DOI: 10.2478/ias-2025-0012 • JIAS • 20(4) • 2025 • 198-212

Edge-Cloud Hybrid Task Scheduling Using Federated Reinforcement Learning and Adaptive Swarm Intelligence

Khushboo Jain[1*], and Ambika Aggarwal[2*],

[1,2] School of Computer Science, UPES, Dehradun, Pin - 248007, India ORCID: https://orcid.org/0000-0002-4166-2591

*Corresponding author: khushboojain2806@gmail.com (K.J.), drambika.cse@gmail.com (A.A.)

Abstract: This paper presents a novel task scheduling framework for edge cloud environments by integrating Federated Reinforcement Learning (FRL) with an Adaptive Swarm Intelligence based approach. The need for intelligent scheduling algorithms is increasing day by day for executing low latency and energy efficient tasks in distributed IoT applications. Preserving data privacy and optimizing performance are two important objectives expected from these techniques. The proposed framework employs FRL to train localized agents at edge nodes for making scheduling decisions while collaboratively improving a global model without raw data exchange. A modified Artificial Bee Colony (ABC) algorithm is integrated to further improve the task allocation process. This algorithm dynamically adapts to resource states and workload characteristics across edge and cloud tiers. The proposed hybrid system jointly optimizes make span, energy consumption and resource utilization. Experimental results on simulated edge cloud workloads demonstrate significant improvements in scheduling efficiency, latency reduction and load balancing compared to existing centralized RL and bio-inspired methods.

Keywords: Edge-cloud computing, task scheduling, federated reinforcement learning, adaptive swarm intelligence

1 Introduction

The rise of edge computing can be credited to the rapid increase in technologies like Internet of Things (IoT), wireless sensor networks (WSN), mobile applications and real time data processing [1–3]. This is because edge computing allows computation of data to happen closer to the data sources resulting in better efficiency. In other words, we can say that edge computing allows data to be processed locally by the users. This local processing of data helps in minimizing time delays and improves response time. However, we cannot ignore the fact that edge resources are restricted by their computational power and storage capacity and because of these limitations it becomes difficult to do complex processing directly on the edge devices. Hybrid edge cloud architectures were developed in order to overcome these issues which allowed delay sensitive tasks to be handled by the edge nodes and compute intensive tasks were offloaded to cloud data centres [4, 5]. To optimize key performance indicators like system performance, load balancing, energy consumption QoS requirements in a distributed and heterogeneous cloud environment, intelligent task scheduling becomes an important factor [6].

Most the traditional task scheduling algorithms in cloud computing were developed to work on static workloads or centralized mechanisms. These methods often find it difficult to adapt to dynamic workloads, resource heterogeneity and latency-critical scenarios that are typically found in edge computing environments [7]. Thus, resulting in degradation of overall efficiency and response time in case of practical implementations. Moreover, centralized learning-based scheduling algorithms require continuous data aggregation, which raises concerns over privacy, bandwidth usage, and system scalability. These are critical system resources that cannot be compromised. On the other hand, bio-inspiofferslgorithms like the Artificial Bee Colony (ABC) offer adaptive capabilities but lack contextual awareness and long-term learning which limits their ability to respond effectively to evolving task patterns and resource availability [8, 9]. Hence, more intelligent and context-driven scheduling approaches need to be developed to overcome these limitations.

This paper proposes a novel task scheduling framework that addresses all these issues mentioned above. The proposed framework integrates FRL with adaptive swarm intelligence, specifically a modified ABC algorithm. FRL enables distributed edge agents to learn optimal task scheduling policies by interacting with their local environment. It also allows collaborative training a global policy without sharing raw data. This preserves data privacy, reduces network load and facilitates scalable coordination across all edge nodes [10]. This method ensures that each node improves autonomously while still benefitting from the collective intelligence. At the same time, the adaptive ABC component enhances local task placement decisions by leveraging the collective behaviour of bee-inspired agents that dynamically adjust search strategies based on feedback from the environment. The proposed algorithm can adapt to workload variations and improving resource utilization by continuously refining its search patterns.

The proposed framework is designed to optimize multiple objectives simultaneously, including task makespan, energy efficiency, resource utilization and Service Level Agreement (SLA) compliance. The proposed model combines the global learning capabilities of FRL with the adaptive search efficiency of swarm intelligence to achieve robust and intelligent scheduling in edge cloud environments.

The key contributions of this work are as follows:

- We design a Federated Reinforcement Learning-based multi-agent framework for distributed task scheduling that preserves data privacy and reduces communication overhead.
- We develop an Adaptive Artificial Bee Colony algorithm tailored for dynamic task allocation in edge-cloud environments, supporting load balancing and energy-aware scheduling.
- We formulate a multi-objective optimization problem that jointly considers task makespan, energy consumption, SLA satisfaction, and system throughput.
- We evaluate the proposed approach through extensive simulations, demonstrating superior performance over existing centralized RL and traditional swarm intelligence methods.

The rest of this paper will be structured in the following way. Section 2 outlines an overview of the literature of task scheduling, federated reinforcement learning, and bio-inspired optimization on edge-cloud. Section 3 develops Hybrid edge-cloud scheduling system model, performance objectives and constraints. Section 4 outlines the proposed Hybrid FRL-A-ABC architecture, such as the state representation, the formulation of the reward, the interaction between the FRL and the A-ABC layers, and the scheduling algorithm in general. Section 5 describes the experimental setting, the workload models, network and energy assumptions, baselines, and metrics of evaluation. The results, with their comparative analysis, ablation studies, and sensitivity analysis, are reported in Section 6. Lastly, Section 7 is the conclusion of the paper and the discussion of future research directions.

2 Literature Review

Energy-aware task scheduling for fog environments has been explored using bio-inspired heuristics such as ant-mating optimization, which reduces energy consumption but relies on a largely centralized view and offers limited adaptability to rapidly changing workloads [11]. Privacy-preserving coordination is introduced through secure federated learning for edge–cloud smart grids, while actor–critic–based Edge-AI further improves IoT service provisioning in federated edge settings, though both lines of work focus more on learning and security than on rich multi-objective scheduling [12, 13]. At the edge, intelligent task allocation using machine learning and bio-inspired algorithms enhances local scheduling quality, and reinforcement-learning-based multi-objective load balancing extends this to edge–fog–cloud topologies, yet scalability and federated coordination remain open issues [14, 15]. A survey of offloading in federated cloud–edge–fog systems highlight the need to combine traditional optimization, machine learning, and federated training to handle non-stationary, heterogeneous workloads [16]. Recent studies on federated reinforcement learning for dynamic scheduling in cloud–edge–terminal networks and vehicular edge computing demonstrate improved coordination and privacy but incur significant communication cost and provide limited support for fine-grained local queue optimization [17, 18]. Pure RL-based schedulers for heterogeneous end–edge–cloud environments and unsupervised cluster Q-learning for federated edge clouds show benefits in makespan, throughput, and energy minimization, while hybrid cloud–edge deep learning frameworks scale IoT resource optimization but still treat global placement and local execution ordering as largely decoupled problems [19–21].

