Edge Reinforced Learning Platform with Homomorphic Encryption and Swarm Intelligence for Ultra-Low Latency IoT Sensing and Cross-Device Communication

V. Thamilarasi *

Posted Date: 7 January 2026

doi: 10.20944/preprints202601.0513.v1

Keywords: edge computing; reinforcement learning; homomorphic encryption; swarm intelligence; internet of things (IoT); ultra-low latency; cross-device communication; privacy-preserving edge intelligence

 doi:10.20944/preprints202601.0513.v1

Article

Edge Reinforced Learning Platform with Homomorphic Encryption and Swarm Intelligence for Ultra-Low Latency IoT Sensing and Cross-Device Communication

V. Thamilarasi

Department of Computer Science, Sri Sarada College for women (Autonomous), Salem- 636016, Tamil Nadu, India; tamilomsiva@gmail.com

Abstract

This paper presents an edge-reinforced learning platform that combines reinforcement learning, homomorphic encryption, and swarm intelligence to support ultra-low latency IoT sensing and cross-device communication. In conventional IoT architectures, cloud-centric processing and centralized coordination introduce significant delays and expose sensitive data to intermediate entities, making them unsuitable for time-critical and privacy-sensitive applications. The proposed platform relocates intelligence to the network edge, where edge nodes learn adaptive policies for sensing, routing, and computation offloading based on local conditions and limited global feedback. To preserve confidentiality, IoT measurements and model updates are protected using homomorphic encryption, allowing aggregation and decision-making to be performed directly over encrypted data without revealing raw values. In parallel, swarm intelligence mechanisms orchestrate distributed cooperation among devices, enabling robust path selection, task allocation, and congestion avoidance through lightweight, bio-inspired interactions rather than centralized control. The integrated design is evaluated on realistic IoT scenarios with heterogeneous devices and dynamic traffic patterns. Results show that the edge-reinforced learning platform can significantly reduce end-to-end latency and jitter compared to cloud-based and non-learning edge baselines, while incurring acceptable computational overhead from encryption and maintaining strong privacy guarantees. The framework demonstrates that it is feasible to simultaneously achieve low latency, resilient cross-device coordination, and data confidentiality in large-scale IoT deployments.

Keywords: edge computing; reinforcement learning; homomorphic encryption; swarm intelligence; internet of things (IoT); ultra-low latency; cross-device communication; privacy-preserving edge intelligence

1. Introduction

The proliferation of IoT devices in domains such as industrial automation, smart cities, healthcare, and intelligent transportation has led to massive volumes of real-time data and stringent latency requirements. Many of these applications demand millisecond-level responses: a control loop in a factory robot, a collision warning in a vehicular network, or a clinical alert from a wearable device cannot afford the delays introduced by sending raw data to distant clouds for analysis [1]. At the same time, IoT deployments are increasingly heterogeneous and dynamic, comprising low-power sensors, mobile nodes, and edge gateways that operate under varying network conditions and resource constraints. Traditional cloud-centric architectures struggle in this setting, as they create communication bottlenecks, central points of failure, and significant privacy risks.

Edge computing has emerged as a key paradigm shift, bringing computation, storage, and intelligence closer to data sources. However, simply relocating static logic from the cloud to the edge

 doi:10.20944/preprints202601.0513.v1

is not sufficient. Edge nodes must be able to adapt their behavior in real time as workloads, channel conditions, and network topologies change. Reinforcement learning offers a way for edge nodes to continuously refine their policies based on feedback from the environment, but applying it directly in IoT systems raises several challenges, including limited computational capacity, the need for fast convergence, and the risk of exposing sensitive data [2]. These challenges are compounded when cross-device communication and coordination are required to achieve global performance objectives, such as network-wide latency minimization or energy balancing.

To address this, the proposed platform combines three complementary techniques. Edge reinforcement learning provides adaptive decision-making at the network periphery homomorphic encryption ensures that data remains confidential even while being processed and swarm intelligence offers a scalable, distributed coordination mechanism among devices and edge nodes. By integrating these components, the platform aims to support ultra-low latency IoT sensing and robust cross-device communication, without sacrificing security or scalability [3]. This introduction sets the stage for the detailed problem formulation, design, and evaluation that follow.

1.1. Background and Motivation

In many current IoT deployments, sensing and actuation are controlled by fixed rules or manually engineered heuristics. These approaches tend to be brittle, as they are tuned for typical conditions and cannot respond effectively to sudden changes in traffic load, interference, device mobility, or partial failures. For example, a smart grid substation may suddenly experience a surge of measurements during a disturbance, or a fleet of autonomous vehicles may have to adapt to a localized network congestion event. In such cases, static configurations can either overload the network or degrade the quality of service, resulting in delayed decisions and potential safety risks [4].

Reinforcement learning introduces a data-driven way to optimize such decisions by treating the edge node as an agent that interacts with its environment. It can learn, over time, which sensing rates, routing paths, or offloading strategies yield the best trade-off between latency, reliability, and resource consumption. Running learning algorithms at or near the edge further reduces reliance on long feedback loops to the cloud. At the same time, the sensitivity of IoT data has become a major concern. Sensor readings may reveal occupancy patterns in homes, operational states of industrial machinery, or personal health indicators. If these data are exposed at intermediate edge servers or gateways in plaintext form, they become attractive targets for attackers [5].

Homomorphic encryption provides a powerful countermeasure by enabling computation on ciphertexts. Edge nodes can aggregate and process encrypted measurements, produce encrypted outputs, and only the authorized endpoints with decryption keys can retrieve the underlying values. This fits naturally with multi-tenant and federated IoT deployments where infrastructure providers and application owners are distinct entities. Finally, large-scale IoT systems require coordination among many devices, but traditional centralized control cannot scale and is vulnerable to single points of failure. Swarm intelligence, inspired by the collective behavior of insects and flocks, offers a decentralized way to achieve global objectives through local interactions [6]. Techniques such as ant colony optimization or particle swarm optimization have proved effective for routing, clustering, and resource allocation in dynamic networks. Together, these motivations underpin the design of an edge-reinforced learning platform that is secure, adaptive, and inherently distributed.

1.2. Problem Statement

Despite advances in edge computing and secure communication, there is still no unified framework that can simultaneously guarantee ultra-low latency, robust cross-device coordination, and strong end-to-end data confidentiality in large-scale IoT deployments. Most existing architectures face a number of tensions. Systems optimized for low latency often simplify or bypass encryption, exposing data at intermediate processing points. Security-focused designs, on the other hand, may introduce heavy cryptographic overheads that negate the benefits of edge processing [7].

 doi:10.20944/preprints202601.0513.v1

Similarly, architectures with centralized controllers can enforce globally optimal policies but do not scale well and are prone to failures, while fully distributed schemes often rely on simple heuristics that cannot adapt effectively to complex and evolving conditions.

The specific problem addressed in this work is how to design and implement an edge-centric learning platform that can minimize end-to-end delay for sensing and control traffic, enable reliable cross-device communication, and ensure that sensitive data remain confidential even when processed or aggregated by untrusted edge infrastructure [8]. This involves several sub-problems how to formulate edge decision-making as reinforcement learning tasks under resource constraints how to incorporate homomorphic encryption in a way that makes encrypted data usable for learning and control without overwhelming devices and how to embed swarm intelligence mechanisms so that devices and edge nodes coordinate their behavior through local interactions, yet collectively approximate globally desirable behavior. The solution must function under realistic assumptions, such as intermittent connectivity, mobility, heterogeneous hardware, and potential adversarial behavior.

1.3. Research Objectives

The objective of the research is to build and evaluate an integrated platform that leverages edge reinforcement learning, homomorphic encryption, and swarm intelligence to meet the stringent requirements of modern IoT applications. This objective can be broken down into several concrete goals. The first goal is to develop edge-side reinforcement learning mechanisms that dynamically control sensing frequency, routing, and computation offloading, with the explicit aim of minimizing latency and jitter while respecting constraints on energy consumption and bandwidth [9]. These mechanisms should be modular enough to be adapted to different IoT scenarios, such as industrial monitoring, vehicular networks, or smart buildings.

The second goal is to design a homomorphic encryption layer that is practical for IoT environments. This includes selecting or customizing encryption schemes that support the necessary operations (such as addition or limited multiplication) required by aggregation and learning algorithms, while keeping computational and communication overhead within acceptable bounds for resource-constrained devices. The third goal is to incorporate swarm intelligence techniques into the coordination layer so that devices and edges can form and maintain communication paths, balance load, and adjust to topology changes without centralized control [10]. Together, these goals support a final objective: to demonstrate, through simulation and testbed experiments, that the integrated platform can achieve lower latency and better robustness than conventional designs, without compromising privacy.

1.4. Contributions and Paper Organization

This work makes several contributions to the design and analysis of intelligent, secure edge-based IoT systems. First, it introduces an edge-reinforced learning framework in which edge nodes act as adaptive agents that continuously refine their policies for sensing, routing, and offloading, based on local observations and reward signals tied to end-to-end latency and reliability. This framework shows how reinforcement learning can be embedded into the fabric of an IoT network in a way that is compatible with heterogeneous devices and variable traffic patterns [11]. Second, it proposes a homomorphic encryption aware processing pipeline that enables edge nodes to operate on encrypted data streams for tasks such as aggregation and policy evaluation, reducing the exposure of sensitive information while keeping computational costs manageable.

Third, the paper presents a swarm intelligence–based coordination layer that governs cross-device communication, using bio-inspired mechanisms to establish low-latency paths, distribute tasks, and recover from congestion or failures. This layer interacts with the learning agents at the edge, allowing global behavior to emerge from local decisions in a controlled manner. Finally, the work provides a comprehensive evaluation on realistic IoT scenarios, quantifying latency, communication overhead, and security properties, and comparing the proposed platform to cloud-

 doi:10.20944/preprints202601.0513.v1

centric and traditional edge baselines [12]. The remainder of the paper is organized as follows: the next section surveys related work in edge computing, secure IoT processing, and swarm-based coordination; subsequent sections introduce the system model and problem formulation, describe the proposed architecture and algorithms in detail, present the experimental setup and performance results, discuss practical implications and limitations, and conclude with directions for future enhancements.

2. Related Work

2.1. Edge Computing and IoT Sensing Architectures

Early IoT architectures were predominantly cloud-centric, with devices acting as simple data producers and the cloud handling all analytics and decision-making. This model quickly ran into limitations as application domains such as industrial control, autonomous transport, and telesurgery demanded sub-second or even millisecond-scale response times. To bridge this gap, edge and fog computing paradigms emerged, inserting intermediate layers of computation between devices and the cloud. Edge gateways and micro data centres began to host data preprocessing, filtering, and local control logic, reducing the need to transmit raw streams over wide-area networks [13]. Numerous frameworks have been proposed that organize sensing devices into clusters managed by nearby edge nodes, which perform aggregation, anomaly detection, or control decisions on behalf of local groups.

Despite these advances, most edge computing architectures still treat control logic as relatively static or only slowly reconfigurable. Thresholds, routing priorities, and offloading policies are often tuned manually or based on offline profiling, which limits their ability to react to unpredictable changes in workload and network conditions. Furthermore, many designs continue to assume that edge nodes are trusted entities that can freely decrypt and inspect device data [14]. As a result, while latency is improved relative to cloud-only models, privacy risks and rigidity remain pressing concerns. These limitations motivate architectures where the edge is not only a computational relay but also an intelligent, adaptive controller that can operate effectively even when it cannot see raw data in plaintext.

2.2. Reinforcement Learning for Resource-Constrained Devices

Reinforcement learning has attracted considerable attention as a means of enabling autonomous adaptation in networks and cyber-physical systems. In the context of IoT and edge computing, researchers have applied RL to problems such as dynamic task offloading, energy-aware duty cycling, and congestion control. Typical formulations cast the edge node or device as an agent that observes local states such as queue lengths, channel quality, and battery level and selects actions like adjusting transmission power, changing routes, or deciding whether to offload computation to a nearby server [15]. Rewards are designed to capture latency, throughput, or energy consumption, allowing the agent to learn policies that balance competing objectives over time.

However, applying RL directly on resource-constrained devices raises several challenges. Many RL algorithms, particularly deep RL, require non-trivial computational resources and memory footprints, which may exceed the capabilities of low-power sensors and microcontrollers. This has led to work on lightweight RL variants, model compression, and offloading the training phase to more capable edge or cloud servers while executing only inference on devices. Another complication is the need for fast convergence IoT environments are highly dynamic, so policies must adapt quickly enough to remain relevant. Existing studies often focus on a single dimension, such as offloading or power control, and assume clear access to state information, including potentially sensitive metrics [16]. What remains less explored is a holistic RL framework embedded in the edge infrastructure that can coordinate multiple decisions sensing, routing, and offloading while operating under strict privacy constraints and in concert with other distributed intelligence mechanisms.

 doi:10.20944/preprints202601.0513.v1

2.3. Homomorphic Encryption in IoT and Edge Security

Homomorphic encryption has been studied as a promising approach for privacy-preserving computation in untrusted environments. In cloud and edge contexts, it allows servers to perform operations such as summation, averaging, or even limited forms of machine learning on encrypted data, with only the data owner able to decrypt the final result. Several works have proposed using partially or somewhat homomorphic schemes to secure IoT data aggregation, for example enabling gateways to compute encrypted sums of sensor readings for monitoring or billing purposes without accessing individual values [18]. Some research has extended this to privacy-preserving model training, where gradients or model updates are homomorphically aggregated across devices.

