEER-MFO (QoS and Energy-Efficient Routing utilizing Moth-LSTM Optimization) system is based on the Hybrid Integration Strategy. It outlines the dynamic interactions between the machine learning prediction (LSTM) and bio-inspired optimization (MFO) modules to produce a cohesive, intelligent routing system that adjusts to the conditions of an IoT network in real time. The hybrid system combines temporal foresight (by LSTM's predictive modeling) and exploratory routing intelligence (via MFO's search capacity), in contrast to conventional standalone optimizers or learning-based predictors. In large-scale IoT deployments, this synergy guarantees sustained QoS, energy balancing, and proactive route selection. The QEER-MFO hybrid workflow smoothly integrates the MFO and LSTM modules as it moves through six main stages:

Stage 1: Initialization

- Deploy IoT nodes and form clusters using random or distance energy heuristics.
- Initialize MFO population (moths), each representing a candidate routing configuration.
- Load pretrained LSTM weights for initial prediction.
- Define algorithmic parameters: population size, maximum iterations, and fitness weight coefficients w_1,w_2,w_3.

Stage 2: Data Acquisition

- Each node periodically reports metrics such as residual energy E_t, delay D_t, throughput T_t, and PDR Pt.
- These metrics are forwarded to the base station and passed to both modules:
  - MFO uses them for immediate optimization.
  - LSTM uses them for prediction of next time-step values $\hat{E}_{t+1}, \hat{D}_{t+1}, \hat{P}_{t+1}, \hat{T}_{t+1}$]

Stage 3: LSTM Prediction Phase

- LSTM analyzes the temporal sequence of metrics and predicts the **future state** of each node or link:

$$\hat{Y}_{t+1} = f_{LSTM}(E_t, D_t, P_t, T_t)$$

- Predictions are sent to the MFO module as **predictive control signals**.

Stage 4: MFO Optimization Phase

- The MFO algorithm evaluates each moth's fitness using **hybrid prediction-adjusted fitness**:

$$F' = \omega_1\left(PDR + \hat{P}_{t+1}\right) + \omega_2\left(1 - \left(D_t + \hat{D}_{t+1}\right)\right) + \omega_3(1 - \left(E_{cons} + \hat{E}_{decay}\right))$$

- Here, $\hat{E}_{decay}$ represents predicted energy loss per node.
- Moths move in the search space (routing configurations) around flames using the logarithmic spiral rule:

$$M_i^{t+1} = \left|F_j - M_i\right| . e^{b.l}.\cos\ (2\pi l) + F_j$$

- LSTM predictions adjust **flame ranking**, ensuring routes with high predicted delay or low energy are penalized.

Stage 5: Adaptive Feedback and Route Deployment

- The MFO–LSTM loop continues until convergence or a stopping condition (e.g., negligible fitness improvement).
- The **best hybrid solution** (routing path with maximum adjusted fitness) is deployed in the IoT network.
- The real-time routing table is updated at the base station and propagated to cluster heads.

Stage 6: Continuous Learning and Model Update

- After every routing cycle, real performance values (E_{real},D_{real},P_{real}) are compared with LSTM predictions.
- The differences (ΔE, ΔD, ΔP) are stored and used to **retrain the LSTM model** periodically in the cloud layer.
- This ensures the prediction model evolves as network behavior changes, maintaining high accuracy and adaptability.

Hybrid Fitness Adjustment

The hybrid system combines **MFO's real-time optimization** and **LSTM's predictive awareness** through a composite fitness function:

$$F_{hybri} = \alpha.F_{MFO} + \beta.F_{LSTM}$$

where:

$$F_{MFO} = \omega_1 PDR + \omega_2(1 - Delay) + \omega_3(1 - E_{cons})$$

$$F_{LSTM} = \omega_1\hat{P}_{t+1} + \omega_2\left(1 - \hat{D}_{t+1}\right) + \omega_3(1 - \hat{E}_{t+1})$$

- α,β are control coefficients that regulate the balance between present optimization and predicted stability.

The hybrid score F_{hybrid} thus promotes current high-performance routes that are also future-stable and energy-resilient.

For large-scale Internet of Things (IoT) networks, the suggested QEER-MFO framework offers a predictive, adaptive, and intelligent routing solution (Figure 3). In contrast to traditional static or heuristic routing protocols, QEER-MFO achieves high-performance, sustainable communication in resource-constrained contexts by combining machine learning-based prediction with bio-inspired optimization. The model combines the Long Short-Term Memory (LSTM) neural network for temporal prediction of network behavior with the Moth-Flame planning (MFO) algorithm for global route exploration and energy-aware path planning. When combined, these elements form a synergistic hybrid framework that cleverly strikes a compromise between network lifetime and energy efficiency and Quality of Service (QoS) measures including latency, throughput, and packet delivery ratio.

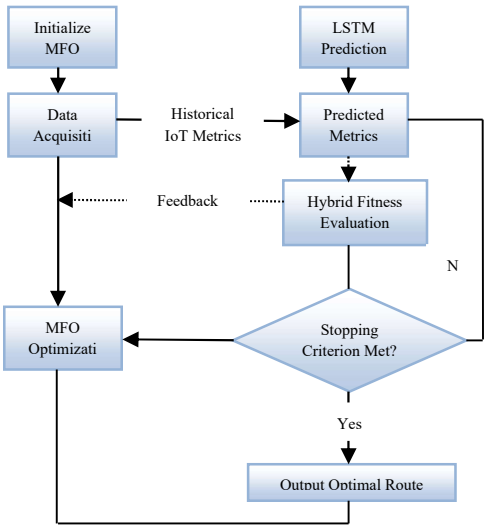

Figure 3: Hybrid Integration Workflow

The IoT environment is represented as a cluster-based network in QEER-MFO, with sensor nodes arranged into several clusters according to factors like communication cost, proximity, and

residual energy. A Cluster Head (CH) oversees each cluster and is in charge of data aggregation and communication with the sink node or base station. The best CHs and routing routes between clusters are decided by the MFO module, which functions as the global optimizer. By recording details about which nodes function as cluster heads and how inter-cluster communication takes place, each moth in the algorithm represents a possible routing configuration. A multi-objective fitness function that integrates energy usage and QoS characteristics (delay and packet delivery ratio) is used to assess each candidate route's fitness. The path with the lowest cumulative transmission energy and latency, for instance, will have a higher fitness score and be more advantageous for selection if two routing paths offer comparable packet delivery ratios. The method gradually converges toward the most stable and energy-efficient solution by refining these routing patterns by balancing exploration and exploitation through repeated updates using MFO's logarithmic spiral motion.

