scientific reports

OPEN

NF-MORL: a neuro-fuzzy multi-objective reinforcement learning framework for task scheduling in fog computing environments

Xiaomo Yu[1,3,5], Ling Tang[2]✉, Jie Mi[3], Long Long[4], Xiao Qin[4], Xiuming Li[7] & Qinglian Mo[6]

The proliferation of IoT devices has exerted significant demand on computing systems to process data rapidly, efficiently, and in proximity to its source. Conventional cloud-based methods frequently fail because of elevated latency and centralized constraints. Fog computing has emerged as a viable option by decentralizing computation to the edge; yet, successfully scheduling work in these dynamic and heterogeneous contexts continues to pose a significant difficulty. This research presents A Neuro-Fuzzy Multi-Objective Reinforcement Learning (NF-MORL), an innovative framework that integrates neuro-fuzzy systems with multi-objective reinforcement learning to tackle task scheduling in fog networks. The concept is straightforward yet impactful: a Takagi–Sugeno fuzzy layer addresses uncertainty and offers interpretable priorities, while a multi-objective actor–critic agent acquires the capacity to reconcile conflicting objectives makespan, energy consumption, cost, and reliability through practical experience. We assessed NF-MORL using empirical data from Google Cluster and EdgeBench. The findings were promising: relative to cutting-edge techniques, our methodology decreased makespan by up to 35%, enhanced energy efficiency by about 30%, reduced operational expenses by up to 40%, and augmented fault tolerance by as much as 37%. These enhancements persisted across various workload sizes, demonstrating that NF-MORL can effectively adjust to fluctuating situations. Our research indicates that integrating human-like reasoning through fuzzy logic with autonomous learning via reinforcement learning can yield more effective and resilient schedulers for actual fog deployments.

Keywords Fog computing, Task scheduling, Neuro-fuzzy systems, Multi-objective reinforcement learning, Energy efficiency, Fault tolerance

The proliferation of billions of IoT devices has resulted in data generation surpassing the capacity of conventional cloud systems, particularly for applications requiring immediate responses [1,2,3], such as autonomous vehicles and remote healthcare [4,5]. Fog computing was developed to address this issue by relocating computation nearer to the devices [6]; however, it has its own challenges: nodes possess varying capabilities, workloads exhibit significant fluctuations, and failures are prevalent [7,8,9].

We promptly recognized that current scheduling algorithms be they heuristic [10], single-objective reinforcement learning [11,12], or some hybrid methodologies encounter difficulties under these circumstances. Most reinforcement learning-based methods [13], for instance, presume pristine and comprehensive state information, which is never the reality in actual fog networks [14]. Conversely, fuzzy logic adeptly manages ambiguity but does not possess the capacity for enhancement over time [15,16].

This finding prompted us to inquire: what if we could amalgamate the advantages of both? Thus, NF-MORL was established a system wherein a neuro-fuzzy module delivers interpretable, real-time priority amidst uncertainty, and a multi-objective reinforcement learning agent perpetually enhances both the policy and the fuzzy rules

[1]Guangxi Colleges and Universities Key laboratory of Intelligent Logistics Technology, Nanning Normal University, Nanning 530001, Guangxi, China. [2]College of The Arts, Guangxi Minzu University, Nanning 530001, Guangxi, China. [3]School of Logistics Management and Engineering, Nanning Normal University, Nanning 530001, Guangxi, China. [4]School of Artificial Intelligence, Nanning Normal University, Nanning 530001, Guangxi, China. [5]Guangxi Academy of Artificial Intelligence, Nanning 530001, Guangxi, China. [6]Nanning Yunou Logistics Co., Ltd., Nanning 530001, Guangxi, China. [7]Guangxi Research Institute of Mechanical Industry Co., Ltd,, Nanning 530001, Guangxi, China. ✉email: tangling0312@163.com

based on empirical performance feedback [17,18]. The proliferation of billions of IoT devices has resulted in data generation surpassing the capacity of conventional cloud systems, particularly for applications requiring immediate responses, such as autonomous vehicles and remote healthcare. Fog computing was developed to address this issue by relocating computation nearer to the devices; however, it has its own challenges: nodes possess varying capabilities, workloads exhibit significant fluctuations, and failures are prevalent.