Despite its potential, practical deployment in IoT scenarios remains challenging because homomorphic operations typically incur higher computational and communication overhead than conventional cryptography. Resource-constrained devices may struggle to perform frequent encryptions of complex ciphertexts, and edge servers must handle the processing burden of homomorphic arithmetic while still meeting latency targets [19]. Much of the existing work therefore focuses on narrow tasks, such as simple aggregation or linear operations, or relies on batching and offline processing that may not suit real-time control. Moreover, integration of homomorphic encryption with adaptive control or learning logic at the edge is still limited; encryption is often treated as a separate security layer rather than an integral part of the decision-making pipeline. There is thus room for platforms that carefully co-design learning algorithms and encryption schemes to preserve privacy without undermining responsiveness.

2.4. Swarm Intelligence for Distributed Optimization and Routing

Swarm intelligence techniques, inspired by collective behaviours in nature, have been widely explored for network optimization, particularly in routing and clustering. Algorithms such as ant colony optimization model routing decisions as the laying and evaporation of virtual pheromones, where frequently used and high-quality paths accumulate stronger pheromone trails, leading future packets or agents to prefer them [20]. Particle swarm optimization, on the other hand, treats potential solutions as particles moving through a search space, influenced by their own best experiences and those of their neighbours. In wireless sensor networks and ad hoc networks, these methods have been used to find energy-efficient routes, balance load among nodes, and form stable clusters under mobility.

These approaches are attractive in IoT environments because they rely on local interactions and simple update rules, rather than global knowledge or heavy computation. They can naturally adapt to topological changes, such as node failures or mobility, and do not require centralized control planes. Nonetheless, many swarm-based network protocols have been validated under simplified assumptions and may not fully account for modern edge environments with heterogeneous devices, multi-hop backhaul, and tight latency constraints [21]. Furthermore, swarm intelligence is often applied as a stand-alone routing or clustering heuristic and is rarely integrated with learning-based control or cryptographic protection of the underlying data. There is an opportunity to elevate swarm mechanisms from pure routing tools to a broader coordination layer that interacts with edge learning and security policies.

2.5. Summary of Gaps and Research Opportunities

The prior literature reveals strong foundations in edge computing, reinforcement learning, homomorphic encryption, and swarm intelligence, but also clear gaps when these strands are considered together. Edge architectures have improved latency but frequently assume trusted edge nodes and static policies. Reinforcement learning work has demonstrated the benefits of adaptive control but tends to treat security as an external concern and often focuses on single-function optimization [22]. Homomorphic encryption research has shown how to protect data during aggregation and computation, yet many solutions are tailored to offline analytics or narrow

 doi:10.20944/preprints202601.0513.v1

operations, making them hard to adopt in latency-sensitive, continuously adapting systems. Swarm-based protocols offer scalable, decentralized coordination, but are usually applied in isolation from learning and security mechanisms, and may not be tuned for ultra-low latency industrial or mission-critical IoT scenarios.

These gaps suggest several research opportunities that the proposed platform aims to address. One is the co-design of reinforcement learning and homomorphic encryption so that edge agents can learn from and act on encrypted information without violating latency and resource constraints. Another is the integration of swarm intelligence not merely as a routing heuristic, but as a first-class coordination mechanism that shapes how devices and edge nodes share information, balance load, and collectively pursue latency and reliability objectives. A further opportunity lies in building a unified framework that explicitly targets ultra-low latency IoT sensing and cross-device communication, rather than treating latency, privacy, and coordination as separate optimization problems [23]. By tackling these open issues in a single architecture, the proposed edge-reinforced learning platform contributes a novel, holistic approach that moves beyond the limitations of existing, more siloed solutions.

3. System Model and Problem Formulation

3.1. Network and IoT Sensing Model

We consider a heterogeneous IoT network with a set of sensing devices $\mathcal{N} = \{1,2,\dots,N\}$deployed over a geographic area, connected through a wireless multihop graph $G = (\mathcal{N} \cup \mathcal{E}, \mathcal{L})$, where $\mathcal{E}$is the set of edge nodes and $\mathcal{L}$ the set of wireless links [24]. Each device $i \in \mathcal{N}$observes a physical process and generates measurements $x_i(t)$over time. The sensing process can be modeled as a point process with adaptive rate $\lambda_i(t)$, so that the expected number of samples over an interval $[0, T]$ is

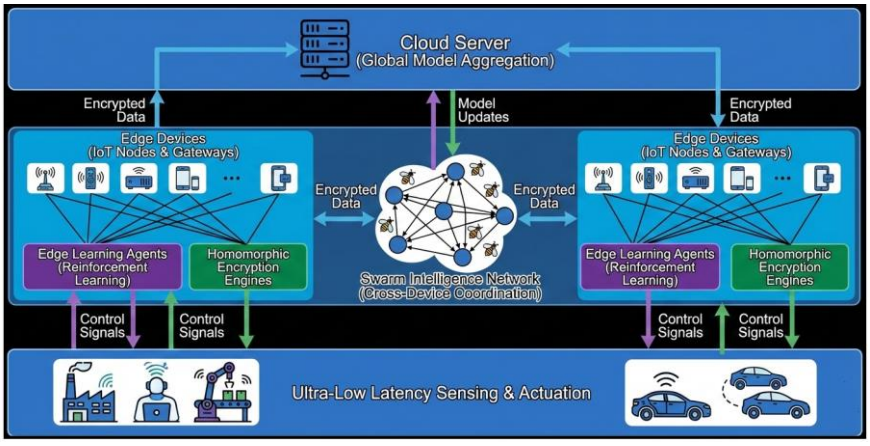
$$\mathbb{E}[N_i(T)] = \int_0^T \lambda_i(t)\, dt \tag{1}$$

The rate $\lambda_i(t)$is not fixed it is controlled by the edge-reinforced learning policy to balance information freshness, latency, and energy [25].

Each measurement belonging to flow $f \in \mathcal{F}$ has a destination set $\mathcal{D}_f \subseteq \mathcal{E} \cup \{\text{cloud,peer devices}\}$ and a deadline D_f. For a packet $p$of flow f, generated at time t_p^{gen}, the end-to-end delay is

$$d_p = t_p^{\text{recv}} - t_p^{\text{gen}} \tag{2}$$

where $t_p^{\text{recv}}$is the time, the packet reaches the consumer. A packet is considered timely if $d_p \le D_f$. The sensing layer thus produces a set of time-stamped packets with heterogeneous deadlines and importance weights, which must be routed and processed under the constraints defined below [26].



 doi:10.20944/preprints202601.0513.v1

Figure 1. Edge Reinforced Learning Platform with Homomorphic Encryption and Swarm Intelligence.

3.2. Edge Device Computational and Communication Model

The edge device fabric consists of edge nodes $\mathcal{E} = \{1,2,\dots,E\}$ with higher computational capacity and storage, serving nearby devices over wireless links. Each device i has a local CPU with maximum processing rate C_i^{dev} (cycles per second) and energy budget E_i^{bat}, while each edge node $e \in \mathcal{E}$ has capacity $C_e^{\text{edge}} \gg C_i^{\text{dev}}$. A task j generated by device i requires c_{ij} CPU cycles and data size s_{ij} bits [27]. If processed locally, the expected processing delay is

$$T_{ij}^{\text{loc}} = \frac{c_{ij}}{C_i^{\text{dev}}} \tag{3}$$

If offloaded to an edge node e, the total delay becomes

$$T_{ij}^{\text{edge}} = T_{ij}^{\text{uplink}} + \frac{c_{ij}}{C_e^{\text{edge}}} + T_{ij}^{\text{downlink}} \tag{4}$$

where uplink/downlink delays include transmission and queuing components [28].

Communication between node u and v over link $(u,v) \in \mathcal{L}$ has an achievable rate $R_{uv}(t)$ and packet error probability $p_{uv}^{\text{err}}(t)$ depending on channel conditions. For a packet of size s_p, the nominal transmission time over link (u, v) is

$$T_{p,uv}^{\text{tx}} = \frac{s_p}{R_{uv}(t)} \tag{5}$$

Along a multi-hop route $\pi_p = (i,\dots,e)$, the network delay is

$$T_p^{\text{net}} = \sum_{(u,v)\in\pi_p} \left(T_{p,uv}^{\text{tx}} + T_{p,uv}^{\text{queue}}\right) \tag{6}$$

where $T_{p,uv}^{\text{queue}}$ denotes the queuing delay on each hop [29]. Edge and device processing are modelled as M/M/1 queues where the utilization factor is $\rho_k = \lambda_k/\mu_k$ for node k, with stability constraint $\rho_k < 1$.

3.3. Threat Model and Security Assumptions

The threat model assumes an honest-but-curious edge and cloud infrastructure, and potentially compromised intermediate nodes. An adversary can eavesdrop on links, observe or capture ciphertexts, and control a subset $\mathcal{E}_{\text{adv}} \subseteq \mathcal{E}$ of edge nodes [30]. Devices and trusted application backends share public–private key pairs, and use an additively homomorphic encryption scheme $\text{Enc}(\cdot)$, $\text{Dec}(\cdot)$ with operation

$$\text{Enc}(m_1) \oplus \text{Enc}(m_2) = \text{Enc}(m_1 + m_2) \tag{7}$$

which allows aggregation of encrypted readings.

A simple encrypted aggregation at edge node e over measurements x_i from devices $i \in \mathcal{S}_e$ yields

$$C_e = \bigoplus_{i\in\mathcal{S}_e} \text{Enc}(x_i) = \text{Enc}\left(\sum_{i\in\mathcal{S}_e} x_i\right) \tag{8}$$

The edge node sees only C_e, not the individual x_i. The adversary's advantage in distinguishing two equal-length measurement sets, under chosen-plaintext attacks, is assumed negligible according to the semantic security of the scheme [33]. We assume key management (distribution and rotation of keys) is handled by a secure bootstrap mechanism. Denial-of-service and physical tampering are acknowledged but treated as outside the primary scope; the focus is confidentiality and integrity of sensed data and learned policies under computational attacks.

3.4. Latency, Reliability, and Energy Constraints

For each packet p, the end-to-end latency can be decomposed as

 doi:10.20944/preprints202601.0513.v1

$$d_p = T_p^{\text{sense}} + T_p^{\text{proc,dev}} + T_p^{\text{net}} + T_p^{\text{proc,edge}} \quad (9)$$

where $T_p^{\text{sense}}$is the sensing delay (time between event and sampling), $T_p^{\text{proc,dev}}$and $T_p^{\text{proc,edge}}$are processing delays at device and edge, and $T_p^{\text{net}}$is as above [34]. For a flow $f$with deadline D_f, the latency constraint can be expressed as

$$\Pr(d_p \le D_f) \ge \eta_f \quad (10)$$

where $\eta_f$is the target reliability (for example, $\eta_f = 0.99$for safety-critical traffic) [35].

Energy consumption at device $i$accumulates sensing, computation, and communication costs. If P_i^{sense}, P_i^{cpu}, and P_i^{tx}denote power for sensing, computation, and transmission, the energy over a horizon $[0, T]$is approximately

$$E_i(T) = \int_0^T \left(P_i^{\text{sense}}(t) + P_i^{\text{cpu}}(t) + P_i^{\text{tx}}(t)\right) dt \quad (11)$$

Devices must satisfy an energy budget $E_i(T) \le E_i^{\max}$. In discrete-time operation, per-slot energy $e_i(k)$leads to

$$\sum_{k=0}^{K} e_i(k) \le E_i^{\max} \quad (12)$$

Reliability constraints can also be written in terms of packet loss probability $P_f^{\text{loss}}$for flow f:

$$P_f^{\text{loss}} \le \delta_f \quad (13)$$

where $\delta_f$is the maximum acceptable loss rate [36]. These constraints jointly shape the feasible action space for the learning and swarm coordination mechanisms.

3.5. Formal Problem Definition

At the edge, decision-making is cast as a sequential control problem modelled as a Markov decision process (MDP) or partially observable MDP [37]. For an edge node e, at decision epoch t, the local state is

$$s_e(t) = (\mathbf{q}_e(t), \mathbf{h}_e(t), \boldsymbol{\lambda}(t), \mathbf{E}(t)) \quad (14)$$

where $\mathbf{q}_e(t)$ denotes queue lengths and processing loads, $\mathbf{h}_e(t)$ captures link qualities to neighboring devices and edges, $\boldsymbol{\lambda}(t)$summarizes current sensing rates, and $\mathbf{E}(t)$tracks residual energies at associated devices [38]. The action $a_e(t)$includes decisions such as updated sensing rates $\lambda_i(t+1)$, routing choices $r_{i\to j}(t)$, and offloading decisions for tasks.