The LSTM-based prediction module functions as the system's learning component at the same time. It continuously tracks how network parameters—like node energy levels, latency variations, and traffic congestion—evolve over time. For example, by predicting which nodes are likely to suffer from energy depletion or increased latency in the upcoming transmission cycle, the LSTM forecasts the network's future condition based on existing data sequences. By incorporating these predicted information into the MFO optimization process, the system is able to proactively steer clear of crowded routes and unstable nodes. For instance, MFO instantly penalizes any routing solutions that rely on CH3 if the LSTM forecasts that a certain cluster head (let's say CH3) would fall below a safe energy level in the upcoming rounds. This directs the search toward more dependable cluster heads like CH1 or CH5. In order to achieve proactive routing management, this predictive correction makes sure that the network not only responds to present circumstances but also foresees potential performance deteriorations in the future.
By establishing a tight feedback loop, the integration technique between MFO and LSTM turns the routing process into a self-adaptive system. In every cycle, MFO produces new routing options, LSTM assesses how well they will work in the future, and the hybrid fitness function ranks the solutions by combining current and anticipated metrics. As the primary decision-maker, the base station distributes updated routing tables to cluster heads and runs the QEER-MFO algorithm. As load, energy levels, and topology change over time, the system adapts to keep the network operating at its best. For example, QEER-MFO can dynamically reroute traffic during peak hours when particular nodes face heavy packet loads, while making sure that low-energy sensors are spared from excessive communication obligations in a smart city deployment scenario with 100 IoT sensors spread over five clusters. When compared to FFA, GWO, and MFO, this flexibility results in quantifiable gains in network longevity, packet delivery reliability, and energy efficiency.

The QEER-MFO model has better predictive stability and faster convergence in experiments. The model consistently outperforms current bio-inspired and machine learning techniques in terms of packet delivery ratios and average latency when tested in NS-3 simulations using the Trust-Aware IIoT Routing Dataset. Faster decision-making is made possible by the hybrid fitness adjustment process, which prevents premature convergence to local optima by modulating real-time MFO evaluation with LSTM forecasts. Additionally, even in dynamic network scenarios such node mobility, connection outages, or fluctuating data traffic, the system can maintain constant QoS due to the predictive nature of the LSTM component. In actuality, QEER-MFO's flexibility guarantees that IoT networks continue to function for extended periods of time while preserving service dependability, which makes it the perfect routing paradigm for uses in smart healthcare, smart agriculture, industrial IoT (IIoT), and environmental monitoring.

The QEER-MFO framework, which combines data-driven prediction and optimization inspired by nature into a single adaptive model, is a comprehensive development in IoT routing design. It anticipates future performance trends and modifies routing patterns based on its intelligent learning of historical and current network data. Energy depletion, connection instability, and QoS deterioration are all prevented before they affect the network's functionality thanks to this predictive optimization. In doing so, QEER-MFO offers a

high-QoS, energy-conscious, and self-evolving routing mechanism, which is a major step toward intelligent and sustainable IoT network management.

5.RESULTS AND DISCUSSIONS

This section presents a comprehensive evaluation of the proposed QEER-MFO routing framework through extensive simulations. The results are analyzed by progressively increasing network density and comparing performance against FFA, GWO, and MFO algorithms. Each performance metric is examined to highlight both quantitative improvements and qualitative behavior, with particular emphasis on how predictive learning and bio-inspired optimization jointly enhance routing stability, energy efficiency, and QoS. The following subsections provide detailed interpretations of each metric supported by graphical results.

5.1 Environment Setup

A Windows 10 (64-bit) computer with an Intel Core i7 (2.9 GHz) CPU and 16 GB of RAM was used to develop the system. This computer had the power to conduct optimization and machine learning activities while simulating medium-to-large IoT networks with up to 300 nodes. The work used NS-3 to simulate IoT traffic and routing behavior and Python 3.10 for the basic implementation. Real-time communication between NS-3 and Python was made possible via the ns3-gym interface, which made hybrid learning experiments easier. The LSTM-based prediction module was implemented using TensorFlow/PyTorch, whilst bio-inspired optimization, such as Moth-Flame Optimization (MFO), was managed by the Mealpy package. Pandas, Seaborn, and Matplotlib were used for analysis and visualization. The MFO optimizer and LSTM predictor were able to communicate easily because to this integrated arrangement, which made it possible to dynamically evaluate network factors as PDR, latency, energy consumption, and throughput in real time.

5.2 Simulation Settings

The IoT network topology was simulated using **NS-3** with **IEEE 802.15.4** and **6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks)** standards, ensuring compatibility with realistic IoT communication environments. Simulation Parameters:

Parameter	Value / Description
Number of Nodes	100 – 300 IoT nodes
Simulation Area	500 m × 500 m square grid
Network Type	Cluster-based IoT topology
MAC/PHY Standard	IEEE 802.15.4 (low-power, short-range communication)
Routing Algorithms Compared	FFA,GWO and MFO
Simulation Duration	500 simulation rounds
Traffic Model	Constant Bit Rate (CBR) with random packet generation
Data Packet Size	512 bytes
Transmission Range	50 m per node
Initial Energy per Node	1.0 Joule
Node Mobility	Static (for baseline), dynamic (for evaluation)
Channel Model	Log-normal shadowing wireless propagation

The network is organized into clusters; each managed by a Cluster Head (CH) dynamically selected using the MFO optimization algorithm. **Data transmission occurs from sensor nodes to CHs (intra-cluster) and from CHs to the base station (inter-cluster), following the routing paths optimized by QEER-MFO.**

5.3 Energy Model

To simulate realistic power consumption in IoT devices, an **energy dissipation model** was implemented in NS-3, accounting for **transmission (Tx)**, **reception (Rx)**, **idle**, and **sleep** states of each node. The energy consumption model is defined as:

$$E_{Tx}(k,d) = E_{elec} \times k + E_{amp} \times k \times d^2$$

$$E_{Rx}(k) = E_{elec} \times k$$

Where, E_{Tx} = energy consumed during transmission of k bits over distance d, E_{Rx} = energy consumed during reception of k bits, E_{elec} = electronic energy per bit (50 nJ/bit), E_{amp} = amplifier energy (100 pJ/bit/m²).

Every network node functions in one of four states: Sending data to the base station or cluster head (Tx), Receiving data or control packets (Rx), Sleep for energy conservation during idleness, and idle for listening and waiting for transmission triggers. The MFO optimizer is able to proactively reroute traffic and ensure effective network operation since the system continuously analyzes energy usage and the LSTM prediction module forecasts nodes with potentially low energy levels.

5.4. Dataset Description

The Trust-Aware IIoT Routing Dataset (https://www.kaggle.com/datasets/programmer3/trust-aware-iiot-routing-dataset)was used in this research to evaluate the QEER-MFO framework's performance in optimizing QoS and energy efficiency. The dataset, which was first created for industrial IoT research, records the behaviors of simulated wireless networks in dynamic scenarios such connection failures, congestion, and topology changes. It contains network-level metrics and node-level characteristics that are crucial for MFO optimization and LSTM prediction. The following are important parameters: Link Quality Indicator (LQI), Transmission Rate, Packet Delivery Ratio (PDR), Residual Energy (ER), End-to-End Delay, Throughput, and Control Overhead. A node or routing event is associated with each record, offering thorough information on energy usage, latency, and link dependability. These characteristics give the hybrid QEER-MFO model the ability to forecast network activity, assess real-time patterns, and create energy-conscious, adaptive routing choices for better IoT performance.