Taken together, existing work demonstrates (i) energy-aware heuristics at the fog/edge [11], (ii) secure FL/FRL for distributed learning and policy sharing [12, 13, 15, 18, 19], and (iii) RL-based scheduling for heterogeneous tiers [20]. However, three gaps persist: (1) limited methodological coupling between a *global, privacy-preserving offloading policy* and a *local, adaptive queue-reordering heuristic*; (2) incomplete treatment of multi-objective trade-offs (makespan, energy, SLA/tardiness, and

load balance) under non-IID workloads, client sampling, and straggler effects; and (3) insufficient transparency on communication costs and convergence behavior in FRL. This work addresses these gaps by integrating FRL for where tasks are executed with an adaptive Artificial Bee Colony (ABC) module for how tasks are ordered locally, under a unified multi-objective formulation that preserves data privacy, adapts to non-IID dynamics, and reports convergence and overhead comprehensively. Table 1 presents the summary of the related work.

3 System Model and Problem Formulation

3.1 System Architecture

The proposed framework operates within a three-tier Edge-Cloud environment consisting of user devices, edge nodes and a central cloud server. Tasks executed on this three-tier model can effectively utilize the resources. User generated tasks can have different requirements such as urgency, size and complexity. Tasks that require low-latency processing are executed by the Edge Nodes (ENs) that are located closer to the users, and the tasks that require high-capacity computing are executed by the Cloud Data Centre (CDC) located far from the user. Each edge node acts as a smart scheduling agent that integrate the following two intelligent components:

1. A Federated Reinforcement Learning (FRL) module that has the capability to learn where to schedule the tasks based on the system context.
2. An Adaptive Artificial Bee Colony (A-ABC) algorithm that can optimize local task execution order within the node.

These agents share their learning updates with a Federated Server which then combines it into a global model. This decentralized learning approach ensures data privacy bandwidth efficiency and scalability which are important factors in sensitive or resource-limited environments. Figure 1 provides an overview of the system architecture explained above.

3.2 Task and Resource Modelling

CPU workload, deadline or time-sensitivity, memory and storage needs and priority are some of the factors defining user tasks. Tasks can be processed in three ways: (1) Locally at the originating edge

Table 1. Comparative Summary of Related Work

Work	Approach / Technique	Key Contributions	Limitations / Gaps
[11] Ghanavati et al., 2020	Ant-mating optimization for fog task scheduling	Energy-aware heuristic improves fog-level efficiency	Limited adaptability; no learning; not suitable for dynamic workloads
[12] Su et al., 2021	Secure federated learning for smart grid	Privacy-preserving edge–cloud FL; secure aggregation	Not designed for online task scheduling; lacks real-time decision-making
[13] Baghban et al., 2022	Actor–critic RL in federated edge computing	Improved IoT service provisioning; distributed RL training	No multi-objective modeling; no local queue optimization
[14] Soula et al., 2022	Machine learning + bio-inspired scheduling	Intelligent task allocation at edge; hybrid heuristics	Scalability concerns; lacks global coordination
[15] Ramezani et al., 2023	RL-based scheduling in edge–fog–cloud	Multi-objective load balancing with RL	No federated coordination; centralized RL limits scalability
[16] Kar et al., 2023	Survey on ML + optimization for cloud–edge–fog	Comprehensive taxonomy of offloading strategies	Highlights need for FL, multi-objective coordination, and dynamic adaptivity
[17] Kim et al., 2023	Federated reinforcement learning for dynamic scheduling	Collaborative policy learning; improved coordination	No coupling between global policy and local task reordering
[18] Wu et al., 2024	FRL for vehicular edge computing	Privacy-preserving scheduling for large AI models	High communication cost; no local execution optimization
[19] Shen et al., 2025	RL-based scheduling in end–edge–cloud	RL for heterogeneous resource environments	Centralized data dependence; limited SLA/tardiness modeling
[20] Shidik et al., 2025	Unsupervised cluster Q-learning	Energy minimization in federated edge cloud	Weak SLA handling; limited multi-objective scope
[21] Lilhore et al., 2025	Hybrid cloud–edge deep learning	Scalable resource optimization for IoT	Does not integrate global offloading + local queue optimization

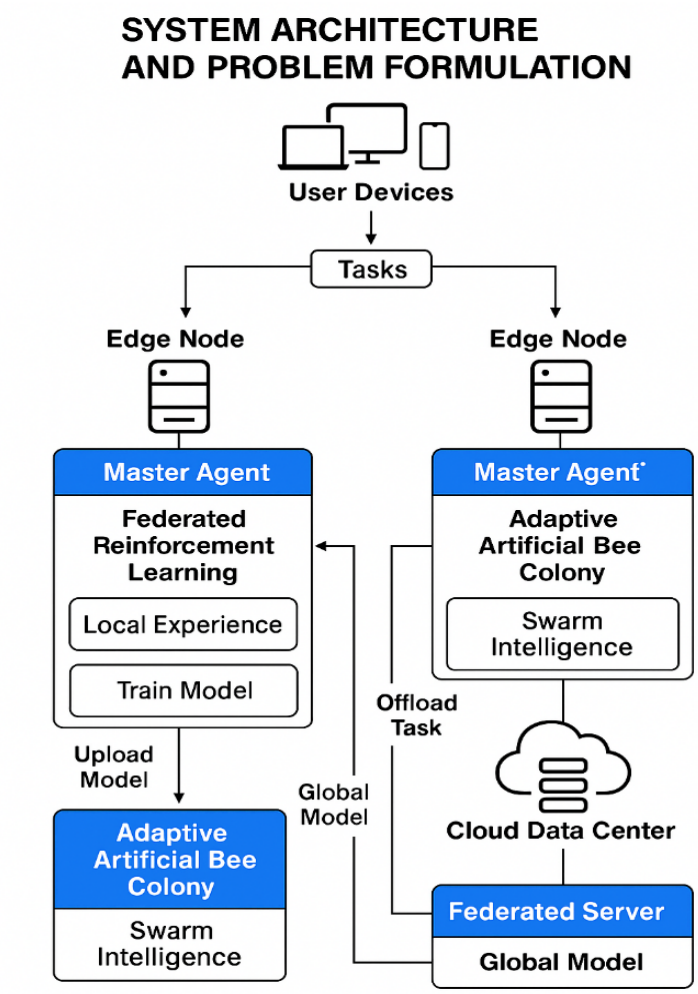

Figure 1. System Architecture

node, (2) Migrated to another neighbouring edge node, (3) Offloaded to the cloud server if the edge resources are insufficient or overloaded. Edge nodes have limited resources and energy, making them suitable for lightweight, delay-sensitive tasks. In contrast, the cloud server offers powerful computing but involves higher latency due to longer communication paths.