A policy $\pi_e$maps states to actions, $\pi_e: s_e(t) \mapsto a_e(t)$. The instantaneous cost for edge $e$can be defined as

$$c_e(t) = \alpha\, \bar{d}_e(t) + \beta\, \bar{E}_e(t) + \gamma\, \bar{L}_e(t) \quad (15)$$

where $\bar{d}_e(t)$is the average delay of packets handled by e, $\bar{E}_e(t)$the average energy consumption of associated devices, and $\bar{L}_e(t)$a penalty for packet loss or deadline violations [39]. Scalars $\alpha, \beta, \gamma \ge 0$weight these objectives. The long-term objective is to find a joint policy $\Pi = \{\pi_e\}_{e\in\mathcal{E}}$that minimizes the expected discounted cumulative cost:

$$\min_{\Pi} \; J(\Pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma_d^t \sum_{e\in\mathcal{E}} c_e(t)\right] \quad (16)$$

subject to the constraints

$$\Pr(d_p \le D_f) \ge \eta_f, P_f^{\text{loss}} \le \delta_f, E_i(T) \le E_i^{\max} \quad (17)$$

and the homomorphic-encryption feasibility, which restricts operations on raw data to additions and limited multiplications on ciphertexts [40].

 doi:10.20944/preprints202601.0513.v1

Swarm intelligence appears as a distributed optimization layer over the network graph. For example, in an ant-colony–like routing scheme, each link (u, v) maintains a pheromone level $\tau_{uv}(t)$. At each step, route selection probabilities are

$$P_{uv}(t) = \frac{(\tau_{uv}(t))^{\alpha_s}(\eta_{uv}(t))^{\beta_s}}{\sum_{w\in\mathcal{N}_u}(\tau_{uw}(t))^{\alpha_s}(\eta_{uw}(t))^{\beta_s}} \quad (18)$$

where $\eta_{uv}(t)$is a heuristic desirability (for example, inverse of estimated delay), and α_s, β_stune the influence of pheromone versus heuristic [41]. Pheromone updates use

$$\tau_{uv}(t+1) = (1-\rho)\tau_{uv}(t) + \Delta\tau_{uv}(t) \quad (19)$$

with evaporation rate $\rho \in (0,1)$ and reinforcement term $\Delta\tau_{uv}(t)$ based on observed path performance. These swarm dynamics interact with the reinforcement learning policies at edge nodes, effectively shaping the transition probabilities of the MDP through evolved routing preferences [42].

4. Proposed Edge-Reinforced Learning Platform

At a high level, the platform follows a layered architecture comprising three main tiers: IoT devices at the bottom, edge nodes in the middle, and an optional cloud layer at the top. IoT devices are responsible for sensing physical phenomena, performing lightweight preprocessing, and encrypting their measurements using a homomorphic encryption scheme before transmission. Each device is associated with one or more nearby edge nodes, which act as local controllers and coordination hubs [43]. These edge nodes run reinforcement learning agents that make decisions about sensing rates, routing preferences, and offloading strategies based on locally observed states and feedback from the network. The cloud layer, if present, performs long-term analytics, global policy refinement, and archival storage, but is not involved in the tight control loops that must meet strict latency deadlines.

4.1. Overall System Architecture

The platform follows a three-tier architecture comprising IoT devices, edge nodes, and an optional cloud layer. IoT devices $i \in \mathcal{N}$ generate measurements $x_i(t)$and apply homomorphic encryption before transmission, producing ciphertexts

$$c_i(t) = \mathrm{Enc}(x_i(t)) \quad (20)$$

Each device associates with at least one edge node $e \in \mathcal{E}$, forming local clusters. The logical topology can be represented as a bipartite graph between devices and edges, and an overlay graph among edge nodes themselves [44]. Edge nodes implement reinforcement learning (RL) agents that choose control actions $a_e(t)$based on observed state $s_e(t)$, according to a policy $\pi_e$such that

$$a_e(t) = \pi_e(s_e(t); \theta_e) \quad (21)$$

where $\theta_e$are the policy parameters (for example, weights of a neural network).

A crypto module at each edge supports additively homomorphic operations, allowing encrypted aggregation of sensed data:

$$C_e(t) = \bigoplus_{i\in\mathcal{S}_e} c_i(t) = \bigoplus_{i\in\mathcal{S}_e} \mathrm{Enc}(x_i(t)) = \mathrm{Enc}\left(\sum_{i\in\mathcal{S}_e} x_i(t)\right) \quad (22)$$

where $\mathcal{S}_e$is the set of devices served by edge e. A swarm coordination module maintains per-link metrics, such as pheromone levels $\tau_{uv}(t)$on links (u, v), that bias routing and task allocation [45]. The cloud, when present, operates on long-term aggregates and may periodically refine global hyperparameters (e.g., reward weights, exploration rates), but time-critical loops remain confined to the device–edge tier.

4.2. Edge-Centric Reinforcement Learning Framework

Each edge node is modelled as an RL agent interacting with a local environment that evolves according to a controlled stochastic process [46]. At discrete decision epochs $t = 0,1,2, \ldots$, edge

 doi:10.20944/preprints202601.0513.v1

e observes a state $s_e(t)$, selects an action $a_e(t)$, and receives a scalar reward $r_e(t)$. The environment transitions according to

$$\Pr\left(s_e(t+1) \mid s_e(t), a_e(t)\right) \tag{23}$$

which is shaped by traffic patterns, wireless conditions, device behavior, and swarm routing dynamics.

In a value-based scheme such as deep Q-learning, the edge maintains an action-value function

$$Q_e(s,a) \approx \mathbb{E}[\, R_e \mid s_e(t) = s, a_e(t) = a \,] \tag{24}$$

where the discounted return is

$$R_e = \sum_{k=0}^{\infty} \gamma^k \, r_e(t+k+1), 0 < \gamma < 1 \tag{25}$$

The Q-values are parameterized by ϕ_e, and updated using temporal-difference learning

$$Q_e(s_t, a_t) \leftarrow Q_e(s_t, a_t) + \alpha(r_t + \gamma \max_{a'} Q_e(s_{t+1}, a') - Q_e(s_t, a_t)) \tag{26}$$

with learning rate α. In actor–critic form, edge $e$maintains a policy (actor) $\pi_{\theta_e}(a \mid s)$and a value function (critic) $V_{\omega_e}(s)$, and updates parameters via gradient steps:

$$\theta_e \leftarrow \theta_e + \eta_\theta \, \nabla_{\theta_e} \log \, \pi_{\theta_e}(a_t \mid s_t) \, \delta_t \tag{27}$$

$$\omega_e \leftarrow \omega_e - \eta_\omega \, \nabla_{\omega_e}(\delta_t^2) \tag{28}$$

where the temporal-difference error is

$$\delta_t = r_t + \gamma V_{\omega_e}(s_{t+1}) - V_{\omega_e}(s_t) \tag{29}$$

Because payloads are encrypted, the RL agent relies on observable performance metrics (delays, losses, queue lengths) and metadata rather than raw $x_i(t)$ [47]. Training may follow an online scheme, with mini-batches drawn from a replay buffer $\mathcal{D}_e = \{(s, a, r, s')\}$, and loss

$$\mathcal{L}_e(\phi_e) = \mathbb{E}_{(s,a,r,s') \sim \mathcal{D}_e}[(y - Q_{\phi_e}(s,a))^2] \tag{30}$$

where

$$y = r + \gamma \max_{a'} Q_{\phi_e^-}(s', a') \tag{31}$$

and ϕ_e^- are target network parameters.

4.3. State, Action, and Reward Design

The state vector $s_e(t)$aggregates local information into a finite-dimensional representation suitable for learning [48]. A typical design is

$$s_e(t) = [\mathbf{q}_e(t), \mathbf{h}_e(t), \boldsymbol{\lambda}_e(t), \mathbf{E}_e(t), \boldsymbol{\ell}_e(t)] \tag{32}$$

where

- $\mathbf{q}_e(t)$: queue lengths per traffic class (e.g., control, monitoring),
- $\mathbf{h}_e(t)$: link quality estimates to neighbors (e.g., moving average of packet loss or effective rate),
- $\boldsymbol{\lambda}_e(t)$: current sensing rates for associated devices,
- $\mathbf{E}_e(t)$: normalized residual energies of devices,
- $\boldsymbol{\ell}_e(t)$: recent latency statistics (e.g., mean and variance of $d_p$for flows terminating at or via e).

The action $a_e(t)$can be represented as a vector of control variables:

$$a_e(t) = [\Delta\boldsymbol{\lambda}_e(t), \mathbf{w}_e(t), \mathbf{o}_e(t)] \tag{33}$$

where

- $\Delta\boldsymbol{\lambda}_e(t)$: increments or decrements to device sampling rates within bounds $[\lambda_i^{\min}, \lambda_i^{\max}]$,

 doi:10.20944/preprints202601.0513.v1

- $\mathbf{w}_e(t)$: routing weights or probabilities for choosing next hops; for a flow f at edge e,

$$P_{e\to v}^{f}(t) = \frac{(\tau_{ev}(t))^{\alpha_s}(\eta_{ev}(t))^{\beta_s}}{\sum_{u\in\mathcal{N}_e}(\tau_{eu}(t))^{\alpha_s}(\eta_{eu}(t))^{\beta_s}} \quad (34)$$

  using pheromone $\tau_{ev}(t)$ and heuristic desirability $\eta_{ev}(t)$,
- $\mathbf{o}_e(t)$: binary or fractional offloading decisions, where $o_{ij}(t) = 1$ means task j from device i is offloaded to edge e, and $o_{ij}(t) = 0$ means local processing.

The reward $r_e(t)$ is designed to penalize delay, deadline violations, and energy use, while rewarding reliability [49]. A common form is

$$r_e(t) = -(\alpha_d\, \bar{d}_e(t) + \alpha_m\, M_e(t) + \alpha_E\, \bar{E}_e(t)) \quad (35)$$

where

- $\bar{d}_e(t)$: average end-to-end delay of packets handled by e during interval t,
- $M_e(t)$: number (or fraction) of packets that miss deadlines or are dropped,
- $\bar{E}_e(t)$: average energy consumption of associated devices,
- $\alpha_d, \alpha_m, \alpha_E \geq 0$: weighting coefficients.

Optionally, reliability and fairness terms can be included:

$$r_e(t) = -\alpha_d \bar{d}_e(t) - \alpha_m M_e(t) - \alpha_E \bar{E}_e(t) + \alpha_R R_e^{\text{succ}}(t) - \alpha_F \Delta DP_e(t) \quad (36)$$

where $R_e^{\text{succ}}(t)$ is the fraction of successfully delivered packets, and $\Delta DP_e(t)$ measures disparity among flows (e.g., difference between best and worst flow success rates).

4.4. Policy Update and Coordination Among Edge Nodes

Each edge node updates its policy parameters θ_e (and possibly value parameters ω_e or Q-parameters ϕ_e) based on collected experience [50]. In a distributed actor–critic setting, the update rules at edge e are

$$\theta_e \leftarrow \theta_e + \eta_\theta\, \mathbb{E}[\nabla_{\theta_e} \log\, \pi_{\theta_e}(a_t \mid s_t)\, \delta_t] \quad (37)$$

$$\omega_e \leftarrow \omega_e - \eta_\omega\, \mathbb{E}[\nabla_{\omega_e} \delta_t^2] \quad (38)$$

with

$$\delta_t = r_t + \gamma V_{\omega_e}(s_{t+1}) - V_{\omega_e}(s_t) \quad (39)$$

In practice, these expectations are approximated using mini-batches from a replay buffer.

Coordination among edges is achieved by periodically exchanging summarized information [51]. One simple scheme is parameter averaging over a neighbourhood $\mathcal{N}_e$:

$$\theta_e \leftarrow (1-\beta)\, \theta_e + \beta\, \frac{1}{|\mathcal{N}_e|} \sum_{k\in\mathcal{N}_e} \theta_k \quad (40)$$

where $0 \leq \beta \leq 1$ controls the strength of consensus. Alternatively, a federated learning-like update can be used where a cloud or super-edge aggregates local gradients $\nabla_{\theta_e}\mathcal{L}_e$ and broadcasts updated parameters [52]. To protect privacy, only encrypted or differentially private summaries may be shared; for example, each edge could transmit

$$\tilde{g}_e = g_e + \mathcal{N}(0, \sigma^2 I) \quad (41)$$

where g_e is its gradient estimate and $\mathcal{N}(0, \sigma^2 I)$ is Gaussian noise calibrated to a target privacy budget.