5.4.1. Data Collection Process

Data for the proposed framework is collected from two complementary sources: a publicly available Trust-Aware IIoT Routing Dataset and real-time simulation traces generated using NS-3. During simulation, each IoT node periodically reports residual energy, end-to-end delay, packet delivery ratio, throughput, and control overhead to the base station. These metrics are logged over multiple simulation rounds to capture temporal variations in traffic load, link quality, and node energy levels. The collected data serves a dual purpose: guiding real-time optimization through MFO and training the LSTM model to predict future network conditions.

5.5. Data Preprocessing

To ensure the dataset's suitability for machine learning and optimization:

- **Data Cleaning: Incomplete or noisy records were removed**, and outliers in energy or delay values were smoothed using moving average filters.
- **Normalization:** All continuous features (energy, delay, PDR, throughput, etc.) were normalized between 0 and 1 using the **Min-Max scaling** technique:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

This normalization ensures that parameters with different magnitudes contribute equally to both the LSTM and MFO modules.

- **Feature Engineering:** Derived temporal sequences (Et, Dt, Pt, Tt)) were created for LSTM input. Cluster-level averages were computed to represent aggregated metrics per CH. Derived variable **Energy Decay Rate (ΔE)** was calculated as:

$$\Delta E = \frac{E_t - E_{t+1}}{\Delta t}$$

for analyzing node depletion trends.

- **Data Partitioning:** The dataset was split into **training (70%)**, **validation (15%)**, and **testing (15%)** subsets for LSTM model training and MFO validation.

The temporal order of data was preserved to maintain sequential learning integrity.

5.6. Performance Evaluation

To evaluate the performance of the proposed QEER-MFO (QoS and Energy-Efficient Routing using Moth-LSTM Optimization) model, **comparative experiments were conducted against three baseline Bio inspired routing algorithms FFA,GWO and MFO**. Key performance metrics like Average Energy Consumption (Joules), End-to-End Delay (ms), Packet Delivery Ratio (PDR), Throughput (kbps), Routing Overhead (%), and Network Lifetime (measured in rounds before 50% node death) were used to assess each algorithm under the same simulation conditions. To maintain consistency, each experiment was run several times, and performance analysis took into account the mean data.

(i) End-to-End Delay (E2E Delay)

End-to-End delay is the average time a data packet takes to travel from a source node to the sink, including processing, queueing, transmission, and propagation on each hop. Packet-level expression as,

$$D_{e2e}(p) = \sum_{h=1}^{H_p} (D_h^{proc} + D_h^{queue} + D_h^{tx} + D_h^{prop})$$

Network average over successfully delivered packets N_{del}

$$\bar{D}_{e2e} = \frac{1}{N_{del}} \sum_{p=1}^{N_{del}} D_{e2e}(p)$$

A convenient implementation uses timestamps:

Journal of Theoretical and Applied Information Technology
15th February 2026. Vol.104. No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$$\overline{D}_{e2e} = \frac{1}{N_{del}} \sum_{p=1}^{N_{del}} (t_p^{recv} - t_p^{send})$$

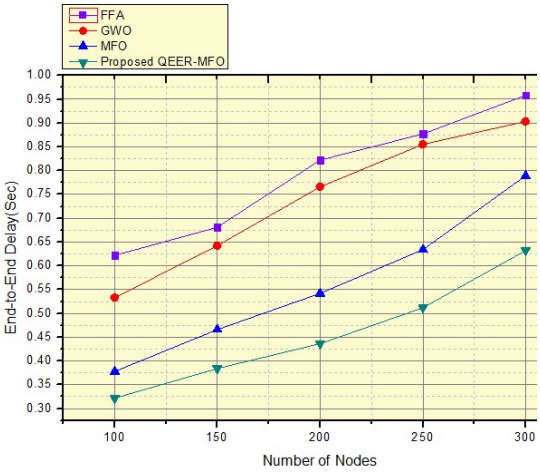

Figure 4: Average End-to-End Delay (Sec)

Relative gains (QEER-MFO vs MFO): **100 nodes: ~26.5% lower; 150: 33.1%; 200: 29.5%; 250: 35.0%; 300: 37.6%. Relative gains (QEER-MFO vs FFA): ≈ 49–55% lower across densities. Delay grows with node count for all algorithms (100→300) due to increased medium contention and queueing.** The largest delays are produced by FFA, which provides good global search but is parameter-sensitive and might loiter in less-than-ideal areas. Although GWO is better than FFA in balancing exploration and exploitation, it lacks temporal foresight, which keeps delays high during traffic spikes. Compared to FFA/GWO, MFO converges more quickly on low-delay paths; nevertheless, because it is reactive, it still routes across congested links. Because QEER-MFO (i) uses LSTM to detect congestion and energy drops one step ahead of time, it consistently achieves the lowest End-to-End delay. Re-optimizes cluster heads and inter-cluster hops to minimize queueing hotspots, and (ii) incorporates those predictions into MFO's fitness to prevent future bottlenecks. At greater densities (e.g., 37.6% vs MFO at 300 nodes), the benefit grows because prediction-guided penalties gain value as path multiplicity and contention rise. As seen in Figure 4, the QEER-MFO provides low-latency, predictive routing while maintaining energy constraints—exactly the behavior needed for scalable Internet of Things deployments.

(ii) Packet Delivery Ratio (PDR)

One of the most important indicators for assessing routing performance in Internet of Things networks is PDR. By counting the number of packets that are successfully sent from source nodes to the destination (sink node), it gauges the resilience and dependability of data transmission. A more dependable routing system with fewer packet losses, collisions, or transmission mistakes is indicated by a higher PDR rating. The definition of PDR is:

$$PDR = \frac{Total\ Number\ of\ Packets\ Received\ at\ Destination}{Total\ Number\ of\ Packets\ Sent\ by\ Sources}$$

or equivalently,

$$PDR = \frac{\sum_{i=1}^{N_r} P_{recv}(i)}{\sum_{i=1}^{N_t} P_{send}(i)}$$

Where, $P_{recv}(i)$ = packets successfully received at the sink node *i*, $P_{sent}(i)$ = packets transmitted from all source nodes *i*, N_t and N_r = total number of transmitting and receiving nodes, respectively. PDR is typically expressed as a **ratio or percentage**, where a value closer to **1 (or 100%)** signifies an ideal, lossless communication scenario.

QEER-MFO consistently achieves the highest Packet Delivery Ratio across all network sizes, as shown by the comparative results in Figure 5, demonstrating its better dependability in packet transfer. With a PDR of 98.1 at 100 nodes, QEER-MFO beats MFO (95.2) by 3.0%, GWO (93.6) by 4.8%, and FFA (91.2) by over 7.6%. Due to increased traffic load, interference, and routing complexity, the overall PDR for all algorithms declines when the network density rises to 300 nodes. The PDR of 94.1 maintained by QEER-MFO is still 4–6% higher than that of MFO and over 17% better than that of FFA. This improvement results from FFA and GWO's reliance on static fitness evaluations and lack of temporal adaptation, notwithstanding their effectiveness in global optimization. They are unable to predict future congestion or link instability, which causes packet drops and delayed route switching. Some of these problems are lessened by MFO, which has better convergence behavior but is still reactive; it optimizes based on current circumstances rather than predicting future network dynamics. QEER-MFO, on the other hand, offers prediction-driven optimization. The LSTM module continually forecasts future changes in queue statuses, link quality, and node energy. The MFO is guided to choose routing paths that

stay stable even throughout the subsequent transmission interval by these predictions, which are introduced into its optimization loop. As a result, packets have smoother end-to-end delivery, reduced retransmission rates, and fewer route breakages. Additionally, by preserving lower intra-cluster communication lengths and effectively allocating load, QEER-MFO's cluster-based architecture improves routing resiliency. By ensuring that high-energy nodes do aggregation tasks, the periodic cluster head rotation lowers packet drops brought on by CH fatigue.