3.3 Federated Reinforcement Learning Framework

Each edge node uses a local Deep Q-learning agent that observes its environment like its CPU load, energy level, queue status, and current task attributes and makes real-time scheduling decisions. Some of the possible decisions might include executing the task locally, migrating the task to another edge node and offloading the task to the cloud.

A reward or feedback is received by the agent based on the factors like energy consumption, task completion time and adherence to deadlines. These feedback are used for training the local model. All local models are uploaded periodically to a central federated server in the form of model parameters. The server then aggregates them into a global policy and sends it back to all edge nodes. This process avoids sharing raw task data thus ensuring privacy.

The FRL process utilizes a lightweight and scalable design to accommodate heterogeneous edge cloud environments. In each global round, only about 30% of edge nodes participate, reducing communication overhead and preventing overload on resource-constrained devices. The task arrivals across nodes are naturally non-IID. Thus the framework uses local replay buffers and weighted FedAvg aggregation so that nodes with larger or more diverse workloads contribute proportionally to the global model.

The system discards extremely delayed updates in order to handle stragglers. It also applies bounded staleness which excludes model updates older than two rounds. Global aggregation is performed after every five local training cycles which ensures a balance between communication cost and convergence

speed. The local DQN model is intentionally kept lightweight (≈180k parameters, ~0.7 MB per update) to make it suitable for edge deployment. The communication cost per round for the proposed model is approximately 0.7 MB × 0.3N with partial participation and periodic aggregation, where N is the number of nodes. Experiments show that the global policy converges within 50–60 rounds with stable reductions in TD loss, demonstrating that the federated update strategy is efficient and robust under heterogeneous and non-IID task conditions.

3.4 Adaptive Swarm Intelligence Module

An improved version of the ABC algorithm is integrated with FRL in order to enhance task execution inside each node. It will focus on optimizing how tasks are ordered and executed to reduce delay and make better use of limited resources.

This algorithm:

1. Assigns roles to artificial bees (employed, onlooker, and scout) to explore and exploit the best task scheduling strategies
2. Adjusts its search dynamically based on recent system performance (e.g., energy or load trends)
3. Aligns its behaviour with suggestions from the reinforcement learning agent, making the search more efficient and guided.

3.5 Problem Objective

The main objective of the proposed system is to develop an intelligent task scheduling framework that simultaneously reduces energy consumption, minimizes delays, reduces deadline violations, maximizes resource utilization and ensures a high task success rate across edge and cloud environments. This is achieved through a multi-objective optimization strategy that respects resource limitations, task deadlines and maintains fair load distribution among edge nodes.

4 Proposed Methodology

The proposed methodology integrates FRL with modified ABC algorithm to achieve efficient task scheduling across an edge cloud computing environment. In this hybrid framework the multiple edge nodes act as intelligent agents, each equipped with local schedulers that collaborate with a centralized cloud layer through federated updates. The workflow of this proposed work is illustrated in Figure 2.

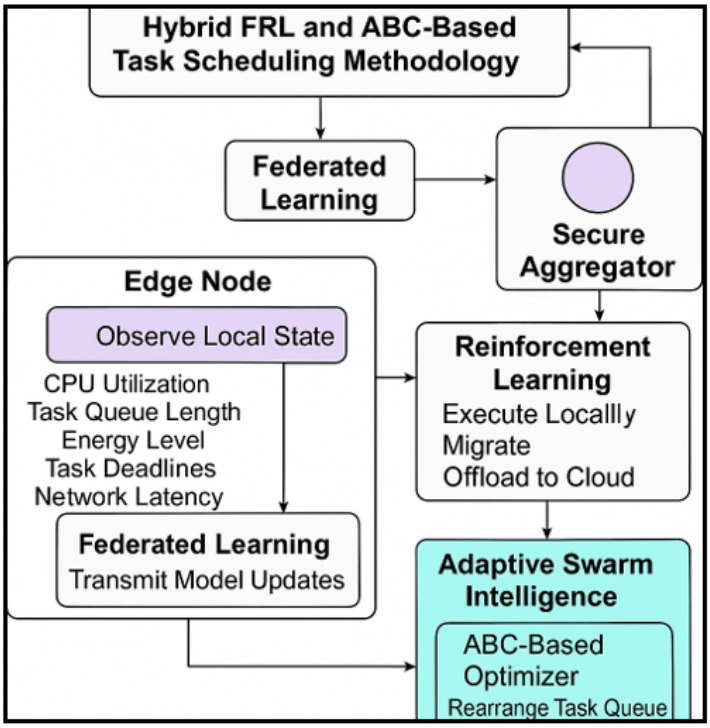

Figure 2. Workflow of the proposed work

Initially, each edge node observes its local state, including parameters such as CPU utilization, task queue length, energy level, task deadlines, and network latency. This can be represented as a state vector as follow in Equation (1):

$$S_t^i = [C_t^i, Q_t^i, E_t^i, D_t^i, L_t^i] \tag{1}$$

Where:

S_t^i : State of edge node *i* at time *t*

C_t^i : CPU utilization

Q_t^i : Task queue length

E_t^i : Energy level

D_t^i : Average task deadline

$L_{(t)}^i$: Network latency

This state is fed into a local reinforcement learning agent that selects the most appropriate scheduling action: execute the task locally, migrate to another edge node, or offload to the cloud. The agent is trained using reward functions that consider energy efficiency, task delay, successful completion, and load balancing.