We promptly recognized that current scheduling algorithms be they heuristic, single-objective reinforcement learning, or some hybrid methodologies encounter difficulties under these circumstances. Most reinforcement learning-based methods, for instance, presume pristine and comprehensive state information, which is never the reality in actual fog networks. Conversely, fuzzy logic adeptly manages ambiguity but does not possess the capacity for enhancement over time.

This finding prompted us to inquire: what if we could amalgamate the advantages of both? Thus, NF-MORL was established a system wherein a neuro-fuzzy module delivers interpretable, real-time priority amidst uncertainty, and a multi-objective reinforcement learning agent perpetually enhances both the policy and the fuzzy rules based on empirical performance feedback.

In contrast to numerous current methods that optimize only one or two objectives, NF-MORL concurrently addresses four essential goals: minimizing completion time (makespan), reducing energy consumption, decreasing costs, and enhancing reliability. Initial trials demonstrated that this collaborative optimization was essential for attaining balanced and practical performance.

The primary contributions of this study are:

- A novel hybrid framework NF-MORL that seamlessly combines an adaptive Takagi–Sugeno neuro-fuzzy system with multi-objective actor–critic reinforcement learning for fog task scheduling.
- A bidirectional learning mechanism: the reinforcement learning agent enhances the scheduling strategy, while performance feedback perpetually adjusts the fuzzy membership functions and rule outcomes capabilities that static fuzzy systems lack.
- A pragmatic three-tier architecture (edge–fog–cloud) that facilitates distributed, low-latency decision-making and scalable training.
- Comprehensive assessment of actual Google Cluster and EdgeBench traces, demonstrating consistent and substantial enhancements above contemporary DRL and hybrid benchmarks across all four objectives.
- This amalgamation of interpretability, adaptability, and multi-objective cognizance renders NF-MORL exceptionally appropriate for practical fog implementations.

This study aims to develop a scheduling system capable of managing the complexities of real fog environments, including uncertain inputs, conflicting objectives, and dynamic conditions. We developed NF-MORL, a framework that integrates neuro-fuzzy reasoning with multi-objective reinforcement learning, which surpasses existing methodologies in performance and provides a level of transparency absent in conventional deep reinforcement learning techniques.

The experimental findings are self-evident: Reductions of up to 35% in makespan, 30% in energy consumption, 40% in costs, and a 37% enhancement in fault tolerance are significant; they denote substantial advancements for real systems. Anticipating future developments, we identify several promising avenues: implementing NF-MORL on actual hardware, augmenting it to accommodate security constraints, and investigating lifelong learning to ensure continuous improvement post-deployment.

We anticipate that this work will inspire additional researchers to investigate hybrid intelligence methodologies integrating the commonsense reasoning in which humans excel with the scalability offered by machines as a means to achieve genuinely autonomous and reliable edge systems.

2. Related Works

Task scheduling in fog computing has been extensively explored to improve performance, scalability, and energy efficiency in distributed Internet of Things environments. Early studies emphasized heuristic and static optimization strategies, yet they struggled to handle the heterogeneity and stochastic nature of fog infrastructures. Recent research trends have shifted toward machine learning and reinforcement learning approaches to enable dynamic and data-driven decision-making. The following review examines key contributions in task scheduling, energy management, and hybrid intelligent optimization within fog computing ecosystems. In[1], an extensive analysis of scheduling strategies in fog computing is conducted, categorizing them into heuristic, meta-heuristic, and learning-based methodologies. Shortcomings of traditional methods are highlighted, particularly their inability to adapt to real-time workload variations and unpredictable network conditions. The review indicates that hybrid models integrating reinforcement learning and fuzzy logic remain underexplored. In[2], a deep reinforcement learning scheduling strategy utilizing a proximal policy optimization (PPO) agent is developed to reduce system load and reaction time in fog-edge-cloud infrastructures through dynamic learning.