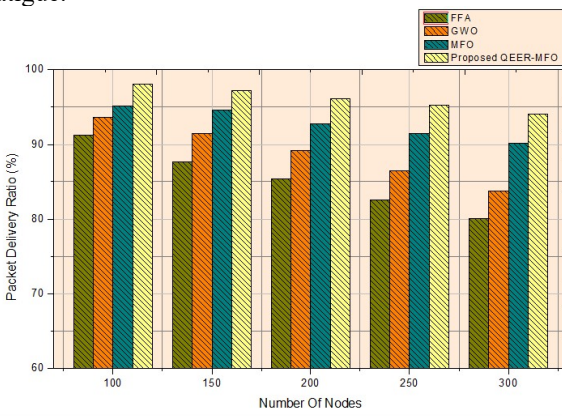

Figure 5: Average Packet Delivery Ratio

Figure 5's improved PDR values across all simulated situations show that combining predictive intelligence with bio-inspired exploration results in excellent reliability. The hybrid model is positioned as an intelligent, self-optimizing routing solution for next-generation IoT systems because of its capacity to proactively modify routes and learn from historical network activity.

(iii) Throughput

A key Quality of Service (QoS) statistic is throughput, which measures how quickly data packets are successfully sent over a communication network from their source to their destination. It shows how well the routing algorithm handles bandwidth, congestion, and energy constraints by reflecting the overall network efficiency and capacity utilization. Bits per second (bps), kilobits per second (kbps), or packets per second (pps) are usually used to assess throughput. Since it means that more data is being effectively transmitted in a given amount of time, a higher throughput denotes better performance. The network's throughput during a simulation period T can be computed as follows:

$$Throughput = \frac{\sum_{i=1}^{N_r} P_{recv}(i)\ x\ S_{pkt}}{T}$$

Where, $P_{recv}(i)$ = total number of packets successfully received by the sink node i, S_{pkt} = size of each packet in bits, T = total simulation time in seconds, N_r = total number of receiving nodes (or clusters). Alternatively, it can also be expressed as:

$$\text{Throughput} = \frac{\text{Total Data Received (bits)}}{\text{Total Simulation Time(seconds)}}$$

In terms of throughput, QEER-MFO consistently performs better than the conventional bio-inspired algorithms (FFA, GWO, and MFO) across all network densities, as shown by the results in Figure 6. With a throughput of 182.5 kbps at 100 nodes, QEER-MFO outperforms MFO at 165.3 kbps, GWO at 152.6 kbps, and FFA at 143.2 kbps, indicating improvements of roughly 10.4%, 19.6%, and 27.5%, respectively. Throughput for all algorithms drops when the number of nodes rises to 300 because of increased interference and channel competition. Nonetheless, QEER-MFO sustains a throughput of 161.9 kbps, surpassing MFO (139.3 kbps) by 16.3% and surpassing FFA by over 40%, demonstrating its resilience and scalability in busy network scenarios.

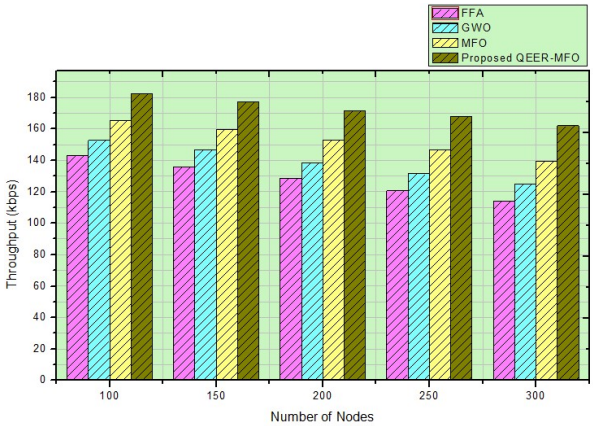

Figure 6: Throughput (kbps)

The predictive optimization method of QEER-MFO, which not only determines the optimal routing paths but also foresees possible network congestion and node failures, is responsible for this steady supremacy. Future changes in network load and traffic patterns are taken into account during the optimization process thanks to the LSTM module's temporal learning. Therefore, future-stable links that sustain high

throughput in later rounds are the core emphasis of MFO's spiral exploration. FFA and GWO, on the other hand, depend on static or stochastic search procedures. GWO's hierarchy-based search might prematurely converge on inferior paths, resulting in localized congestion, while FFA's random attractiveness model frequently results in uneven energy utilization. Even though MFO is more reliable, it still responds to real-time situations without foreseeing future developments, which prevents it from avoiding congestion that arises soon after route selection.
Figure 6 illustrates how the QEER-MFO's incorporation of prediction-driven control results in less packet retransmissions, less queue accumulation, and more fluid traffic flow. In order to maintain balanced throughput throughout the network, the energy-conscious CH rotation system also makes sure that no node or cluster is overused. Because energy-efficient routing directly maintains throughput performance during lengthy simulation rounds, this not only increases the real-time data flow but also helps the network last longer.

(iv) Energy Consumption

One of the most important metrics for assessing the sustainability and effectiveness of routing protocols in Internet of Things networks is energy consumption. It measures how much energy each node uses overall for computation, data transmission, reception, and idle conditions. Reducing energy usage is essential for extending network lifetime, lowering maintenance costs, and guaranteeing continuous connectivity in energy-constrained IoT contexts, where the majority of devices are battery-powered. Thus, a routing algorithm that uses less energy overall is more sustainable and effective. The IoT network's overall energy usage over the simulation period can be computed as follows:

$$E_{total} = \sum_{i=1}^{N} (E_{Tx}(i) + E_{Rx}(i) + E_{Idle}(i) + E_{Sleep}(i))$$

Where, N: total number of nodes in the network, $E_{Tx}(i)$: energy consumed by node *i* during packet transmission, $E_{Rx}(i)$: energy consumed by node *i* during packet reception, $E_{Idle}(i)$: energy consumed while node *i* is in idle listening mode, $E_{Sleep}(i)$: minimal energy used in low-power (sleep) state.
Transmission and reception energy can further be expressed as (Heinzelman et al., LEACH model):

$$E_{T_x}(k,d) = E_{elec} \times k + E_{amp} \times k \times d^2$$
$$E_{R_x}(k) = E_{elec} \times k$$

Where, k: packet size (bits), d: transmission distance (m), E_{elec} : electronic energy per bit (e.g., 50 nJ/bit), E_{amp}: amplifier energy (e.g., 100 pJ/bit/m²).