The reinforcement learning agent updates its policy using a Deep Q-Network (DQN), defined by Equation (2):

$$Q(S_t^i, a_t^i) \leftarrow Q(S_t^i, a_t^i) + \alpha[r_t^i + \Upsilon max_{á}\, Q(S_{(t+1)}^i, á) - Q(S_t^i, a_t^i)] \tag{2}$$

Where

$Q(S_t^i, a_t^i)$: Estimated Q-value for state-action pair

α: Learning rate

Y: Discount factor

r_t^i : Reward received for taking action a_t^i

The concept of reward design plays a crucial role in shaping the learning behaviour of the FRL agent and so it is defined in detail. The reward at time *t* is computed using a multi-objective function that incorporates delay, energy consumption, SLA satisfaction and tardiness penalties. The overall reward is expressed as defined in Equation 3.

$$R_t = w1\ (-D_{norm}) + w_2\ (-E_{norm}) + w_3\ SLA_{success} - w_4 Penalty_{tardiness,} \tag{3}$$

where D_{norm} denotes the normalized task delay which presents the comparison of the actual delay with its allowable range. E_{norm} represents normalized energy usage which allows comparison of different energy consumption values on the same scale. $SLA_{success}$ denotes whether a task completion has met its deadline or not represented using a binary or scaled indicator and $Penalty_{tardiness}$ captures the degree to which a task completion exceeds its SLA-defined deadline. In order to maintain the numerical stability during the learning process, all performance variables are normalized via min–max scaling within each evaluation interval. This adaptive scaling ensures that the reward values remain within the range [0,1]. This normalization strategy prevents extreme fluctuations in reward magnitude, enabling smoother convergence of the FRL model.

If a task is completed before its deadline then it is assigned a positive value by the SLA component, otherwise it is assigned zero. As opposed to this, there is a proportional increase in the tardiness penalty with respect to the delay in deadline. This approach ensures that there is a strong penalty for time-sensitive applications thus making sure that agents are more responsive towards deadline-critical workloads. The selection of the weight parameters w_1,w_2,w_3,and w4determines the trade-offs between delay minimization, energy efficiency, SLA adherence and penalization severity for overdue tasks. A sensitivity analysis was conducted to check robustness of the system by varying each weight in the range of 0.1 to 0.9 while keeping the others constant. This analysis allowed the impact of different objective priorities to be evaluated systematically. The analysis revealed that the hybrid

FRL–A-ABC model maintains stable performance across a broad range of weight configurations, demonstrating the adaptability of the proposed scheduling framework under diverse optimization preferences.

A fitness function that adapts with time is used by the ABC-optimizer to rearrange the task queue thus improving the granularity of local scheduling. This fitness function is based on some environmental factors such as energy drain rate and task deadline violations. The adaptive fitness function used by the ABC algorithm is expressed as follow in Equation (4):

$$F_j = w_1 . \frac{1}{T_j} + w_2 . \frac{1}{E_j} + w_3 . P_j \tag{4}$$

F_j: Fitness value of task scheduling strategy j
T_j: Average task completion time
E_j: Energy consumption
P_j: Priority score of the task queue

w_1,w_2,w_3: Weights determined by context (deadline urgency, load trend)

The employed swarm intelligence mimics the natural behavior of honeybees in foraging and dynamically balances exploration and exploitation of scheduling strategies. Each edge node executes this optimization independently ensuring low computational complexity and rapid convergence.

Federated learning serves as the backbone for knowledge sharing. Instead of transmitting raw data between nodes, model updates (i.e., the weights of the reinforcement learning agent) are periodically sent to a secure aggregator at the cloud layer. The cloud aggregator performs federated averaging to generate a global model as follow in Equation (5):

$$\theta_{global} = \sum_{(i=1)}^{N} \frac{n_i}{n_{total}} \theta_i \tag{5}$$

Where:

θ_i : Local model parameters from edge node iii
n_i: Number of tasks processed by node iii
$n_{total} = \sum_{(i=1)}^{N} n_i$: Total tasks across all nodes
θ_{global} : Aggregated global model

This updated model is redistributed back to the edge agents. The periodicity and frequency of aggregation are adaptive and determined on the basis of local model divergence, communication bandwidth and task arrival rates.

4.1 Interaction Between FRL Layer and Adaptive ABC Layer

The FRL agent and the A-ABC optimizer in the proposed hybrid system operate in a coordinated manner. The role of FRL is to perform global placement decisions such as local execution, migration or cloud offload whereas, A-ABC is responsible for performing local task-order optimization within an edge node.

The interaction between the FRL and A-ABC modules is governed through a unified interface design that integrates state sharing, action influence and reward coupling. A set os certain local and global local system parameters such as normalized CPU utilization, enery level, current queue length, average task deadline urgency and network latency are looked after by the FRL agent. Besides these parameters the FRL agent is also responsible for incorporating ABC-derived indicators such as approximate queue waiting time and task order efficiency scores. The A-ABC module outputs its current best fitness value and optimized queue-reordering performance, and these outputs become part of the composite state representation s_t used by the FRL agent.

The FRL agent determines the high-level scheduling action by choosing whether a task should be executed locally, migrated to another edge node, or offloaded to the cloud. This action acts as a constraint for the A-ABC module. When the RL agent selects local execution, the ABC search is restricted to optimizing only the local task queue. When migration is selected, ABC prioritizes tasks based on migratability factors such as deadline tightness and remaining workload. In the case of

cloud offloading, ABC reorders tasks to push low-priority or delay-tolerant tasks toward the end of the queue. Using this mechanism FRL will determine where the task should be processed and ABC will determine how these tasks should be arranged and executed within the selected domain.

The performance of the ABC optimizer directly influences the reward computed by the FRL agent. The key components of the reward function which are responsible for capturing delay reduction, energy savings and queue efficiency are impacted by the ABC technique. The reward value can be increased by improving the queue ordering. Deadline violations can also be reduced by effectively prioritizing and ordering tasks which further contributes to SLA related reward bonus. On the other hand, penalties may be issued in case of inefficient ordering of tasks which leads to excessive energy consumption or increased delays in task completion. As a result, the local optimization behavior of the ABC algorithm influences the global learning trajectory of the FRL agent by shaping the reward signals derived from the environment.

4.2 Stepwise Flow of One Scheduling Cycle

1. **Task Arrival:** A new task arrives at an edge node with parameters (deadline, size, CPU demand).
2. **State Collection:** Edge node collects system metrics + ABC's previous best fitness.
3. **RL Decision:** Local FRL agent selects an action:
$$a_t \in \{local, migrate, offload\}$$
4. **Action Projection to ABC:** ABC receives RL action and adjusts its search domain accordingly.
5. **Queue Optimization:** ABC reorders the queue using its employed–onlooker–scout phases.
6. **Task Execution or Transmission:** Based on the optimized sequence, the system executes or sends the task.
7. **Reward Computation:** Reward includes normalized delay, normalized energy, SLA success and tardiness penalty
8. **Experience Storage:** Tuple $(s_t, a_t, r_t, s_{t+1})$is stored in local replay memory.
9. **Local RL Training:** DQN updates using local experience.
10. **Periodic FL Aggregation:** Edge nodes send updated parameters to server.