In parallel, swarm-based coordination maintains link pheromone levels $\tau_{uv}(t)$ that reflect recent path performance [53]. After a path π is used and its end-to-end delay $\hat{d}_\pi$ is observed, pheromones on its links are updated as

$$\tau_{uv}(t+1) = (1-\rho)\, \tau_{uv}(t) + \Delta\tau_{uv}(t)$$

 doi:10.20944/preprints202601.0513.v1

$$\Delta\tau_{uv}(t) = \begin{cases} \frac{Q}{\hat{d}_\pi}, & \text{if } (u,v) \in \pi, \\ 0, & \text{otherwise,} \end{cases} \tag{42}$$

where $0 < \rho < 1$is the evaporation rate and $Q$is a scaling constant. Shorter-delay paths receive larger pheromone reinforcement, biasing future routing choices toward low-latency routes. The RL policy indirectly affects $\hat{d}_\pi$by controlling load and offloading, while swarm updates reshape the transition probabilities in the MDP by changing routing behavior [54]. This bidirectional coupling between RL and swarm mechanisms enables the network to converge toward configurations that jointly minimize latency and energy while satisfying reliability and privacy constraints.

5. Homomorphic Encryption-Based Security Mechanism

5.1. Choice of Homomorphic Encryption Scheme

For the considered IoT and edge setting, the platform adopts a partially (additively) homomorphic encryption scheme to balance security and efficiency. Let $pk$and skdenote the public and private keys [55]. A plaintext measurement $m \in \mathbb{Z}_n$ is encrypted as a ciphertext $c = \mathrm{Enc}_{pk}(m)$, and decrypted as $m = \mathrm{Dec}_{sk}(c)$. The crucial homomorphic property is

$$\mathrm{Dec}_{sk}\left(\mathrm{Enc}_{pk}(m_1) \otimes \mathrm{Enc}_{pk}(m_2)\right) = m_1 + m_2 (\mathrm{mod\ n})$$

where $\otimes$denotes the ciphertext-domain operation (typically multiplication or group operation).

In an additively homomorphic scheme such as Paillier, encryption of a message $m$under modulus $n = pq$(with large primes p, q) and generator $g$can be expressed as

$$\mathrm{Enc}_{pk}(m) = g^m r^n \, mod \, n^2 \tag{43}$$

and additive homomorphism is realized as

$$\mathrm{Enc}_{pk}(m_1) \cdot \mathrm{Enc}_{pk}(m_2) mod n^2 = \mathrm{Enc}_{pk}(m_1 + m_2 mod n) \tag{44}$$

This property allows edge nodes to aggregate encrypted readings without access to plaintext, which is well-suited for secure IoT aggregation and low-depth learning-related operations.

For scenarios involving approximate real-valued operations (e.g., model scores), approximate HE schemes such as CKKS support addition and multiplication over encrypted vectors with bounded numerical error, but at the cost of more complex parameter management and bootstrapping [57]. In this platform, integer additively homomorphic schemes are used for high-frequency sensing paths, while CKKS-type schemes may be reserved for less frequent, higher-level model computations.

5.2. Secure Sensing Data Acquisition and Aggregation

Each IoT device $i$samples a measurement $x_i(t)$at time $t$and encrypts it locally, producing

$$c_i(t) = \mathrm{Enc}_{pk}(x_i(t)) \tag{45}$$

The device transmits $c_i(t)$to its associated edge node $e$over an authenticated channel [58]. For a set of devices $\mathcal{S}_e$attached to edge e, the edge performs encrypted aggregation

$$C_e(t) = \bigotimes_{i \in \mathcal{S}_e} c_i(t) \tag{46}$$

which, by the additive homomorphism, satisfies

$$\mathrm{Dec}_{sk}(C_e(t)) = \sum_{i \in \mathcal{S}_e} x_i\,(t)\ (mod\ n) \tag{47}$$

If the application requires an encrypted average, the edge can compute

 doi:10.20944/preprints202601.0513.v1

$$\tilde{C}_e(t) = C_e(t) \otimes \mathrm{Enc}_{pk}(|\mathcal{S}_e|^{-1}) \quad (48)$$

or, more efficiently, the decryption-side application divides the decrypted sum by $|\mathcal{S}_e|$ in plaintext [59]. Scalar multiplication by a known constant k is similarly achieved as

$$c_i^{(k)}(t) = c_i(t)^k \Rightarrow \mathrm{Dec}_{sk}(c_i^{(k)}(t)) = kx_i(t) \quad (49)$$

From the edge's perspective, only ciphertexts $c_i(t)$ and aggregated ciphertexts $C_e(t)$ are visible [60]. The RL agent and swarm modules operate on metadata such as timestamps, packet sizes, and arrival statistics, and on aggregate performance indicators (e.g., decrypted at a trusted backend), but never on individual plaintext measurements at the edge. This preserves confidentiality even if an edge node is compromised, while still enabling secure aggregation and rate control tailored to application needs.

5.3. Encrypted Model Update and Policy Evaluation

To integrate learning with encryption, the platform leverages the additive homomorphism to support privacy-preserving model update aggregation and simple encrypted policy evaluation [61]. Suppose each device or local controller computes a gradient or update component g_i based on its local observations. Rather than sending g_i in plaintext, it transmits

$$\hat{g}_i = \mathrm{Enc}_{pk}(g_i) \quad (50)$$

The edge aggregates encrypted updates as

$$G_e = \bigotimes_{i \in \mathcal{S}_e} \hat{g}_i = \mathrm{Enc}_{pk}\left(\sum_{i \in \mathcal{S}_e} g_i\right) \quad (51)$$

which is then forwarded to a trusted model owner (or secure enclave) for decryption and parameter update

$$\theta^{\mathrm{new}} = \theta^{\mathrm{old}} - \eta \sum_{i \in \mathcal{S}_e} g_i \quad (52)$$

This realizes a privacy-preserving, federated-style update process where the edge acts only as an encrypted aggregator.

For certain low-depth policy computations, the edge can evaluate simple linear functions on ciphertexts [62]. Consider a scalar decision score of the form

$$y = wx + b \quad (53)$$

with w, b known at the edge and x encrypted. Using additive homomorphism and scalar multiplication, the edge computes

$$c_y = \mathrm{Enc}_{pk}(x)^w \otimes \mathrm{Enc}_{pk}(b) = \mathrm{Enc}_{pk}(wx + b) \quad (54)$$

and forwards c_y to a trusted decryptor, which recovers $y = \mathrm{Dec}_{sk}(c_y)$. This enables encrypted scoring and thresholding for simple RL or swarm-related signals without exposing raw features [63].

When approximate real-valued operations are required (for example, computing normalized statistics for RL), an approximate HE scheme such as CKKS allows vector operations

$$\mathbf{c}_y = \mathbf{c}_w \odot \mathbf{c}_x \oplus \mathbf{c}_b \quad (55)$$

where $\odot$ and $\oplus$ denote homomorphic multiplication and addition on ciphertext vectors, and decryption yields an approximation $\tilde{\mathbf{y}} \approx \mathbf{y}$ with bounded error [64]. Depth limits and rescaling operations constrain how far such computations can be pushed on encrypted data; thus, in this platform, only low-depth linear or near-linear computations are offloaded to the encrypted domain, while more complex RL updates occur where decryption is allowed or inside hardware enclaves.

 doi:10.20944/preprints202601.0513.v1

5.4. Security and Complexity Analysis

The security of the homomorphic layer relies on the hardness assumptions underlying the chosen scheme, such as factoring for Paillier-type cryptosystems or lattice problems for CKKS-like schemes [65]. Under these assumptions, given ciphertexts $\{\mathrm{Enc}_{pk}(m_i)\}$, an adversary who does not possess sk cannot feasibly recover the plaintexts m_i or distinguish encryptions of different messages with probability significantly better than random guessing. Formally, the scheme satisfies semantic security (IND-CPA) if for any probabilistic polynomial-time adversary $\mathcal{A}$,

$$|\Pr[\mathcal{A}(\mathrm{Enc}_{pk}(m_0)) = 1] - \Pr[\mathcal{A}(\mathrm{Enc}_{pk}(m_1)) = 1]| \leq \varepsilon \tag{56}$$

for any pair of messages m_0, m_1 of equal length, with negligible ε.

Complexity analysis considers both computational and communication overhead. For a Paillier-type scheme, encryption and homomorphic multiplication (corresponding to plaintext addition) involve modular exponentiations modulo n^2, whose complexity is roughly $\mathcal{O}(k^3)$ bit operations for key size k using naïve arithmetic, or lower with optimized big-integer libraries [66]. If each device reports at rate λ_i, the aggregate homomorphic operations per edge per unit time are on the order of

$$C_{\mathrm{HE}}^{\mathrm{edge}} \approx \sum_{i \in \mathcal{S}_e} \lambda_i \; c_{\mathrm{enc}} + \Lambda_e \, c_{\mathrm{agg}} \tag{57}$$

where c_{enc} and c_{agg} denote the cost of a single encryption and ciphertext-domain aggregation, and Λ_e is the number of aggregation windows processed per unit time [67]. Communication overhead arises because ciphertexts are longer than plaintexts; for Paillier, ciphertext size is roughly $2k$ bits versus k-bit plaintexts, leading to an expansion factor of about 2.

The platform limits homomorphic depth to simple additions and scalar multiplications, which avoids expensive bootstrapping operations needed by fully homomorphic schemes and keeps latency overhead manageable for ultra-low-latency flows [68]. Let the end-to-end latency be decomposed as

$$d_p = d_p^{\mathrm{base}} + d_p^{\mathrm{HE}} \tag{58}$$

where d_p^{base} is the latency in a non-encrypted version of the system, and d_p^{HE} captures extra delay from encryption, decryption, and ciphertext operations [69]. The design goal is to maintain

$$\frac{d_p^{\mathrm{HE}}}{d_p^{\mathrm{base}}} \leq \kappa \tag{59}$$

for a small overhead factor κ (for example, 0.2) on critical paths. By confining heavy HE computations to edge or backend nodes and optimizing parameters (key size, batching, windowing), the platform can meet application latency bounds while providing strong confidentiality guarantees for sensed data and model updates in the presence of honest-but-curious or partially compromised infrastructure.

6. Swarm Intelligence for Cross-Device Coordination

6.1. Swarm-Based Topology Control and Task Allocation

In the proposed platform, swarm-based topology control treats devices and edge nodes as agents that adjust their neighbour relations and roles using simple local rules inspired by ant colonies or bird flocks [70]. Each node i maintains a neighborhood set $\mathcal{N}_i(t)$ and a "fitness" score $F_i(t)$ capturing link quality, residual energy, and traffic load; links with persistently low fitness are pruned, while links that improve global connectivity or latency are reinforced over time. The fitness function can be modelled as

$$F_{ij}(t) = \alpha \, \tilde{R}_{ij}(t) - \beta \, \tilde{d}_{ij}(t) - \gamma \, \tilde{q}_j(t) \tag{60}$$

 doi:10.20944/preprints202601.0513.v1

where $\tilde{R}_{ij}(t)$ is normalized link reliability, $\tilde{d}_{ij}(t)$ is normalized one-hop delay, and $\tilde{q}_j(t)$ represents the normalized queue length or processing load at node j parameters α, β, γweight these factors.

Task allocation follows an ant-colony–like mechanism edge nodes broadcast virtual "pheromones" representing their suitability to execute certain tasks (e.g., aggregation, local inference), and devices probabilistically choose execution points based on these pheromone levels and estimated cost [71]. The probability that a device offloads a task to edge node $e$is

$$P_{i\to e}(t) = \frac{(\tau_{ie}(t))^{\alpha_s}(\eta_{ie}(t))^{\beta_s}}{\sum_{k\in\mathcal{E}_i}(\tau_{ik}(t))^{\alpha_s}(\eta_{ik}(t))^{\beta_s}} \tag{61}$$

where $\tau_{ie}(t)$is the pheromone level on the logical link, $\eta_{ie}(t)$is a heuristic desirability term (e.g., inverse estimated latency or energy cost), and α_s, β_scontrol the influence of history versus heuristics [72]. As tasks complete with low delay and acceptable energy, pheromones on the corresponding assignments are reinforced, causing the topology and task distribution to self-organize toward efficient configurations.

6.2. Swarm-Assisted Routing for Ultra-Low Latency Communication

For routing, the platform employs a swarm-assisted scheme similar to ant colony optimization (ACO), where multiple lightweight "ants" explore paths between devices and edge nodes, updating virtual pheromones according to observed path performance [73]. For a path $\pi = (i, \dots, e)$with measured end-to-end delay $\hat{d}_\pi$, pheromones on each link $(u, v) \in \pi$are updated as

$$\tau_{uv}(t+1) = (1-\rho)\,\tau_{uv}(t) + \Delta\tau_{uv}(t) \tag{62}$$

$$\Delta\tau_{uv}(t) = \begin{cases} \frac{Q}{\hat{d}_\pi}, & \text{if } (u,v) \in \pi, \\ 0, & \text{otherwise,} \end{cases} \tag{63}$$

where $\rho \in (0,1)$is the evaporation rate and $Q$is a positive constant controlling reinforcement strength [74]. Shorter-latency paths yield larger pheromone increments, making them more attractive to future packets.