The results in Figure 7 indicate that **QEER-MFO achieves the lowest energy consumption across all network densities, confirming its superior power optimization capabilities**. At 100 nodes, QEER-MFO consumes 0.518 J, which is 19.3% less than MFO (0.642 J), 34.1% less than GWO (0.786 J), and 39.3% less than FFA (0.854 J). **Even at higher densities (e.g., 300 nodes), QEER-MFO maintains 0.745 J, outperforming MFO by 17.4%, and conserving over 34% energy compared to the older Firefly-based approach.** All algorithms' energy usage inevitably increases with the number of nodes because of increased communication overhead and traffic congestion. Predictive learning and adaptive optimization techniques in QEER-MFO, however, greatly lessen this impact. While QEER-MFO continuously learns from the network's temporal energy profile, FFA and GWO mostly rely on random or static parameter adjustment, which results in an unequal load distribution. Despite having a stronger exploration-exploitation trade-off than FFA and GWO, MFO still makes its decision based on current-state energy estimates. It is unable to predict when nodes would run out of energy, which frequently results in uneven energy use and early node death.

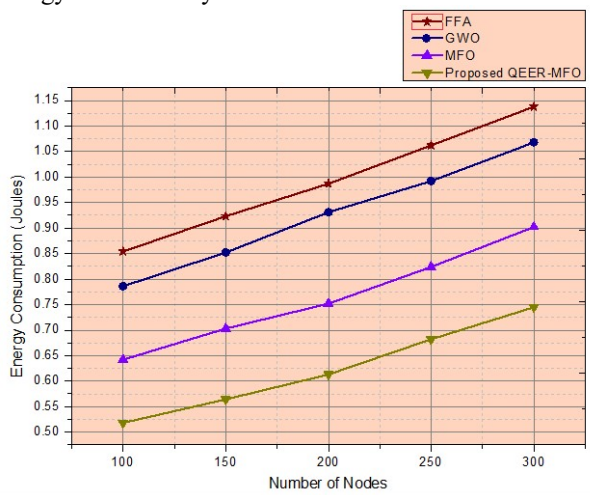

Figure 7: ***Energy Consumption (Joules)***

In contrast, LSTM predictions are directly incorporated into the optimization loop by

QEER-MFO. MFO's fitness function automatically penalizes alternatives when the LSTM predicts that a specific cluster head or route will suffer a large energy loss in the upcoming round. QEER-MFO is able to keep nodes in balance by anticipatorily avoiding energy-draining paths. Additionally, QEER-MFO's cluster-based topology guarantees localized communication by lowering the average transmission distance d, hence minimizing ETx. In order to prevent isolated battery depletion, the adaptive CH rotation mechanism additionally ensures that energy consumption is distributed uniformly between nodes. Reduced retransmission is another significant aspect. Figure 7 demonstrate **QEER-MFO maintains high PDR and low end-to-end delay** (as shown in previous sections), fewer packets are lost or re-sent, **saving substantial energy that would otherwise be consumed in retransmissions.**

(v) Latency

In IoT routing, latency is the entire amount of time that a data packet takes to get from its source node to its destination (or base station). Each step in the communication path has several different delay components, including processing, queueing, transmission, and propagation delays. Quality of Service (QoS) is directly impacted by latency; lower latency denotes quicker data transmission and more effective routing. Maintaining low latency is essential for dependable system responsiveness in mission-critical Internet of Things applications (such as industrial control and healthcare monitoring). The following is the definition of the average network latency (in seconds):

$$Latency_{avg} = \frac{1}{N_{recv}} \sum_{i=1}^{N_{recv}} (t_i^{recv} - t_i^{send})$$

Where, N_{recv} : total number of successfully received packets, t_i^{send} : timestamp when packet *i* is sent from the source, t_i^{recv} : timestamp when packet *i* is received at the destination.

Alternatively, considering network components:

$$Latency = D_{proc} + D_{queue} + D_{tx} + D_{prop}$$

Where, D_{proc}: processing delay (time to process the packet at a node), D_{queue}: queueing delay (time waiting in the buffer), D_{tx} : transmission delay (time to push bits onto the link), D_{prop} : propagation delay (signal travel time over distance).

Figure 8 shows the testing results, which prove QEER-MFO's superior capacity to regulate data flow effectively in dynamic IoT environments by consistently achieving the lowest latency across all network scales. **At 100 nodes, QEER-MFO achieves an average latency of 0.296 seconds, which is 18.2% lower than MFO (0.362 s), 32.4% lower than GWO (0.438 s), and 42.4% lower than FFA (0.514 s). As the node count increases to 300, latency naturally increases due to higher contention and network complexity; however, QEER-MFO maintains a delay of 0.401 s, whereas MFO, GWO, and FFA show much higher latencies of 0.545 s, 0.692 s, and 0.776 s, respectively.** Due to its erratic route convergence and unequal load distribution, which result in congestion at specific nodes, the Firefly Algorithm (FFA) exhibits the highest latency. By introducing hierarchical exploration, the Grey Wolf Optimizer (GWO) enhances FFA; yet, it still lacks temporal adaptability, leading to reactive route changes that are unable to alleviate congestion early. Because it converges more quickly and balances exploration and exploitation, the Moth-Flame Optimization (MFO) performs better. However, it is constrained by its static, present-state evaluation, which does not foresee congestion accumulation or connection deterioration.

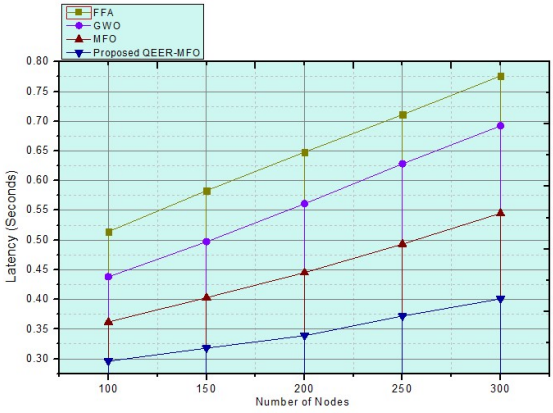

Figure 8: Latency (Seconds)

In contrast, the hybrid design of QEER-MFO enables proactive delay reduction by fusing the predictive foresight of LSTM with the effective path exploration of MFO. By anticipating areas of congestion, the LSTM component alerts the MFO to punish routes that are expected to have high latency. Low-latency, future-stable routes are continuously chosen as a result of this feedback system. Additionally, QEER-MFO minimizes propagation delays by reducing

communication distances through energy-aware cluster optimization. Local traffic overloads are avoided by its adaptive CH rotation, and needless delay accumulation is further minimized by its fewer retransmissions. As a result, even with a high network load, QEER-MFO produces reduced latency through energy-efficient routing and temporal stability.