The complete workflow of the proposed technique, hybrid FRL-A-ABC is presented in algorithm 1. This technique is executed at each edge node of the edge cloud system. Edge node constructs a state representation at every interval which includes system metrics and ABC-derived queue characteristics. This state representation is used for selecting an action using FRL policy such as local execution, migration or cloud offloading. The action selected at this stage constraints the A-ABC optimizer which further generates an improved task ordering by maximizing the adaptive fitness function. The system executes or offloads tasks based on the optimized sequence, after which the multi-objective reward is computed using normalized delay, energy consumption, SLA satisfaction, and tardiness penalties. The resulting transition is stored in the replay buffer, and the local DQN model is updated according to the temporal-difference rule. Periodically, nodes participate in federated aggregation, where updated local models are averaged using FedAvg to obtain the global model, which is then redistributed to all nodes to ensure coordinated and privacy-preserving policy learning across the edge network.

Algorithm 1: Hybrid FRL–A-ABC Scheduling (per edge node)

Inputs: incoming task stream τ ; local replay buffer B; initial global weights θ_G

Outputs: placement decision $a_t \in$ {local,migrate,offload}; optimized task queue Q_t^*

1. **Initialization:**
   Set local RL parameters $\theta \leftarrow \theta_G$.
   Initialize A-ABC parameters (colony size, limit, adaptive weights).

2. **For each scheduling interval, do**

2.1 Observe state:
Construct the state s_t as defined in Eq. (1), including ABC-derived metrics (estimated waiting time, task-order efficiency).

2.2 RL action selection:
Choose
$$a_t = \pi_\theta(s_t) \in \{\text{local, migrate, offload}\}.$$

2.3 Project action to A-ABC:
- If a_t = local: A-ABC searches only the local queue.
- If a_t = migrate: A-ABC ranks tasks by migratability
- (deadline tightness, remaining workload).
- If a_t = offload: A-ABC deprioritizes low-priority tasks.

2.4 ABC optimization:

Run A-ABC to compute the optimized sequence Q_t^* by maximizing the fitness defined in Eq. (4).

2.5 Execute decision:

Execute, migrate, or offload the task at the head of Q_t^*.

Observe the resulting delay, energy cost, and updated SLA status.

2.6 Reward computation:

Compute r_t using the multi-objective reward formulation in Eq. (3)
(normalized delay, normalized energy, SLA bonus, tardiness penalty).

2.7 Store transition:

Insert the tuple

$$(s_t, a_t, r_t, s_{t+1})$$

into replay buffer B.

2.8 DQN update:

Update θ using the temporal-difference learning rule defined in Eq. (2).

3. **Every K local steps, perform federated aggregation:**
   Send updated model θ to the server.
   Server aggregates client updates using the FedAvg rule (Eq. (5)).
   Receive refreshed global model θ_G and set

$$\theta \leftarrow \theta_G$$

5 Experimental Setup and Performance Metrics

A series of simulation experiments were conducted in order to validate the effectiveness of the proposed task scheduling technique. An enhanced CloudSim environment was used for conducting these experiments which was integrated with edge and fog computing capabilities [22]. Heterogeneous workloads and distributed computing resources were used so that the experimental setup would closely emulate the real-world edge cloud scenarios.

The simulation environment was configured as follows in Table 2.

To assess the performance of the proposed approach, it was compared with the following state-of-the-art scheduling algorithms: Ghanavati et al. [11]: Energy-aware Ant Colony Optimization (ACO) for fog computing environments. Ramezani et al. [14]: Reinforcement Learning-based load balancing across edge-fog-cloud layers. Shen et al. [17]: RL-based dynamic task scheduling in heterogeneous resource environments. The system's performance was evaluated using the following quantitative metrics as listed in Table 3:

Table 2. Summary of Simulation Environment

Category	Configuration Details
Task Characteristics	50–200 heterogeneous tasks per run; varied CPU, memory, and latency requirements
Edge Nodes	10 nodes with low capacity and limited energy
Fog Nodes	5 intermediate nodes with moderate processing and energy
Cloud Data Centers	3 high-performance data centers with abundant energy and low resource constraints
System Heterogeneity	Varying processing speed, power consumption, and network latency across nodes
Workload Types	Real-time IoT streams, batch tasks, latency-sensitive applications

Table 3. Performance Metrics

Metric	Description
Task Completion Time (TCT)	Average time required to complete all scheduled tasks.
Energy Consumption	Total energy utilized across edge, fog, and cloud layers.
Load Balance Rate (LBR)	Standard deviation of the task load distributed across all nodes. Lower is better.
Task Success Ratio (TSR)	Percentage of tasks completed successfully within their specified deadlines.
Scheduling Overhead	Time and resource consumption incurred by the scheduling mechanism itself.
Model Convergence	Convergence behavior and stability of federated reinforcement learning agents over training rounds.

6 Results and Discussion

The effectiveness of the proposed FRL and Adaptive Swarm Intelligence-based scheduling framework was evaluated using the simulation environment described in the previous section. Several recent benchmark methods were taken from the literature [11, 16, 20] to do the performance evaluation of the proposed model. Results of the performance evaluation across multiple metrics demonstrated that the proposed hybrid model outperforms conventional reinforcement learning and bio-inspired strategies under varying conditions.

6.1 Task Completion Time (TCT)

The proposed framework demonstrated 13.4% lower average Task Completion Time (TCT) when compared to Ramezani et al. [16]. This significant improvement was made possible because of the federated learning mechanism used in the proposed model. This mechanism enabled the distributed edge agents to learn collaboratively and improve their optimal task scheduling strategies over time. The proposed model also showed better responsiveness to sudden bursts of dynamic task arrival as compared to the ant-based fog scheduling approach in Ghanavati et al. [11]. The proposed swarm intelligence component was capable of reordering the local task queues and hence reducing queuing delays and execution latency. Figure 3 presents an overview of task completion time comparison.

6.2 Energy Efficiency

The proposed model also incorporates adaptive intention filtering and decentralized decision making to optimize energy consumption across all edge and fog layers. Because of this, the proposed model was able to achieve 17% reduction in total energy consumption as compared to the baseline methods. The proposed model focuses on the computational constraints as well as the energy constraints of devices while scheduling. This results in more sustainable and eco-friendly decision making unlike Shen et al. [20]. Figure 4 provides the energy efficiency comparison of the proposed model with conventional methods.