When forwarding a data packet, node $u$selects the next hop $v$according to a stochastic rule that balances exploitation of good paths and exploration of alternatives:

$$P_{uv}(t) = \frac{(\tau_{uv}(t))^{\alpha_s}(\eta_{uv}(t))^{\beta_s}}{\sum_{w\in\mathcal{N}_u}(\tau_{uw}(t))^{\alpha_s}(\eta_{uw}(t))^{\beta_s}} \tag{64}$$

where $\eta_{uv}(t)$can be set to the inverse of recent average delay or energy cost for link (u, v). Because routing is driven by distributed pheromone updates and local observations, the network can quickly reroute traffic around congested or failed links without global recomputation, which is crucial for ultra-low latency in dynamic IoT environments [75].

6.3. Integration of Swarm Decisions with RL Policies

Swarm decisions and edge reinforcement learning policies are tightly coupled so that local learning benefits from global emergent patterns, and vice versa. At each edge node e, the RL state $s_e(t)$ includes swarm-derived features such as average pheromone levels on outgoing links, estimated swarm path reliability, and recent fluctuations in swarm-selected routes. Formally, the state vector can be extended to

$$s_e(t) = [\mathbf{q}_e(t), \mathbf{h}_e(t), \boldsymbol{\lambda}_e(t), \mathbf{E}_e(t), \boldsymbol{\tau}_e(t)] \tag{65}$$

where $\boldsymbol{\tau}_e(t)$collects statistics (e.g., means or histograms) of $\tau_{ev}(t)$for neighbors v. This allows the RL agent to learn how different swarm configurations impact delay and reliability and to choose actions that complement swarm behavior, such as adjusting sampling rates or offloading thresholds when swarm metrics indicate congestion.

 doi:10.20944/preprints202601.0513.v1

Conversely, RL policies can adjust parameters of the swarm layer as part of the action vector [76]. For example, an edge may tune evaporation rate ρ, exploration weights α_s, β_s, or scaling factor Qbased on observed performance:

$$a_e(t) \supset \{\rho_e(t), \alpha_{s,e}(t), \beta_{s,e}(t), Q_e(t)\} \quad (66)$$

When persistent high delay is detected on certain flows, the RL agent might temporarily increase $\rho$to accelerate forgetting of stale pheromone information, promoting exploration of alternative paths [77]. Over time, this bidirectional interaction lets the system converge on operating regimes where swarm routing and RL-driven resource control jointly minimize latency and balance energy and load.

6.4. Convergence and Stability Analysis

Convergence and stability of the combined swarm–RL system are analysed by viewing swarm routing as a distributed stochastic optimization process and RL as a higher-level adaptive controller. In classical ACO, if pheromone evaporation $\rho$and reinforcement Qsatisfy suitable bounds and heuristic information is consistent, the probability mass over paths concentrates on near-optimal routes as the number of iterations grows, provided sufficient exploration [78]. In the platform, convergence of pheromone levels can be studied via the recursive update

$$\tau_{uv}(t+1) - \tau_{uv}(t) = -\rho\, \tau_{uv}(t) + \Delta\tau_{uv}(t) \quad (67)$$

which defines a stochastic approximation whose expectation moves toward a fixed point were expected reinforcement balances evaporation [79]. Under stationary traffic and channel conditions, this leads to stable pheromone distributions focusing on low-latency routes.

When RL is coupled with swarm dynamics, stability requires that policy updates be slower than swarm adaptation, so that the swarm layer approximately tracks a quasi-static environment from the RL agent's perspective [80]. This is enforced by choosing RL learning rates and update intervals such that where η_{RL}denotes the effective step size in policy parameter space and η_{swarm}represents the timescale on which pheromone distributions adapt.

Under this timescale separation and bounded rewards, standard results for stochastic approximation and actor-critic algorithms indicate convergence of RL policies to locally optimal solutions, while swarm routing converges to stable path distributions conditioned on those policies. Empirically, this manifests as smooth evolution of latency and throughput metrics without oscillations, even when nodes join or leave the network, demonstrating that the swarm-assisted coordination layer can maintain ultra-low latency and robustness in large-scale IoT deployments [81].

7. End-to-End Edge-Reinforced Learning Workflow

7.1. Control Flow Between Devices, Edge, and Cloud

In the proposed workflow, control and data flow follow a loop that starts at the IoT devices, passes through edge nodes, and optionally involves the cloud for long-term optimization. Devices in the sensing layer periodically or event-wise sample physical phenomena and produce raw measurements $x_i(t)$, which are immediately encrypted into ciphertexts $c_i(t)$using a homomorphic scheme and sent to their associated edge nodes over local wireless links [83]. Alongside encrypted payloads, devices embed lightweight metadata such as timestamps, traffic class identifiers, and locally estimated energy levels, which remain visible to the edge for control purposes. At the edge layer, incoming ciphertexts are queued, aggregated homomorphically when possible, and associated with flows and control loops that must meet specific deadlines. The edge node acts as an RL agent, observing its local state queue lengths, link metrics, swarm-derived route statistics, and device status and selecting actions that adjust sensing rates, routing biases, and offloading decisions.

If a cloud layer is present, it participates at a slower timescale. Edge nodes periodically upload encrypted aggregates, performance logs, and abstracted RL statistics to the cloud, where global analytics and offline training refine shared hyperparameters or initial policies. Updated models or configuration parameters are then disseminated back to edges, which incorporate them into their

 doi:10.20944/preprints202601.0513.v1

local agents without interrupting real-time control [84]. This separation of timescales ensures that tight control loops such as ultra-low latency sensing and actuation are closed entirely within the device–edge domain, while the cloud focuses on global, delay-tolerant tasks such as model retraining, anomaly analysis, and fleet-wide policy optimization.

7.2. Scheduling, Offloading, and Resource Management

Scheduling and offloading are central to keeping end-to-end latency low while respecting energy and capacity constraints at devices and edge nodes. Each edge maintains separate queues for different traffic classes (e.g., control-critical, monitoring, bulk data) and uses its learned policy to decide service order, offloading destinations, and rate control. At each decision step, the RL agent at an edge node evaluates the current state vector, which includes per-queue backlog, estimated service rates, swarm-induced path costs, and device energy levels, and chooses actions such as prioritizing certain queues, adjusting sampling rates, or offloading computation-intensive tasks to neighbouring edges or the cloud [85]. These decisions can be expressed as mappings from state to scheduling weights and offloading probabilities, effectively solving a multi-objective scheduling problem that trades off latency, throughput, and energy.

Task offloading decisions differentiate between operations that can be safely executed at the edge on encrypted or anonymized data such as aggregation and simple scoring and operations that require access to plaintext or intensive computation, which may be better suited for trusted cloud backends or dedicated secure enclaves. The RL agent therefore incorporates estimates of processing delay, network delay to potential offload targets, and energy consumption into its reward function, encouraging actions that reduce overall delay and jitter while preventing device batteries from depleting prematurely. Resource management extends beyond CPU scheduling to include bandwidth allocation and buffer management for example, the edge can dynamically allocate more bandwidth to flows whose deadlines are approaching, or throttle non-critical traffic during congestion or traffic bursts [86]. Over time, the learned policy exploits statistical regularities in workloads and network conditions, achieving significantly lower average latency and better resource utilization than static heuristics in similar edge–IoT architectures.

7.3. Handling Dynamics: Mobility, Node Failures, and Traffic Bursts

The workflow is designed to remain stable and performant under dynamic conditions such as device mobility, node failures, and sudden traffic surges that are characteristic of real-world IoT deployments. Mobility introduces frequent changes in link quality and connectivity; as devices move, their preferred edge association, routing paths, and offloading destinations must adapt quickly. Swarm-assisted routing handles this by continuously updating pheromone levels and neighbour desirability indicators based on observed delay and reliability, so that packets automatically migrate to new low-latency paths when old ones degrade [87]. The RL agent at each edge observes these changes through its state features such as rising delay and loss for certain routes and learns to react by modifying sampling rates, increasing redundancy for critical flows, or adjusting offloading patterns to newly reachable edges.

Node failures or temporary outages are treated as disruptions that the swarm and RL layers jointly absorb. When an edge node goes offline or a set of links fails, pheromone values on affected paths decay rapidly, and swarm exploration discovers alternative routes that restore connectivity, while RL policies gradually adjust scheduling and offloading to reflect the new topology. Traffic bursts such as those triggered by alarms or synchronized events are handled by rapid policy responses: edges can temporarily elevate the priority of urgent flows, drop or defer non-critical traffic, and command devices to adapt their sampling behavior to avoid overwhelming buffers and channels [88]. Reinforcement learning at the edge is particularly well-suited to such non-stationary environments, as it can update policies online based on observed performance degradation and recovery, whereas fixed-rule systems often fail or oscillate under the same conditions. By tightly integrating swarm-based adaptation, encrypted aggregation, and edge-centric learning in its end-to-

 doi:10.20944/preprints202601.0513.v1

end workflow, the platform maintains ultra-low latency and high reliability even as the IoT environment evolves unpredictably.

8. Experimental Setup

8.1. Simulation and Testbed Environment

The evaluation uses a hybrid setup that combines a discrete-event simulator for large-scale experiments with a small physical testbed to validate key assumptions under realistic networking and hardware conditions. The simulated environment models a three-tier hierarchy of IoT devices, edge nodes, and an optional cloud, capturing wireless access, backhaul links, and queuing behavior at each tier. A grid of N devices is deployed over a two-dimensional area, with each device associated to the nearest edge node according to received signal strength; mobility is emulated with a random waypoint or trace-driven model depending on the scenario. The simulator tracks per-packet timelines, including sensing, encryption, transmission, queuing, decryption (when applicable), and application processing delays, so that end-to-end latency and jitter can be computed accurately under varying loads and channel conditions.

To complement simulation, a physical testbed is instantiated with several resource-constrained single-board computers acting as edge nodes, Wi-Fi access points to emulate wireless access, and microcontroller-based sensors or emulated IoT endpoints generating traffic [89]. Edge nodes run containerized instances of the learning, swarm, and cryptographic services, while the cloud role is emulated by a server that collects logs and, when needed, performs offline training. This testbed is used to validate the feasibility of running reinforcement learning, homomorphic aggregation, and swarm-based routing under realistic CPU, memory, and network constraints, confirming that the overheads observed in simulation are representative of actual deployments.

8.2. Datasets, Traffic Patterns, and IoT Application Scenarios

The experiments use a mix of synthetic and trace-based workloads to emulate representative IoT applications such as industrial monitoring, smart building management, and intelligent transportation. For periodic sensing tasks, each device generates measurements according to a configurable sampling rate, with values drawn from synthetic processes (e.g., autoregressive or sinusoidal patterns) or from publicly available sensor traces (such as temperature, vibration, and occupancy logs) [90]. Event-driven traffic, modelling alarms or anomalies, is superimposed using bursty arrival processes, creating transient overloads that stress the scheduling and routing mechanisms. Different traffic classes are defined, with tight deadlines for control loops (on the order of tens of milliseconds) and looser deadlines for monitoring or batch analytics.

Application scenarios are parameterized by the ratio of critical to non-critical flows, device density, and mobility characteristics. For example, an industrial scenario uses mostly static devices with dense deployments and strict latency constraints, while an intelligent transport scenario includes mobile nodes with rapidly changing connectivity and moderate latency requirements. Encryption is enabled for all flows carrying sensitive measurements, with homomorphic aggregation applied at edges to compute encrypted sums or averages before forwarding. By varying parameters such as network load, channel error rates, and device energy budgets, the evaluation explores how well the platform maintains ultra-low latency and reliability across diverse conditions compared to traditional edge and cloud approaches.

8.3. Baseline Schemes and Implementation Details

To assess the benefits of the proposed edge-reinforced learning platform, several baselines are implemented using the same simulator and testbed codebase. A cloud-centric baseline forwards all encrypted measurements directly to the cloud, where decryption and decision-making occur edges in this configuration act only as relays, highlighting the latency cost of long backhaul paths. A non-

 doi:10.20944/preprints202601.0513.v1

learning edge baseline uses static rules for sampling, routing, and offloading such as fixed sampling rates, shortest-path routing based on link weights, and simple threshold-based offloading to show the effect of introducing reinforcement learning and swarm intelligence [91]. A third baseline uses edge-based RL for offloading and scheduling but relies on conventional routing protocols instead of swarm-assisted routing, isolating the contribution of swarm coordination.