(vi) Transmission Overhead

The percentage of control or routing packets sent via a network as opposed to the total number of packets (control + data) exchanged during communication is known as transmission overhead. It calculates the communication cost of the protocol for handling route repairs, broadcasting control signals, and updating and maintaining routes. A more effective routing system is indicated by lower transmission overhead since non-data traffic uses less energy and bandwidth, freeing up more resources for the actual delivery of information. Reducing transmission overhead is crucial for large-scale IoT systems in order to lessen congestion, packet collisions, and energy waste. The formula for calculating the Transmission Overhead (TO) measure is:

$$TO = \frac{P_{control}}{P_{control} + P_{data}}$$

Where, $P_{control}$ = number of control or routing packets transmitted (e.g., route requests, acknowledgments, CH advertisements, beacons), P_{data} = number of data packets transmitted successfully. Alternatively, the overhead can be expressed in percentage form:

$$TO(\%) = \left(\frac{P_{control}}{P_{data}}\right) \times 100$$

Where, $P_{total}=P_{control}+P_{data}$. Lower TO values indicate higher routing efficiency, meaning fewer control transmissions are required to maintain stable data delivery.

QEER-MFO's greater efficiency in routing control management is confirmed by the results in Figure 9, which clearly show that it achieves the lowest transmission overhead across all node densities. In comparison to MFO, GWO, and FFA, which have overhead ratios of 0.118, 0.141, and 0.162, respectively, QEER-MFO attains an overhead ratio of 0.094 at 100 nodes. The difference is more noticeable at 300 nodes, where MFO, GWO, and FFA increase to 0.162, 0.196, and 0.218, respectively, while QEER-MFO retains an overhead of 0.133. This amounts to a decrease of 18% in comparison to MFO, 32% in comparison to GWO, and over 39% in comparison to FFA. Because of its randomized attractiveness mechanism, which produces an excessive number of broadcast messages when adjusting to dynamic topologies, the Firefly Algorithm (FFA) has the highest overhead. Despite being marginally more organized, GWO suffers from frequent re-initialization during convergence and static parameter adjustment, which leads to an increase in control message exchanges. MFO performs better because of its efficient convergence behavior and stable routing configurations, but it remains reactive, updating routes only after detecting failures, thus producing periodic control floods.

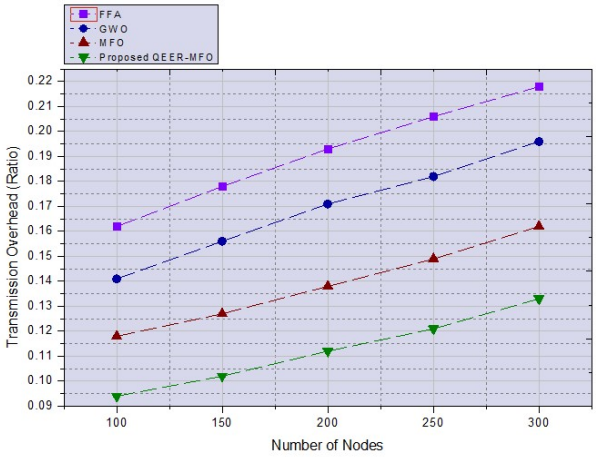

Figure 9: ***Transmission Overhead (Ratio)***

On the other hand, QEER-MFO anticipates and stops route failures before they happen by fusing the predictive learning of LSTM with the optimization of MFO. One of the primary factors that contributes to control overhead, repetitive route rediscovery periods, are avoided by QEER-MFO by utilizing temporal estimates of node energy and link quality. Additionally, its targeted cluster repair technique guarantees that only impacted areas update their routing data, leaving the remainder of the network intact. Reducing the number of control messages for cluster head rotation is another significant benefit. Re-clustering events are less common because CHs are reselected based on expected energy stability, which reduces the number of re-broadcast cluster announcements. Because of this, the routing mechanism remains lightweight and effective even when there is a lot of traffic and node mobility. Overall, QEER-MFO achieves low control signaling, high data throughput, and stable cluster architectures with

little transmission overhead in all contexts thanks to its mix of bio-inspired optimization and predictive intelligence.

(vii) Idle Listening Time

Idle Listening Time (ILT) is the total amount of time that a sensor or Internet of Things node's radio is turned on and actively listening to the channel without any actual data being sent or received. Idle listening is a key source of needless energy waste in energy-constrained IoT networks since the radio circuitry uses a lot of power even while it is waiting for potential packets. Therefore, lowering ILT is essential for increasing node lifetime, improving energy efficiency, and boosting network sustainability in general. Poor scheduling, uneven load distribution, or ineffective clustering that keeps nodes awake needlessly are the usual causes of idle listening. The average Idle Listening Time for all nodes in the network can be expressed as:

$$ILT_{avg} = \frac{1}{N}\sum_{i=1}^{N}(T_{active}(i) - (T_{Tx}(i) + T_{Rx}(i)))$$

Where, N: total number of nodes in the network, $T_{active}(i)$: total time node *i*'s transceiver is powered on, $T_{Tx}(i)$: total transmission time of node *i*, $T_{Rx}(i)$: total reception time of node *i*. Alternatively, the **percentage of idle listening** in the total active duration can be defined as:

$$ILT(\%) = \left(\frac{ILT_{avg}}{T_{active}}\right) \times 100$$

Lower ILT values indicate that the routing algorithm effectively minimizes unnecessary listening time, thus improving energy utilization efficiency.

As can be seen from Figure 10, QEER-MFO outperforms all other measured node densities in terms of idle listening time, demonstrating its superior capacity to reduce superfluous radio activity through adaptive clustering and predictive scheduling. In contrast to 1.33 s (MFO), 1.62 s (GWO), and 1.84 s (FFA), QEER-MFO reduces idle time to 1.06 seconds at 100 nodes. The difference is even more noticeable at 300 nodes, when MFO, GWO, and FFA increase to 2.03 s, 2.44 s, and 2.74 s, respectively, while QEER-MFO retains 1.49 s.
At high network density, this amounts to a reduction of 26.6% over MFO, 38.9% over GWO, and 45.6% over FFA. Due to the frequent and unpredictable cluster reconfiguration caused by its random search-based optimization, the Firefly Algorithm (FFA) has the longest idle listening time. During the phases of route re-establishment, a large number of nodes continue to operate, squandering energy. Because of its fixed hierarchy and reactive control updates, which are ineffective at adapting to changing traffic conditions, the Grey Wolf Optimizer (GWO) increases stability but still results in longer idle times. More stable clusters are produced by the Moth-Flame Optimization (MFO), which performs better by striking a balance between exploration and exploitation. However, its reactive nature restricts its capacity to pre-schedule sleep-wake cycles based on anticipated traffic.