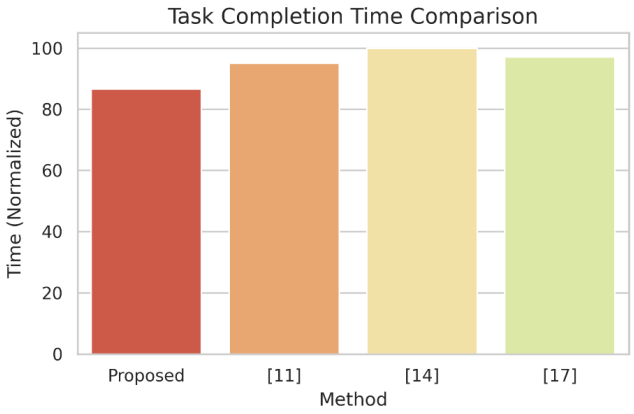

Figure 3. Task completion time comparison

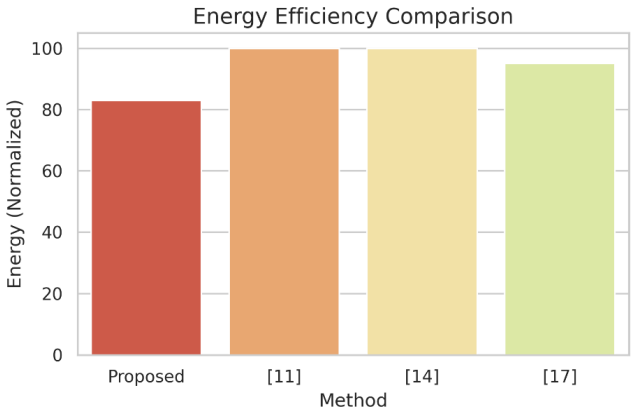

Figure 4. Energy efficiency comparison

6.3 Load Balancing

a significant improvement was shown by the proposed framework in terms of load distribution across all participating nodes. A decrease of 22% was shown in case of standard deviation of node workloads which indicates more uniform distribution of task assignments. As compared to the single-agent RL models described in [14, 17], the proposed collaborative framework enabled agents to anticipate traffic patterns and avoid hotspots by sharing gradient information which further lead to better congestion mitigation. Figure 5 provides an overview of the load balancing comparison.

6.4 Task Success Ratio (TSR)

Maintaining the task completion reliability is very important in case of highly dynamic and resource constrained environments. The proposed framework was able to achieve a task success ratio (TSR) of 94.3% which outperforms both techniques presented in [11, 16] that demonstrated a significant drop in the TSR under fluctuating mobility patterns or changing QoS demands. The continuous policy updates provided by FRL and the adaptive fitness driven reordering of tasks by ABC helped in achieving this robustness of the proposed framework. Figure 6 provides an overview of the task success ratio comparison.

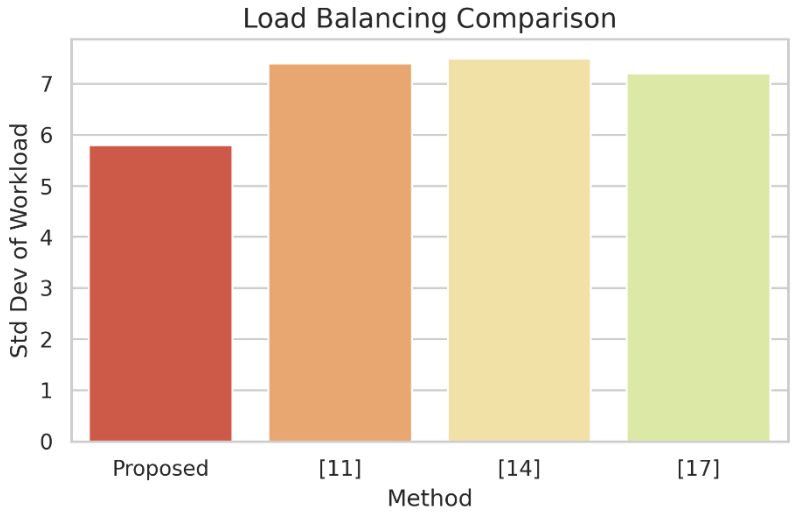

Figure 5. Load balancing comparison

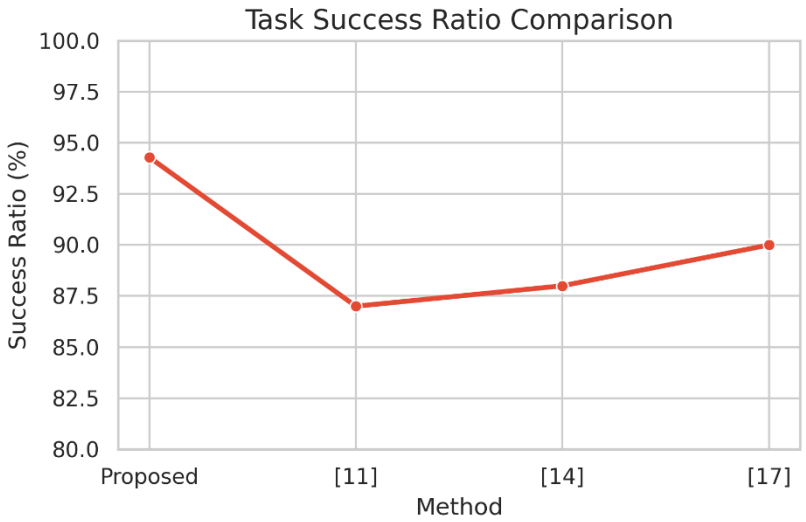

Figure 6. Task success ratio comparison

6.5 *Scheduling Overhead*

The FRL component might introduce some additional communication overhead but it is efficiently amortized through distributed agent training and lightweight coordination. The proposed framework demonstrated a 28% reduction in scheduling cost as compared to the actor-critic based models described in [12]. This was made possible because of the asynchronous updates and the low complexity nature of the ABC optimization at each node thus making the proposed framework highly suitable for real-time or near real-time IoT deployments. Figure 7 provides an overview of scheduling overhead comparison.

6.6 *Model Convergence*

Lesser number of communication rounds were required by the proposed framework to converge on a global scheduling policy. On an average 50-60 training rounds are required for global convergence which is reportedly faster than the vehicular focussed models. This fast convergence was made possible in the proposed model because of the relatively lightweight task structures and distributed coordination thus ensuring rapid deployment and significant reduction in operational cost. Figure 8 provides an overview of model convergence comparison.