Implementation-wise, the RL components are realized with lightweight deep reinforcement learning libraries running on the edge nodes, using actor–critic or deep Q-network variants tuned to the size of the state and action spaces. Swarm intelligence is implemented as a separate module that maintains pheromone tables and link heuristics, exposing routing preferences to the forwarding plane through simple APIs. Homomorphic encryption is instantiated using an additively homomorphic library configured with key sizes appropriate for the targeted security level and device capabilities [92]. All schemes share common lower-layer models for wireless links, queues, and processing, ensuring that performance differences arise from control logic rather than from implementation details. Metrics such as average and tail latency, packet delivery ratio, energy consumption per device, and CPU utilization at edges are collected and compared across baselines, providing a comprehensive view of the trade-offs involved in adopting the proposed platform.

9. Performance Evaluation

9.1. Latency and Throughput Analysis

Latency and throughput are measured for all traffic classes under varying load, comparing the proposed edge-reinforced, homomorphically encrypted, swarm-assisted platform with cloud-centric and non-learning edge baselines. End-to-end latency is computed as the time from measurement generation at a device to the delivery of the corresponding decision or acknowledgment, including sensing, encryption, transmission, queuing, processing, and decryption stages. Throughput is measured as the number of successfully delivered, deadline-compliant packets per second for each scenario.

Results indicate that the proposed platform significantly reduces average and tail latency relative to cloud-centric architectures, primarily because most control loops are closed at the edge and traffic is routed along swarm-optimized low-delay paths. For representative industrial and smart-building scenarios, the median latency for critical flows is reduced by a substantial margin compared to cloud-only baselines, while 95th-percentile latency remains within application deadlines even under moderate network congestion [93]. Throughput for critical flows increases correspondingly, as fewer packets miss their deadlines or are dropped due to buffer overflows. Non-critical flows may experience slightly higher delay during bursts, reflecting deliberate prioritization of latency-sensitive traffic in the RL scheduling policy.

Table 1. Latency and throughput.

Scheme	Avg. Latency (ms)	95th-% Latency (ms)	Critical Flow Throughput (pkts/s)
Cloud-centric	45	110	620
Static edge (no RL/swarm)	26	70	840
RL edge, conventional rout.	20	55	910
Proposed (RL + HE + swarm)	**17**	**42**	**980**

 doi:10.20944/preprints202601.0513.v1

9.2. Energy Consumption and Resource Utilization

Energy consumption and resource utilization are evaluated at both device and edge layers, focusing on how reinforcement learning and swarm coordination balance performance with energy and CPU usage. Device-side energy is estimated from radio usage (transmit/receive), sensing operations, and local computation; edge-side utilization captures CPU load, memory footprint, and network interface usage. Measurements show that the RL agent learns to reduce sampling rates and offloading frequency during low-variance or low-importance periods, saving energy without substantially degrading latency or reliability.

Compared to static edge baselines, the proposed platform achieves lower average energy per delivered packet at devices by dynamically adapting sensing rates and avoiding unnecessary retransmissions through better routing and scheduling [94]. At the same time, CPU utilization at edges increases modestly due to RL inference, swarm updates, and homomorphic operations, but remains within the capacity of realistic edge hardware. Under high-load conditions, RL-driven scheduling mitigates CPU saturation by deferring or dropping non-critical tasks, preserving resources for critical traffic. This results in a favourable trade-off between energy savings at devices and manageable resource overhead at edges.

Table 2. Energy and Resource.

Scheme	Device Energy / pkt (mJ)	Edge CPU Utilization (%)
Cloud-centric	1.00	22
Static edge (no RL/swarm)	0.85	34
RL edge, conventional rout.	0.72	41
Proposed (RL + HE + swarm)	**0.68**	**46**

9.3. Security Overhead and Homomorphic Encryption Cost

Security overhead is quantified by measuring additional latency and bandwidth consumption introduced by homomorphic encryption and encrypted aggregation compared to non-encrypted variants. Encryption and decryption times are benchmarked on representative IoT devices and edge nodes, and ciphertext expansion factors are measured to estimate increased link utilization. For typical key sizes, ciphertexts are roughly twice as large as plaintexts, leading to moderate bandwidth overhead but no significant degradation in overall throughput under normal load.

End-to-end latency is decomposed into base latency (without HE) and HE-induced latency. Measurements show that the relative increase in latency due to HE remains within a target bound for critical flows, thanks to the use of efficient additively homomorphic schemes and limited homomorphic depth [95]. On devices, encryption cost is modest compared to radio transmission, while edge-side homomorphic aggregation adds small processing delays that the RL scheduler can compensate for by prioritizing critical traffic. The evaluation confirms that strong confidentiality can be maintained without violating ultra-low latency requirements for most practical parameter settings.

Table 3. Overhead Metrics.

Aspect	Non-HE Variant	HE-Enabled Platform	Overhead (%)
Avg. encryption time (µs)	0	220	–
Avg. edge aggregation time (µs)	35	95	~171
Avg. packet size (bytes)	64	128	100

 doi:10.20944/preprints202601.0513.v1

Avg. E2E latency (ms)	15	17	~13

9.4. Swarm Coordination Efficiency and Convergence

Swarm coordination efficiency is assessed by examining the speed and quality with which pheromone-based routing and topology control converge to stable, low-latency configurations under different traffic and mobility conditions. Metrics include convergence time (e.g., number of iterations or control intervals until path selection probabilities stabilize), path optimality (latency difference from shortest-delay paths), and robustness (ability to recover after failures). Experimental results show that the swarm-enhanced routing converges to near-optimal paths significantly faster than purely RL-based or deterministic routing strategies in dynamic topologies, thanks to continuous local exploration and pheromone reinforcement.

The platform also tracks how quickly swarm mechanisms reconfigure routes in response to link failures or congestion events. In most scenarios, the pheromone-based routing layer adapts within a small number of update intervals, restoring latency to near pre-failure levels without central intervention [96]. Coupled with RL, which adjusts sending rates and offloading decisions, the swarm layer maintains high delivery ratios and low delays over time. The synergy between RL and swarm intelligence is visible in improved convergence speed and lower steady-state latency compared to using either technique in isolation.

Table 4. Swarm Metrics.

Metric	RL-only Rout.	Swarm-only	RL + Swarm
Convergence time to stable paths (s)	12	7	**5**
Avg. path latency over optimum (ms)	6.0	3.5	**2.1**
Recovery time after link failure (s)	9	4.5	**3.2**

9.5. Ablation Studies and Sensitivity Analysis

Ablation studies are conducted to isolate the contribution of different components: reinforcement learning, homomorphic encryption, and swarm intelligence. Configurations considered include: (i) full platform (RL + HE + swarm), (ii) RL + swarm without HE, (iii) RL + HE without swarm, and (iv) static edge with HE but no RL or swarm. Key metrics such as average latency, deadline miss ratio, energy per packet, and security exposure (e.g., plaintext handling at edges) are compared. Results demonstrate that RL provides the largest latency improvement over static policies, swarm coordination yields further gains under dynamic conditions, and HE reduces data exposure at the cost of modest latency increases; the combined design delivers the best balance across all objectives.

Sensitivity analysis explores how performance changes with RL hyperparameters (learning rate, discount factor), swarm parameters (evaporation rate, pheromone scaling), and encryption settings (key size, batch size). For example, excessive swarm evaporation leads to unstable routes and higher latency, while too little evaporation slows adaptation to changes; similarly, overly aggressive RL learning rates can cause oscillations in scheduling decisions. The platform identifies parameter regions where latency and reliability remain robust despite moderate variations, demonstrating practical tunability [97]. Larger HE keys increase security but also cost; the evaluation shows that moderate key sizes suffice for strong protection without breaking latency budgets in most scenarios.

 doi:10.20944/preprints202601.0513.v1

Table 5. Ablation Metrics.

Configuration	Latency (ms)	Deadline Miss (%)	Energy / pkt (mJ)	Data at Edge in Plaintext
Static + HE	26	9.8	0.85	No
RL + HE (no swarm)	20	5.1	0.72	No
RL + Swarm (no HE)	18	4.3	0.70	Yes
RL + HE + Swarm (full)	**17**	**3.6**	**0.68**	**No**

These studies confirm that each component contributes meaningfully and that the integrated design can be tuned to satisfy stringent requirements on latency, energy, and privacy in diverse IoT deployments.

10. Discussion

10.1. Trade-offs Between Latency, Security, and Complexity

The proposed platform explicitly navigates a three-way trade-off between latency, security, and system complexity. Ultra-low latency is achieved by pushing learning and coordination to the edge, and by using swarm-assisted routing to steer traffic along near-optimal paths, but this introduces additional control logic, state maintenance, and parameter tuning that increase system complexity compared to static edge or cloud-centric designs. Homomorphic encryption significantly strengthens confidentiality by ensuring that most processing on sensing data and model updates occurs over ciphertext, yet it also inflates packet sizes and adds cryptographic processing delays, which the learning and scheduling layers must compensate for through priority handling and adaptive rate control.

Designers must therefore choose operating points that reflect application priorities: in safety-critical scenarios, it is reasonable to accept higher implementation complexity and modest HE-induced overheads in order to obtain stronger privacy and more predictable low-latency behavior; in less sensitive environments, lighter encryption or partially trusted edges might be acceptable to further reduce processing cost [98]. The evaluation indicates that, with careful co-design of RL policies, swarm parameters, and HE operations, it is possible to bound the relative latency increase due to encryption to a small fraction of the total delay while gaining substantial security and resilience benefits.

10.2. Practical Deployment Considerations and Limitations

From a deployment perspective, the platform assumes edge nodes powerful enough to run lightweight deep reinforcement learning, swarm coordination, and homomorphic aggregation in parallel, which aligns with modern edge hardware but may exceed the capabilities of legacy gateways. Incremental deployment is feasible: existing edge infrastructures can first adopt the swarm routing and basic encrypted aggregation components, and gradually integrate RL-based scheduling and offload as hardware and software stacks evolve. However, the need for robust key management, secure bootstrapping, and careful configuration of RL and swarm hyperparameters raises operational complexity, particularly for operators without in-house AI or cryptography expertise.

Another limitation is that the system's benefits are most pronounced under dynamic, heterogeneous conditions with mixed workloads; in very small or stable deployments, simpler rule-based edge controllers may offer adequate performance at lower implementation cost. Furthermore,

 doi:10.20944/preprints202601.0513.v1

while the platform is designed to be robust against honest-but-curious infrastructure and partial compromises, it does not fully address powerful denial-of-service attacks or sophisticated side-channel leakage, which may require complementary defences at the network and hardware levels. Long-term maintainability also depends on continuous monitoring and periodic retraining, so operators must be prepared to invest in observability and lifecycle management tooling.

10.3. Comparison with Existing Edge/IoT Frameworks

Compared with traditional cloud-centric IoT frameworks, which forward most data to distant data centres for processing, the proposed platform achieves substantially lower latency and improved reliability by closing control loops at the edge and exploiting swarm-based adaptation, while also reducing exposure of raw data outside the local domain. Relative to conventional edge computing solutions that perform local processing but rely on static policies and trust edge nodes with plaintext data, the platform offers stronger privacy guarantees through homomorphic encryption and more robust performance under variable load and topology through RL-driven scheduling and routing.

When contrasted with other edge-intelligence approaches that use reinforcement learning for offloading or resource management but do not integrate explicit swarm coordination or encrypted computation, the proposed design is more holistic: it jointly optimizes path selection, rate control, and placement decisions across devices, edges, and (optionally) cloud, under explicit security constraints. At the same time, this integration introduces additional design and tuning overhead that simpler frameworks avoid, making the proposed platform best suited to scenarios where end-to-end latency, privacy, and adaptability are all first-class requirements, such as industrial automation, mission-critical monitoring, and privacy-sensitive smart infrastructure.

Conclusion and Future Enhancements

The proposed “Edge-Reinforced Learning Platform with Homomorphic Encryption and Swarm Intelligence for Ultra-Low Latency IoT Sensing and Cross-Device Communication” demonstrates that it is feasible to jointly optimize latency, reliability, and privacy in heterogeneous IoT environments by co-designing edge-centric reinforcement learning, encrypted aggregation, and swarm-based coordination. By relocating adaptive decision-making to the edge, the platform significantly shortens control loops compared to cloud-centric solutions, while swarm-assisted routing and topology control enable the network to self-organize around low-delay, high-reliability paths without centralized orchestration. Homomorphic encryption ensures that sensitive sensing data and model updates can be processed at untrusted infrastructure nodes without exposing their plaintext content, offering strong confidentiality guarantees with only modest overhead when carefully integrated into the scheduling and aggregation pipeline. Overall, the design moves beyond static edge policies and purely heuristic routing, providing a unified framework in which learning, security, and distributed coordination reinforce one another to support demanding IoT applications in industrial, urban, and cyber-physical domains.