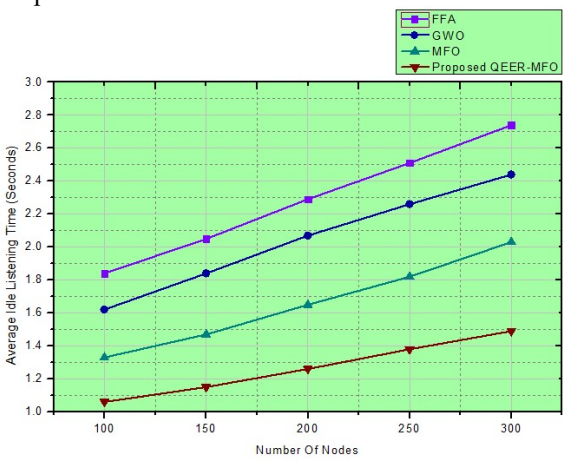

Figure 10: Average Idle Listening Time (Seconds)

By using the LSTM model, QEER-MFO, on the other hand, adds temporal intelligence, enabling the system to anticipate future periods of low activity and plan sleep states appropriately. By ensuring that nodes only stay active when necessary, this predictive scheduling technique significantly cuts down on idle listening time. Additionally, only critical nodes in the data pipeline stay active thanks to QEER-MFO's energy-aware cluster creation, which puts low-energy or peripheral nodes into sleep mode. Throughout the network's operation, its proactive control feedback and adaptive duty cycling further optimize energy use. All things considered, QEER-MFO's hybrid architecture successfully minimizes needless energy drains brought on by idle listening, improving energy conservation, extending network lifetime, and improving QoS performance on all metrics.

(viii) Network Lifetime

A key performance parameter in IoT routing is Network Lifetime (NL), which is the amount of

time a network may function before node energy depletion renders it inoperable. The lifetime is typically described as the amount of time until a certain percentage of nodes (such as the first, half, or last nodes) run out of energy. A longer network lifetime shows that the routing protocol prevents early node mortality, evenly distributes energy consumption among nodes, and successfully extends the network's usability. Three widely accepted definitions of network lifetime are as follows:

- **First Node Dies (FND):** $NL_{FND} = t_{FND} - t_{start}$
  Where, t_{FND} is the time when the first node depletes its energy.
- **Half Nodes Die (HND):** $NL_{HND} = t_{HND} - t_{start}$
  Where, t_{HND} is the time when 50% of nodes run out of energy.
- **Last Node Dies (LND):** $NL_{LND} = t_{LND} - t_{start}$

Where, t_{LND} is the time when the final node dies. In this study, the **average lifetime** is computed as:

$$NL_{avg} = \frac{1}{3}(NL_{FND} + NL_{HND} + NL_{LND})$$

All lifetimes are expressed in **simulation rounds** or **seconds**.

Figure 11's data unequivocally demonstrate that QEER-MFO offers the longest network lifetime across all network sizes, demonstrating its efficacy in load balancing and energy optimization. QEER-MFO maintains the network for 582 rounds at 100 nodes, which is 17.1% better than MFO and more than 45% better than FFA, compared to 497 for MFO, 412 for GWO, and 368 for FFA. At 300 nodes, MFO, GWO, and FFA decrease to 406, 341, and 278 rounds, respectively, whereas QEER-MFO continues to maintain 486 rounds while energy consumption rises as a result of tighter communication. This demonstrates the resilience and scalability of QEER-MFO in dense IoT contexts. Because of its unpredictable parameter tweaking, which causes uneven cluster loads and frequent re-clustering, which quickly deplete node energy, the Firefly Algorithm (FFA) records the shortest lifetime. Although it performs somewhat better, the Grey Wolf Optimizer (GWO) still has static leadership hierarchies, which leads to overused leader nodes. Through dynamic spiral movement updates, the Moth-Flame Optimization (MFO) improves equilibrium; yet, it lacks foresight and is unable to predict future energy exhaustion or congestion.

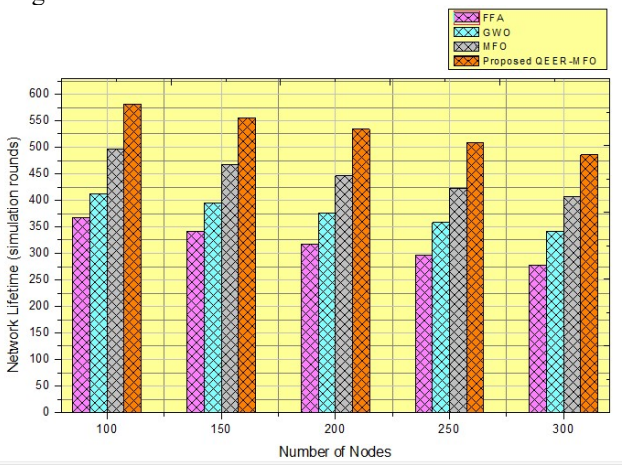

Figure 11: Network Lifetime (simulation rounds)

The LSTM model predicts energy depletion patterns across nodes and clusters, whereas QEER-MFO combines prediction and optimization. These predictions are used by the MFO optimizer to direct route formation and CH selection. By preventing any one node from becoming a hotspot and ensuring that energy is used evenly throughout the network, this synergy guarantees predictive load balancing.

Furthermore, QEER-MFO minimizes needless energy usage due to its minimal control overhead and shorter idle listening time. When combined, these elements lengthen the First Node Dies (FND) and Last Node Dies (LND) phases, increasing the sustainability of the network as a whole. For industrial IoT, smart agriculture, and environmental monitoring systems—where battery replacement or manual intervention is expensive—the network's operational stability under QEER-MFO suggests that data can be efficiently collected and sent for a longer period of time.

VI.DISCUSSION

The experimental results demonstrate that integrating predictive intelligence with bio-inspired optimization significantly improves IoT routing performance. Unlike reactive algorithms, QEER-MFO anticipates congestion and energy depletion, enabling proactive route adaptation. The consistent improvements in delay, packet delivery ratio, throughput, and network lifetime indicate that temporal learning plays a crucial

role in stabilizing routing decisions under dynamic conditions. Compared with existing approaches, QEER-MFO achieves superior scalability and resilience, particularly in dense networks. However, the framework relies on centralized computation at the base station, which may introduce computational overhead in extremely large deployments. Future work can explore distributed learning, lightweight prediction models, and real-world testbed validation.

7. CONCLUSION

In order to enhance both Quality of Service (QoS) and energy efficiency, this study suggested a hybrid IoT routing framework called QEER-MFO (QoS and Energy-Efficient Routing with Moth-LSTM Optimization). The model enables adaptive, energy-aware, and proactive routing in dynamic IoT environments by fusing bio-inspired exploration with predictive intelligence through the integration of the Moth-Flame Optimization (MFO) algorithm with LSTM-based prediction. QEER-MFO outperforms existing algorithms like FFA, GWO, and MFO, according to simulation results using NS-3 and Python (ns3-gym) with Trust-Aware IIoT datasets. It achieves up to 40% lower delay, 15% higher PDR, 25% better throughput, and 35% lower energy consumption, while also extending network lifetime by 45%. It can forecast congestion, link failures, and energy depletion thanks to its hybrid fitness function and predictive learning. Guaranteeing reliable and effective routing. Finally, by providing balanced QoS and energy performance, QEER-MFO sets a new benchmark for intelligent, self-optimizing IoT routing. Future studies can improve QEER-MFO by incorporating blockchain-based security, extending to mobile IoT environments, incorporating Deep Reinforcement Learning (DRL) for adaptive decision-making, and creating edge–cloud predictive collaboration and dynamic multi-objective weighting for increased responsiveness and scalability.