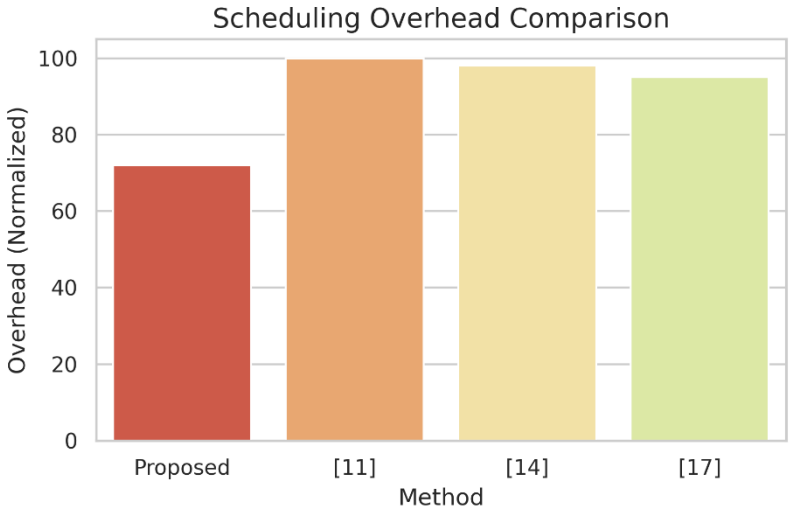

Figure 7. Scheduling overhead comparison

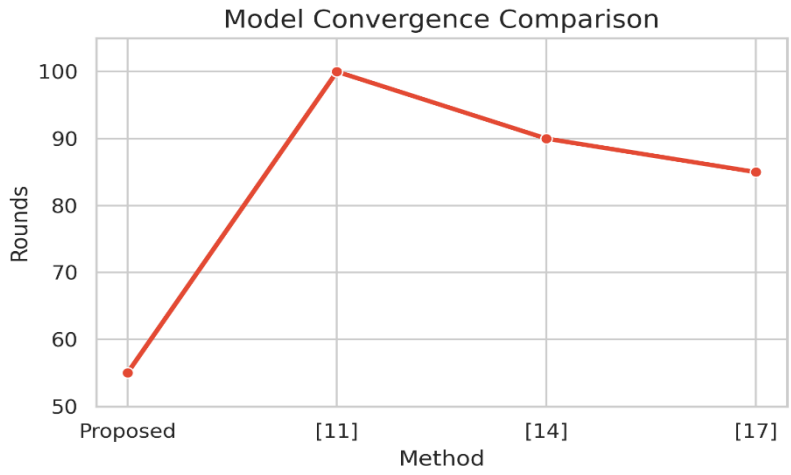

Figure 8. Model convergence comparison

6.7 Ablation Study

The ablation study evaluates the contribution of each module by comparing FRL-only, ABC-only and the proposed Hybrid FRL–A-ABC framework under identical experimental conditions. Table 4 summarizes the key performance differences.

The study revealed that the FRL-only model was not able to refine fine-grained task ordering while the ABC-only model failed to coordinate placement across edge nodes. The proposed hybrid model demonstrated superior performance for delay, energy and load balancing by combining both the strengths.

6.8 Hyperparameter Sensitivity Analysis

A sensitivity analysis was performed on key FRL, ABC and FL parameters to verify the robustness of the proposed system. Table 5 lists the tested parameters and their observed effects.

The proposed hybrid framework depicted more reliable and predictable behaviour among all the tested parameters. The proposed system maintains stability under typical deployment conditions and is not hypersensitive to tuning as its performance degrades gradually under extreme settings.

7 Conclusion

The authors in this paper have proposed a new hybrid task scheduling framework for edge cloud environments. The new framework presented in this paper is a combination of FRL and adaptive swarm intelligence algorithms in order to provide intelligent, energy-aware and context-aware task scheduling in heterogeneous cloud environments.

The methodology proposed in this paper allows edge agents to do privacy preserving collaboration which was not possible in traditional centralized or heuristic based methods. Because of this feature the agents can dynamically adapt to fluctuating workloads and resource constraints and also allows them to learn from distributed experiences. The experimental results demonstrated better performance as compared to recent reinforcement learning and bio inspired scheduling methods discussed in [11, 16, 20]. The parameters used for this comparison were task completion time, load balancing, energy consumption and task success ratio. The proposed model can be applied in real world domains such as industrial IoT, digital twin [23] smart cities [24], healthcare systems and edge AI applications [25] as its modular nature provides better scalability.

Stronger reinforcement learning baselines such as PPO, DDPG and SAC can be incorporated in the future work to make the experimental comparison more extensive. Multi-objective evolutionary algorithms like NSGA-II and MOEA-D can also be incorporated in the future work. These algorithms will allow a deeper performance evaluation of the proposed model under equal computational budgets. The simulation setup of the future experiments can also be extended by integrating richer workload models, detailed network and energy assumptions. More diverse edge-fog-cloud topology can also be used and additional statistical validation can be done using multiple independent runs and significance tests. Incorporation of all these enhancements will help in further strengthening the robustness and generalizability of the proposed framework.

Table 4. Ablation Study of System Components

Configuration	Strengths	Limitations	Overall Outcome
FRL-Only	• Learns global placement decisions • Reduces unnecessary offloading	• No local queue optimization • Higher waiting time More • SLA violations	• Moderate performance; constrained by poor task ordering
ABC-Only	• Efficient local reordering • Reduces queue waiting time	• No global context • Creates hotspots • Poor load balancing	• Good local efficiency but unstable global performance
Hybrid FRL + A-ABC (Proposed)	• Best global placement + local optimization • Lowest delay and energy • Balanced load distribution	• Slightly higher scheduling overhead due to dual modules	• Consistently highest performance across all metrics

Table 5. Sensitivity Analysis of Key Hyperparameters

Parameter Group	Hyperparameters Tested	Sensitivity Observation
FRL Module	Learning rate, discount factor, exploration rate	Minor performance variations; stable convergence within broad ranges
A-ABC Module	Colony size, limit value, employed/ onlooker ratio	Larger colonies improve quality but increase overhead; performance remains stable across typical ranges
Federated Learning	Aggregation frequency, participation rate, non-IID severity	Hybrid model remains robust; slight slowdown under extreme non-IID settings but no convergence failures

Author Contributions

Conceptualization, KJ and AA; methodology, KJ; software, KJ; validation, KJ and AA; formal analysis, KJ; investigation, KJ; resources, AA; data curation, KJ; writing—original draft preparation, KJ; writing—review and editing, AA; visualization, KJ; supervision, AA; project administration, KJ and AA. All authors have read and agreed to the published version of the manuscript.