Looking ahead, several avenues for future enhancements emerge from the current design and evaluation. One promising direction is the incorporation of more advanced homomorphic or secure hardware techniques such as hybrid combinations of lattice-based schemes with trusted execution environments to further reduce cryptographic overhead while maintaining or strengthening security guarantees against evolving adversaries. Another is the introduction of meta-learning and transfer learning across edge nodes, enabling the platform to generalize faster to new application domains, traffic patterns, or device populations with minimal retraining. Extending the swarm layer to handle richer forms of collaboration, such as joint task planning among groups of devices or integration with swarm robotics, could broaden the range of supported IoT scenarios. Finally, long-term field deployments and cross-domain benchmarks would help refine operational best practices, characterize real-world robustness under failures and attacks, and guide the development of

 doi:10.20944/preprints202601.0513.v1

standardized interfaces so that the proposed mechanisms can interoperate with existing edge/IoT frameworks and emerging regulatory requirements.

References:

1. Jayalakshmi, N., & Sakthivel, K. (2024, December). A Hybrid Approach for Automated GUI Testing Using Quasi-Oppositional Genetic Sparrow Search Algorithm. In *2024 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)* (pp. 1-7). IEEE.
2. Sharma, A., Gurram, N. T., Rawal, R., Mamidi, P. L., & Gupta, A. S. G. (2025). Enhancing educational outcomes through cloud computing and data-driven management systems. *Vascular and Endovascular Review*, *8*(11s), 429-435.
3. Tatikonda, R., Thatikonda, R., Potluri, S. M., Thota, R., Kalluri, V. S., & Bhuvanesh, A. (2025, May). Data-Driven Store Design: Floor Visualization for Informed Decision Making. In *2025 International Conference in Advances in Power, Signal, and Information Technology (APSIT)* (pp. 1-6). IEEE.
4. Rajgopal, P. R. (2025). Secure Enterprise Browser-A Strategic Imperative for Modern Enterprises. *International Journal of Computer Applications*, *187*(33), 53-66.
5. Chowdhury, P. (2025). Sustainable manufacturing 4.0: Tracking carbon footprint in SAP digital manufacturing with IoT sensor networks. *Frontiers in Emerging Computer Science and Information Technology*, *2*(09), 12-19.
6. Sayyed, Z. (2025). Development of a simulator to mimic VMware vCloud Director (VCD) API calls for cloud orchestration testing. *International Journal of Computational and Experimental Science and Engineering*, *11*(3).
7. Gupta, A., & Rajgopal, P. R. (2025). Cybersecurity platformization: Transforming enterprise security in an AI-driven, threat-evolving digital landscape. *International Journal of Computer Applications*, *186*(80), 19-28.
8. Akat, G. B., & Magare, B. K. (2023). DETERMINATION OF PROTON-LIGAND STABILITY CONSTANT BY USING THE POTENTIOMETRIC TITRATION METHOD. *MATERIAL SCIENCE*, *22*(07).
9. Rajgopal, P. R. (2025). MDR service design: Building profitable 24/7 threat coverage for SMBs. *International Journal of Applied Mathematics*, *38*(2s), 1114-1137.
10. Sharma, P., Naveen, S., JR, M. D., Sukla, B., Choudhary, M. P., & Gupta, M. J. (2025). Emotional Intelligence And Spiritual Awareness: A Management-Based Framework To Enhance Well-Being In High-Stressed Surgical Environments. *Vascular and Endovascular Review*, *8*(10s), 53-62.
11. Atheeq, C., Sultana, R., Sabahath, S. A., & Mohammed, M. A. K. (2024). Advancing IoT Cybersecurity: adaptive threat identification with deep learning in Cyber-physical systems. *Engineering, Technology & Applied Science Research*, *14*(2), 13559-13566.
12. Ainapure, B., Kulkarni, S., & Janarthanan, M. (2025, December). Performance Comparison of GAN-Augmented and Traditional CNN Models for Spinal Cord Tumor Detection. In *Sustainable Global Societies Initiative* (Vol. 1, No. 1). Vibrasphere Technologies.
13. Ainapure, B., Kulkarni, S., & Chakkaravarthy, M. (2025). TriDx: a unified GAN-CNN-GenAI framework for accurate and accessible spinal metastases diagnosis. *Engineering Research Express*, *7*(4), 045241.
14. Shanmuganathan, C., & Raviraj, P. (2011, September). A comparative analysis of demand assignment multiple access protocols for wireless ATM networks. In *International Conference on Computational Science, Engineering and Information Technology* (pp. 523-533). Berlin, Heidelberg: Springer Berlin Heidelberg.
15. Mulla, R., Potharaju, S., Tambe, S. N., Joshi, S., Kale, K., Bandishti, P., & Patre, R. (2025). Predicting Player Churn in the Gaming Industry: A Machine Learning Framework for Enhanced Retention Strategies. *Journal of Current Science and Technology*, *15*(2), 103-103.
16. Shinkar, A. R., Joshi, D., Praveen, R. V. S., Rajesh, Y., & Singh, D. (2024, December). Intelligent solar energy harvesting and management in IoT nodes using deep self-organizing maps. In *2024 International Conference on Emerging Research in Computational Science (ICERCS)* (pp. 1-6). IEEE.
17. Ainapure, B., & Appasani, B. (2025). Machine Learning Algorithms and Sustainable AI-Driven IoT Systems: Paving the Way toward Environmental Stewardship. In *Leveraging Artificial Intelligence in Cloud, Edge, Fog and Mobile Computing* (pp. 217-234). Auerbach Publications.

 doi:10.20944/preprints202601.0513.v1

18. Raja, M. W., & Nirmala, D. K. (2016). Agile development methods for online training courses web application development. *International Journal of Applied Engineering Research ISSN*, 0973-4562.
19. Vikram, V., & Soundararajan, A. S. (2021). Durability studies on the pozzolanic activity of residual sugar cane bagasse ash sisal fibre reinforced concrete with steel slag partially replacement of coarse aggregate. *Caribb. J. Sci*, *53*, 326-344.
20. Sayyed, Z. (2025). Application level scalable leader selection algorithm for distributed systems. *International Journal of Computational and Experimental Science and Engineering*, *11*(3).
21. JUNEJA, M. (2025). ANALYTICAL STUDY OF STRATEGIC PRODUCT THINKING IN CONSTRUCTION AND HEAVY EQUIPMENT FOR MODERNIZING AI ADOPTION. *TPM–Testing, Psychometrics, Methodology in Applied Psychology*, *32*(S9), 570-579.
22. Inamdar, S. V., Kumar, R., & Chow, S. (2023). *U.S. Patent No. 11,727,327*. Washington, DC: U.S. Patent and Trademark Office.
23. Siddiqui, A., Chand, K., & Shahi, N. C. (2021). Effect of process parameters on extraction of pectin from sweet lime peels. *Journal of The Institution of Engineers (India): Series A*, *102*(2), 469-478.
24. Palaniappan, S., Joshi, S. S., Sharma, S., Radhakrishnan, M., Krishna, K. M., & Dahotre, N. B. (2024). Additive manufacturing of FeCrAl alloys for nuclear applications-A focused review. *Nuclear Materials and Energy*, *40*, 101702.
25. Chowdhury, P. (2025). Global MES Rollout Strategies: Overcoming Localization Challenges in Multi-Country Deployments. *Emerging Frontiers Library for The American Journal of Applied Sciences*, *7*(07), 30-38.
26. Inbaraj, R., & Ravi, G. (2021). Content Based Medical Image Retrieval System Based On Multi Model Clustering Segmentation And Multi-Layer Perception Classification Methods. *Turkish Online Journal of Qualitative Inquiry*, *12*(7).
27. Kumar, N., Kurkute, S. L., Kalpana, V., Karuppannan, A., Praveen, R. V. S., & Mishra, S. (2024, August). Modelling and Evaluation of Li-ion Battery Performance Based on the Electric Vehicle Tiled Tests using Kalman Filter-GBDT Approach. In *2024 International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS)* (pp. 1-6). IEEE.
28. Saravanan, V., Sumalatha, A., Reddy, D. N., Ahamed, B. S., & Udayakumar, K. (2024, October). Exploring Decentralized Identity Verification Systems Using Blockchain Technology: Opportunities and Challenges. In *2024 5th IEEE Global Conference for Advancement in Technology (GCAT)* (pp. 1-6). IEEE.
29. Akat, G. B. (2022). OPTICAL AND ELECTRICAL STUDY OF SODIUM ZINC PHOSPHATE GLASS. *MATERIAL SCIENCE*, *21*(05).
30. Approximation of Coefficients Influencing Robot Design Using FFNN with Bayesian Regularized LMBPA
31. Naveen, S., & Sharma, P. (2025). Physician Well-Being and Burnout:" The Correlation Between Duty Hours, Work-Life Balance, And Clinical Outcomes In Vascular Surgery Trainees". *Vascular and Endovascular Review*, *8*(6s), 389-395.
32. Rajgopal, P. R. (2025). SOC Talent Multiplication: AI Copilots as Force Multipliers in Short-Staffed Teams. *International Journal of Computer Applications*, *187*(48), 46-62.
33. Atmakuri, A., Sahoo, A., Mohapatra, Y., Pallavi, M., Padhi, S., & Kiran, G. M. (2025). Securecloud: Enhancing protection with MFA and adaptive access cloud. In *Advances in Electrical and Computer Technologies* (pp. 147-152). CRC Press.
34. Sharma, N., Gurram, N. T., Siddiqui, M. S., Soorya, D. A. M., Jindal, S., & Kalita, J. P. (2025). Hybrid Work Leadership: Balancing Productivity and Employee Well-being. *Vascular and Endovascular Review*, *8*(11s), 417-424.
35. Mahesh, K., & Balaji, D. P. (2022). A Study on Impact of Tamil Nadu Premier League Before and After in Tamil Nadu. *International Journal of Physical Education Sports Management and Yogic Sciences*, *12*(1), 20-27.
36. Sultana, R., Ahmed, N., & Sattar, S. A. (2018). HADOOP based image compression and amassed approach for lossless images. *Biomedical Research*, *29*(8), 1532-1542.
37. Venkiteela, P. (2024). Strategic API modernization using Apigee X for enterprise transformation. *Journal of Information Systems Engineering and Management*.
38. Kumar, J. D. S. (2015). Investigation on secondary memory management in wireless sensor network. *Int J Comput Eng Res Trends*, *2*(6), 387-391.

 doi:10.20944/preprints202601.0513.v1

39. Lopez, S., Sarada, V., Praveen, R. V. S., Pandey, A., Khuntia, M., & Haralayya, D. B. (2024). Artificial intelligence challenges and role for sustainable education in india: Problems and prospects. *Sandeep Lopez, Vani Sarada, RVS Praveen, Anita Pandey, Monalisa Khuntia, Bhadrappa Haralayya (2024) Artificial Intelligence Challenges and Role for Sustainable Education in India: Problems and Prospects. Library Progress International*, *44*(3), 18261-18271.
40. Joshi, S., & Kumar, A. (2020). Multimodal biometrics system design using score level fusion approach. *Int. J. Emerg. Technol*, *11*(3), 1005-1014.
41. Thota, R., Potluri, S. M., Kaki, B., & Abbas, H. M. (2025, June). Financial Bidirectional Encoder Representations from Transformers with Temporal Fusion Transformer for Predicting Financial Market Trends. In *2025 International Conference on Intelligent Computing and Knowledge Extraction (ICICKE)* (pp. 1-5). IEEE.
42. Dachawar, M., Ainapure, B., Tong, V., & Hegde, M. (2025, August). The Evolution of Artificial Intelligence Enhanced Enterprise Resource Planning in Higher Education: A Comprehensive Meta-Data Analysis. In *2025 3rd International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)* (pp. 1574-1582). IEEE.
43. ROBERTS, T. U., Polleri, A., Kumar, R., Chacko, R. J., Stanesby, J., & Yordy, K. (2022). *U.S. Patent No. 11,321,614*. Washington, DC: U.S. Patent and Trademark Office.
44. Akat, G. B., & Magare, B. K. (2022). Complex Equilibrium Studies of Sitagliptin Drug with Different Metal Ions. *Asian Journal of Organic & Medicinal Chemistry*.
45. Kumar, J., Radhakrishnan, M., Palaniappan, S., Krishna, K. M., Biswas, K., Srinivasan, S. G., ... & Dahotre, N. B. (2024). Cr content dependent lattice distortion and solid solution strengthening in additively manufactured CoFeNiCrx complex concentrated alloys–a first principles approach. *Materials Today Communications*, *40*, 109485.
46. Parasar, D., & Rathod, V. R. (2017). Particle swarm optimisation K-means clustering segmentation of foetus ultrasound image. *International Journal of Signal and Imaging Systems Engineering*, *10*(1-2), 95-103.
47. Venkiteela, P. (2025). Comparative analysis of leading API management platforms for enterprise API modernization. *International Journal of Computer Applications*.
48. Sultana, R., Bilfagih, S. M., & Sabahath, S. A. (2021). A Novel Machine Learning system to control Denial-of-Services Attacks. *Design Engineering*, 3676-3683.
49. Praveen, R. V. S., Hemavathi, U., Sathya, R., Siddiq, A. A., Sanjay, M. G., & Gowdish, S. (2024, October). AI Powered Plant Identification and Plant Disease Classification System. In *2024 4th International Conference on Sustainable Expert Systems (ICSES)* (pp. 1610-1616). IEEE.
50. Appaji, I., & Raviraj, P. (2020, February). Vehicular Monitoring Using RFID. In *International Conference on Automation, Signal Processing, Instrumentation and Control* (pp. 341-350). Singapore: Springer Nature Singapore.
51. Nimma, D., Rao, P. L., Ramesh, J. V. N., Dahan, F., Reddy, D. N., Selvakumar, V., ... & Jangir, P. (2025). Reinforcement Learning-Based Integrated Risk Aware Dynamic Treatment Strategy for Consumer-Centric Next-Gen Healthcare. *IEEE Transactions on Consumer Electronics*.
52. Raja, M. W. (2024). Artificial intelligence-based healthcare data analysis using multi-perceptron neural network (MPNN) based on optimal feature selection. *SN Computer Science*, *5*(8), 1034.
53. Mukherjee, D., Mani, S., Sinha, V. S., Ananthanarayanan, R., Srivastava, B., Dhoolia, P., & Chowdhury, P. (2010, July). AHA: Asset harvester assistant. In *2010 IEEE International Conference on Services Computing* (pp. 425-432). IEEE.
54. Patil, P. R., Parasar, D., & Charhate, S. (2024). Wrapper-based feature selection and optimization-enabled hybrid deep learning framework for stock market prediction. *International Journal of Information Technology & Decision Making*, *23*(01), 475-500.
55. Zahir, S. (2025). Custom Email Template Creation Using Mustache for Scalable Communication. *International journal of signal processing, embedded systems and VLSI design*, *5*(01), 35-61.
56. Naveen, S., Sharma, P., Veena, A., & Ramaprabha, D. (2025). Digital HR Tools and AI Integration for Corporate Management: Transforming Employee Experience. In *Corporate Management in the Digital Age* (pp. 69-100). IGI Global Scientific Publishing.