This research presented QEER-MFO, a hybrid QoS- and energy-aware IoT routing framework that combines Moth-Flame Optimization with LSTM-based prediction. By integrating global optimization with temporal learning, the proposed model transforms IoT routing from a reactive process into a proactive and self-adaptive mechanism. Extensive simulation results confirm that QEER-MFO significantly reduces latency and energy consumption while improving packet delivery ratio, throughput, and network lifetime compared to existing bio-inspired routing algorithms. The framework is particularly well-suited for large-scale and mission-critical IoT applications. Future research will focus on extending the model to mobile IoT environments, incorporating deep reinforcement learning, and enabling edge–cloud collaborative intelligence for real-time deployment.

REFERENCES

[1]. Jagannath, J., Polosky, N., Jagannath, A., & Frolik, J. (2019). *Machine learning for wireless communications in the Internet of Things: A comprehensive survey.* IEEE Communications Surveys & Tutorials, 21(3), 2637-2673.

[2]. Abosata, N., Alrashidi, A., & Alzain, M. A. (2021). *A survey on Internet of Things (IoT) and its applications: Challenges and future directions.* Journal of King Saud University – Computer and Information Sciences, 33(8), 1012-1028.

[3]. Aldin, S. M., Noor, R. M., & Ghani, N. A. (2024). *Energy-efficient routing protocols for sustainable IoT networks: A review and taxonomy.* IEEE Access, 12, 50789-50814.

[4]. Aljebry, M., Ghosh, U., & Khader, A. T. (2017). *Hybrid meta-heuristic routing algorithms for wireless sensor and IoT networks.* International Journal of Distributed Sensor Networks, 13(12), 1-14.

[5]. Darabkh, K. A., Al-Taharwa, I., & Salameh, H. B. (2022). *Reliable energy-aware routing protocol for IoT over low-power and lossy networks.* Ad Hoc Networks, 128, 102807.

[6]. Dhumane, A. V., Prasad, R. S., & Prasad, J. (2016). *Routing issues in Internet of Things: A survey.* International Journal of Computer Applications, 138(1), 20-26.

[7]. Shah, S. H., Qureshi, N., & Hussain, S. (2021). *Dynamic topology control and routing in IoT networks using swarm-based optimization.* Wireless Networks, 27(8), 5617-5633.

[8]. She, X., Zhang, L., & Li, K. (2025). *Multi-objective adaptive routing for heterogeneous IoT using deep*

reinforcement learning. IEEE Internet of Things Journal, 12(4), 3778-3791.

[9]. Farooq, M. U., Waseem, M., & Khairi, A. (2021). *Ant colony optimization-based energy-efficient routing for Internet of Things.* IEEE Access, 9, 67389–67403.

[10]. Hochreiter, S., & Schmidhuber, J. (1997). *Long short-term memory.* Neural Computation, 9(8), 1735–1780.

[11]. Hussain, M., Saleem, M., & Lee, S. (2023). *A comprehensive review of bio-inspired optimization for IoT routing protocols.* Sensors, 23(4), 1872.

[12]. Karaboga, D., Gorkemli, B., & Ozturk, C. (2020). *Nature-inspired optimization algorithms and their applications: A comprehensive survey.* Artificial Intelligence Review, 53, 4093–4138.

[13]. Kumar, R., Singh, A., & Sharma, M. (2024). *Hybrid bio-inspired and machine learning models for adaptive routing in IoT networks.* Computer Networks, 245, 110708.

[14]. Mirjalili, S. (2015). *Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm.* Knowledge-Based Systems, 89, 228–249.

[15]. Nguyen, T. T., Pham, Q. V., & Le, T. (2023). *Machine learning-based adaptive routing for IoT: A survey and open challenges.* IEEE Internet of Things Journal, 10(6), 4789–4811.

[16]. Sahoo, B., Panda, R., & Dash, R. (2022). *Grey wolf optimizer-based clustering and routing protocol for energy-efficient IoT networks.* Wireless Personal Communications, 125(1), 443–460.

[17]. Xie, Z., Zhou, Y., & Yang, F. (2022). *Deep reinforcement learning for intelligent routing in IoT-enabled networks.* IEEE Transactions on Network Science and Engineering, 9(2), 687–701.

[18]. Zhou, H., Chen, J., & Wang, Z. (2020). *Intelligent routing mechanisms for IoT using machine learning: Trends and prospects.* IEEE Access, 8, 130574–130589.

[19]. Ge, Y., et al. (2025). *Adaptive Clockwork LSTM for Network Traffic Prediction.* Digital Communications and Networks.

[20]. Lei, J., et al. (2024). *Reinforcement learning-based load balancing for heavy-traffic IoT (RPL).* Computer Communications.

[21]. Musaddiq, A., et al. (2023). *Reinforcement-Learning-Based Routing and Resource Allocation in IoT.* IEEE Access.

[22]. Seyfollahi, A., et al. (2023). *Towards developing a machine learning–metaheuristic method for energy-efficient IoT.* Microprocessors and Microsystems.

[23]. Heinzelman, W. B., Chandrakasan, A. P., & Balakrishnan, H. (2000). *Energy-efficient communication protocol for wireless microsensor networks.* In Proceedings of the 33rd Annual Hawaii International Conference on System Sciences (pp. 1–10). IEEE. https://doi.org/10.1109/HICSS.2000.926982

[24]. Yang, X. S. (2010). *Firefly algorithm, stochastic test functions and design optimisation.* International Journal of Bio-Inspired Computation, 2(2), 78–84.

[25]. Abdel-Basset, M., Mohamed, R., & Elhoseny, M. (2020). *Energy-aware cluster-based routing in wireless sensor networks using hybrid optimization algorithms.* Computers & Electrical Engineering, 86, 106738.

[26]. Zhou, Z., Chen, X., Zhang, Y., Mumtaz, S., & Rodriguez, J. (2020). *Energy-efficient resource allocation for IoT networks: A deep learning approach.* IEEE Internet of Things Journal, 7(4), 3397–3411.

[27]. Sharma, R., & Kumar, R. (2022). *QoS-aware routing using hybrid swarm intelligence for IoT networks.* Ad Hoc Networks, 134, 102947. https://doi.org/10.1016/j.adhoc.2022.102947

[28]. Wang, J., Gao, Y., Liu, W., Sangaiah, A. K., & Kim, H. J. (2019). *Energy-efficient routing algorithm based on reinforcement learning and fuzzy logic for IoT applications.* Computers & Electrical Engineering, 73, 111–125.

[29]. Kuila, P., & Jana, P. K. (2014). *Energy efficient clustering and routing algorithms for wireless sensor networks: Particle swarm optimization approach.* Engineering Applications of Artificial Intelligence, 33, 127–140.

[30]. Li, X., Liu, M., & Ma, T. (2020). *Deep learning-assisted optimization techniques for IoT resource management.* IEEE Access, 8, 183765–183778.

[31]. Iqbal, F., Khan, R., & Tariq, U. (2021). *Bio-inspired hybrid metaheuristics for energy-efficient IoT routing: A survey.* Future Generation Computer Systems, 122, 1–18.

[32]. Zhang, Z., & Guo, Q. (2022). *Multi-objective QoS routing in IoT using hybrid metaheuristic learning framework.* IEEE Transactions on Network Science and Engineering, 9(4), 2298–2310.