Funding

This research received no external funding.

Conflict of Interest Statement

Author declares no conflict of interest.

Data Availability Statement

No new data were created or analysed in this study.

References

[1] Hassan, N., Gillani, S., Ahmed, E., Yaqoob, I., & Imran, M. (2018). The role of edge computing in internet of things. *IEEE Communications Magazine, 56*(11), 110–115.

[2] Jain, K., Gupta, M., & Abraham, A. (2021). A review on privacy and security assessment of cloud computing. *Journal of Information Assurance and Security, 16*, 161–168.

[3] Chawla, D., Jain, K., Mehra, P. S., Das, A. K., & Bera, B. (2025). Quantum cryptography as a solution for secure wireless sensor networks: Roadmap, challenges and solutions. *Internet of Things, 32*, Article 101610.

[4] Wang, H., et al. (2020). Architectural design alternatives based on cloud/edge/fog computing for connected vehicles. *IEEE Communications Surveys & Tutorials, 22*(4), 2349–2377.

[5] Avasthi, S., Jain, K., Prakash, A., & Sanwal, T. (2024, February). A blockchain architecture for secure decentralized system and computing. In *Proceedings of the 3rd International Conference on Power Electronics and IoT Applications in Renewable Energy and Its Control (PARC)* (pp. 415–420).

[6] Bu, T., et al. (2024). Task scheduling in the internet of things: Challenges, solutions, and future trends. *Cluster Computing, 27*(1), 1017–1046.

[7] Aggarwal, A., Kumar, S., Bhansali, A., Alsekait, D. M., & AbdElminaam, D. S. (2024). A novel optimization approach for energy-efficient multiple workflow scheduling in a cloud environment. *Computer Systems Science & Engineering, 48*(4).

[8] Alsuwat, H., & Alsuwat, E. (2025). Energy-aware and efficient cluster head selection and routing in wireless sensor networks using improved artificial bee colony algorithm. *Peer-to-Peer Networking and Applications, 18*(2), Article 65.

[9] Aggarwal, A., Kumar, S., Bhatt, A., & Shah, M. A. (2022). Solving user priority in cloud computing using enhanced optimization algorithm in workflow scheduling. *Computational Intelligence and Neuroscience, 2022*(1), Article 7855532.

[10] Sagar, A. S., Haider, A., & Kim, H. S. (2025). A hierarchical adaptive federated reinforcement learning for efficient resource allocation and task scheduling in hierarchical IoT network. *Computer Communications, 229*, Article 107969.

[11] Ghanavati, S., Abawajy, J., & Izadi, D. (2020). An energy-aware task scheduling model using ant-mating optimization in fog computing environment. *IEEE Transactions on Services Computing, 15*(4), 2007–2017.

[12] Su, Z., Wang, Y., Luan, T. H., Zhang, N., Li, F., Chen, T., & Cao, H. (2021). Secure and efficient federated learning for smart grid with edge-cloud collaboration. *IEEE Transactions on Industrial Informatics, 18*(2), 1333–1344.

[13] Baghban, H., Rezapour, A., Hsu, C. H., Nuannimnoi, S., & Huang, C. Y. (2022). Edge-AI: IoT request service provisioning in federated edge computing using actor-critic reinforcement learning. *IEEE Transactions on Engineering Management, 71*, 12519–12528.

[14] Soula, M., Karanika, A., Kolomvatsos, K., Anagnostopoulos, C., & Stamoulis, G. (2022). Intelligent task allocation at the edge based on machine learning and bio-inspired algorithms. *Evolving Systems, 13*(2), 221–242.

[15] Ramezani Shahidani, F., Ghasemi, A., Toroghi Haghighat, A., & Keshavarzi, A. (2023). Task scheduling in edge-fog-cloud architecture: A multi-objective load balancing approach using reinforcement learning algorithm. *Computing, 105*(6), 1337–1359.

[16] Kar, B., Yahya, W., Lin, Y. D., & Ali, A. (2023). Offloading using traditional optimization and machine learning in federated cloud-edge-fog systems: A survey. *IEEE Communications Surveys & Tutorials, 25*(2), 1199–1226.

[17] Kim, D. Y., Lee, D. E., Kim, J. W., & Lee, H. S. (2023). Collaborative policy learning for dynamic scheduling tasks in cloud-edge-terminal IoT networks using federated reinforcement learning. *IEEE Internet of Things Journal, 11*(6), 10133–10149.

[18] Wu, H., Gu, A., & Liang, Y. (2024). Federated reinforcement learning-empowered task offloading for large models in vehicular edge computing. *IEEE Transactions on Vehicular Technology.*

[19] Shen, W., Lin, W., Wu, W., Wu, H., & Li, K. (2025). Reinforcement learning-based task scheduling for heterogeneous computing in end-edge-cloud environments. *Cluster Computing, 28*(3), Article 179.

[20] Shidik, G. F., et al. (2025). Novel unsupervised cluster reinforcement Q-learning in minimizing energy consumption of federated edge cloud. *IEEE Access.*

[21] Lilhore, U. K., Simaiya, S., Sharma, Y. K., Rai, A. K., Padmaja, S. M., Nabilal, K. V., ... & Alsufyani, H. (2025). Cloud-edge hybrid deep learning framework for scalable IoT resource optimization. *Journal of Cloud Computing, 14*(1), 5.

[22] Del-Pozo-Puñal, E., García-Carballeira, F., & Camarmas-Alonso, D. (2023). A scalable simulator for cloud, fog and edge computing platforms with mobility support. *Future Generation Computer Systems, 144*, 117–130.

[23] Jain, K., Agarwal, A., Agrawal, S., & Aggarwal, A. (2025). Digital twins in modern healthcare: A comprehensive review of architectures, applications, and challenges. *Wiley Interdisciplinary Reviews: Computational Statistics, 17*(3), Article e70041.

[24] Shinde, A., Iyer, S., Jain, K., & Sagi, S. (2022). Smart City and Village: Future Trends. In *Artificial Intelligence for Smart Cities and Smart Villages: Advanced Technologies, Development, and Challenges* (pp. 29-43). Bentham Science Publishers.

[25] Chawla, D., Jain, K., Mehra, P. S., Das, A. K., & Bera, B. (2025). Quantum cryptography as a solution for secure Wireless Sensor Networks: Roadmap, challenges and solutions. *Internet of Things*, *32*, 101610.