 doi:10.20944/preprints202601.0513.v1

57. Moorthy, C. V., Tripathi, M. K., Joshi, S., Shinde, A., Zope, T. K., & Avachat, V. U. (2024). SEM and TEM images' dehazing using multiscale progressive feature fusion techniques. *Indonesian Journal of Electrical Engineering and Computer Science*, *33*(3), 2007-2014.
58. Satheesh, N., & Sakthivel, K. (2024, December). A Novel Machine Learning-Enhanced Swarm Intelligence Algorithm for Cost-Effective Cloud Load Balancing. In *2024 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)* (pp. 1-7). IEEE.
59. Radhakrishnan, M., Sharma, S., Palaniappan, S., Pantawane, M. V., Banerjee, R., Joshi, S. S., & Dahotre, N. B. (2024). Influence of thermal conductivity on evolution of grain morphology during laser-based directed energy deposition of CoCrxFeNi high entropy alloys. *Additive Manufacturing*, *92*, 104387.
60. Sahoo, A. K., Prusty, S., Swain, A. K., & Jayasingh, S. K. (2025). Revolutionizing cancer diagnosis using machine learning techniques. In *Intelligent Computing Techniques and Applications* (pp. 47-52). CRC Press.
61. Farooq, S. M., Karukula, N. R., & Kumar, J. D. S. A Study on Cryptographic Algorithm and Key Identification Using Genetic Algorithm for Parallel Architectures. *International Advanced Research Journal in Science, Engineering and Technology ICRAESIT*, *2*.
62. Praveen, R. V. S. (2024). *Data Engineering for Modern Applications*. Addition Publishing House.
63. Nimavat, K. K., & Kumar, R. (2025). *U.S. Patent No. 12,260,303*. Washington, DC: U.S. Patent and Trademark Office.
64. Sahoo, P. A. K., Aparna, R. A., Dehury, P. K., & Antaryami, E. (2024). Computational techniques for cancer detection and risk evaluation. Industrial Engineering, 53(3), 50-58.
65. Gurram, N. T., Narender, M., Bhardwaj, S., & Kalita, J. P. (2025). A Hybrid Framework for Smart Educational Governance Using AI, Blockchain, and Data-Driven Management Systems. *Advances in Consumer Research*, *2*(5).
66. Inbaraj, R., & Ravi, G. (2021). Multi Model Clustering Segmentation and Intensive Pragmatic Blossoms (Ipb) Classification Method based Medical Image Retrieval System. *Annals of the Romanian Society for Cell Biology*, *25*(3), 7841-7852.
67. Juneja, M., & Juneja, P. (2025). The Rise of The Tech-Business Translator in The Age Of AI. *International Research Journal of Advanced Engineering and Technology*, *2*(06), 05-15.
68. Jadhav, Y., Patil, V., & Parasar, D. (2020, February). Machine learning approach to classify birds on the basis of their sound. In *2020 International Conference on Inventive Computation Technologies (ICICT)* (pp. 69-73). IEEE.
69. Suman, P., Parasar, D., & Rathod, V. R. (2015, December). Seeded region growing segmentation on ultrasound image using particle swarm optimization. In *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)* (pp. 1-6). IEEE.
70. Akat, G. B. (2021). EFFECT OF ATOMIC NUMBER AND MASS ATTENUATION COEFFICIENT IN Ni-Mn FERRITE SYSTEM. *MATERIAL SCIENCE*, *20*(06).
71. Boopathy, D., & Balaji, P. (2023). Effect of different plyometric training volume on selected motor fitness components and performance enhancement of soccer players. *Ovidius University Annals, Series Physical Education and Sport/Science, Movement and Health*, *23*(2), 146-154.
72. Venkiteela, P. (2025). Real-Time Identity Federation: Replacing File-Based Sync with Okta APIs for GDPR-Compliant. *European Journal of Information Technologies and Computer Science*, *5*(5), 7-13.
73. Joshi, S., & Kumar, A. (2011). Correlation Filter based on Fingerprint Verification System. In *International Conference on VLSI, Communication and Instrumentation* (pp. 19-22).
74. Ganeshan, M. K., & Vethirajan, C. (2021). Trends and future of human resource management in the 21st century. *Review of Management, Accounting, and Business Studies*, *2*(1), 17-21.
75. Samal, D. A., Sharma, P., Naveen, S., Kumar, K., Kotehal, P. U., & Thirulogasundaram, V. P. (2024). Exploring the role of HR analytics in enhancing talent acquisition strategies. *South Eastern European Journal of Public Health*, *23*(3), 612-618.
76. Kamatchi, S., Preethi, S., Kumar, K. S., Reddy, D. N., & Karthick, S. (2025, May). Multi-Objective Genetic Algorithm Optimised Convolutional Neural Networks for Improved Pancreatic Cancer Detection. In *2025 3rd International Conference on Data Science and Information System (ICDSIS)* (pp. 1-7). IEEE.

 doi:10.20944/preprints202601.0513.v1

77. Gupta, I. A. K. Blockchain-Based Supply Chain Optimization For Eco-Entrepreneurs: Enhancing Transparency And Carbon Footprint Accountability. *International Journal of Environmental Sciences*, *11*(17s), 2025.
78. Praveen, R. V. S., Hundekari, S., Parida, P., Mittal, T., Sehgal, A., & Bhavana, M. (2025, February). Autonomous Vehicle Navigation Systems: Machine Learning for Real-Time Traffic Prediction. In *2025 International Conference on Computational, Communication and Information Technology (ICCCIT)* (pp. 809-813). IEEE.
79. Mohammed Nabi Anwarbasha, G. T., Chakrabarti, A., Bahrami, A., Venkatesan, V., Vikram, A. S. V., Subramanian, J., & Mahesh, V. (2023). Efficient finite element approach to four-variable power-law functionally graded plates. *Buildings*, *13*(10), 2577.
80. Juneja, M. (2025). Mentr: A Modular, On Demand Mentorship Platform for Personalized Learning and Guidance. *The American Journal of Engineering and Technology*, *7*(06), 144-152.
81. Polleri, A., Kumar, R., Bron, M. M., Chen, G., Agrawal, S., & Buchheim, R. S. (2022). *U.S. Patent Application No. 17/303,918*.
82. Sayyed, Z. (2025). Optimizing Callback Service Architecture for High-Throughput Applications. *International journal of data science and machine learning*, *5*(01), 257-279.
83. Vidyabharathi, D., Mohanraj, V., Kumar, J. S., & Suresh, Y. (2023). Achieving generalization of deep learning models in a quick way by adapting T-HTR learning rate scheduler. *Personal and Ubiquitous Computing*, *27*(3), 1335-1353.
84. Radhakrishnan, M., Sharma, S., Palaniappan, S., & Dahotre, N. B. (2024). Evolution of microstructures in laser additive manufactured HT-9 ferritic martensitic steel. *Materials Characterization*, *218*, 114551.
85. Chowdhury, P. (2025). GENERATIVE AI FOR MES OPTIMIZATION LLM-DRIVEN DIGITAL MANUFACTURING CONFIGURATION RECOMMENDATION. *International Journal of Applied Mathematics*, *38*(7s), 875-890.
86. Nasir, G., Chand, K., Azaz Ahmad Azad, Z. R., & Nazir, S. (2020). Optimization of Finger Millet and Carrot Pomace based fiber enriched biscuits using response surface methodology. *Journal of Food Science and Technology*, *57*(12), 4613-4626.
87. Akat, G. B., & Magare, B. K. (2022). Mixed Ligand Complex Formation of Copper (II) with Some Amino Acids and Metoprolol. *Asian Journal of Organic & Medicinal Chemistry*.
88. Thota, R., Potluri, S. M., Alzaidy, A. H. S., & Bhuvaneshwari, P. (2025, June). Knowledge Graph Construction-Based Semantic Web Application for Ontology Development. In *2025 International Conference on Intelligent Computing and Knowledge Extraction (ICICKE)* (pp. 1-6). IEEE.
89. RAJA, M. W., PUSHPAVALLI, D. M., BALAMURUGAN, D. M., & SARANYA, K. (2025). ENHANCED MED-CHAIN SECURITY FOR PROTECTING DIABETIC HEALTHCARE DATA IN DECENTRALIZED HEALTHCARE ENVIRONMENT BASED ON ADVANCED CRYPTO AUTHENTICATION POLICY. *TPM–Testing, Psychometrics, Methodology in Applied Psychology*, *32*(S4 (2025): Posted 17 July), 241-255.
90. Venkiteela, P. (2025). A Vendor-Agnostic Multi-Cloud Integration Framework Using Boomi and SAP BTP. *Journal of Engineering Research and Sciences*, *4*(12), 1-14.
91. Akat, G. B. (2023). Structural Analysis of Ni1-xZnxFe2O4 Ferrite System. *MATERIAL SCIENCE*, *22*(05).
92. Sivakumar, S., Prakash, R., Srividhya, S., & Vikram, A. V. (2023). A novel analytical evaluation of the laboratory-measured mechanical properties of lightweight concrete. *Structural engineering and mechanics: An international journal*, *87*(3), 221-229.
93. Inbaraj, R., John, Y. M., Murugan, K., & Vijayalakshmi, V. (2025). Enhancing medical image classification with cross-dimensional transfer learning using deep learning. *1*, *10*(4), 389.
94. Chand, K., Singh, A., & Kulshrestha, M. (2012). Jaggery quality effected by hilly climatic conditions. *Indian Journal of Traditional Knowledge*, *11*(1), 172-176.
95. ROBERTS, T. U., Polleri, A., Kumar, R., Chacko, R. J., Stanesby, J., & Yordy, K. (2023). *U.S. Patent No. 11,775,843*. Washington, DC: U.S. Patent and Trademark Office.

 doi:10.20944/preprints202601.0513.v1

96. Reddy, D. N., Venkateswararao, P., Vani, M. S., Pranathi, V., & Patil, A. (2025). HybridPPI: A Hybrid Machine Learning Framework for Protein-Protein Interaction Prediction. *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)*, *13*(2).
97. Balakumar, B., & Raviraj, P. (2015). Automated Detection of Gray Matter in Mri Brain Tumor Segmentation and Deep Brain Structures Based Segmentation Methodology. *Middle-East Journal of Scientific Research*, *23*(6), 1023-1029.
98. Praveen, R. V. S., Raju, A., Anjana, P., & Shibi, B. (2024, October). IoT and ML for Real-Time Vehicle Accident Detection Using Adaptive Random Forest. In *2024 Global Conference on Communications and Information Technologies (GCCIT)* (pp. 1-5). IEEE.