A distributed deep reinforcement learning system is introduced in[3] that concurrently improves energy consumption and reliability for job scheduling in fog computing. Despite realizing considerable energy savings, it employs single-objective scaling and does not include interpretable uncertainty modeling using neuro-fuzzy systems, leading to suboptimal trade-offs among time, cost, and fault tolerance when compared to the Pareto-efficient NF-MORL technique.

A evolutionary approach including selective repair is presented for scheduling IoT operations with time limitations in fog-cloud situations in[4]. While successful for static processes, its meta-heuristic characteristics restrict online adaptation to dynamic workload fluctuations and do not use reinforcement learning or fuzzy reasoning for managing uncertainty. Researchers in[5] have examined live migration algorithms for associated virtual machines in cloud data centers to enhance resource utilization and fault tolerance. This study is confined to centralized cloud infrastructures and does not include distributed fog layer scheduling or real-time multi-objective optimization. The research in[6] introduced a dynamic network performance provisioning technique to facilitate "network in a box" for industrial applications.

RT-SEAT, a real-time hybrid scheduler designed to concurrently reduce energy consumption and peak temperature on heterogeneous multi-core systems, was created in[7]. Although it demonstrates superior thermal-aware performance, it fails to account for multi-objective trade-offs such as cost and fault tolerance, and it does not include learning-based adaptation. A fault-tolerant real-time scheduler for heterogeneous multiprocessor systems using redundancy and migration approaches is introduced in[8]. Notwithstanding robust reliability assurances, this technique remains static, lacking online learning and neuro-fuzzy uncertainty management capabilities.

A separate research[9] presented a secure multi-reference attribute-based access control mechanism for fog-enabled e-health systems, termed SMAC, which only focuses on data privacy and access security, excluding job scheduling, resource allocation, and performance optimization. A hybrid knowledge- and data-driven methodology was developed for mass flow scheduling in hybrid workshops with dynamic order input[10]. This technique, although successful in industrial settings, is unsuitable for fog computing environments and lacks components of distributed reinforcement learning and fuzzy inference.

A dynamic prescriptive performance controller using event-driven reinforcement learning was presented for nonlinear systems with input delays in[11]. This work emphasizes continuous-time control inside the RL basis, excluding discrete-time job scheduling, fuzzy logic, and multi-objective Pareto optimization.

In[12], a self-stimulated reinforcement learning system using particle swarm optimization was devised for fault-tolerant optimum control in zero-sum games with saturated inputs. It excels in competitive environments but is inapplicable to fog task scheduling and lacks neuro-fuzzy interpretation and multi-objective management capabilities. A distributed adaptive sliding mode formation controller with specified time restrictions for heterogeneous nonlinear multi-agent systems is suggested in[13]. This study focuses on physical formation tracking, rather than on computational job scheduling and resource management in dispersed and heterogeneous fog networks.

In[14], researchers introduced a multi-objective scheduling and offloading framework inside Fog-Cloud settings, where system uncertainties are represented using fuzzy logic. The objective is to concurrently minimize workflow execution duration and energy use, with findings indicating that a fuzzy approach achieves a superior equilibrium between efficiency and processing expenses compared to deterministic alternatives. This study emphasizes the optimization of trade-offs and the modeling of uncertainty.

Another research[15] examined work management in fog computing via the lens of federated reinforcement learning. This concept creates a common policy for distributed decision-making without requiring raw data interchange between nodes, hence improving data security and system scalability. This research's primary novelty is in the integration of Federated Learning with Reinforcement Learning to optimize job management and mitigate processing congestion inside the fog network.

[16] also examines offloading techniques based on Reinforcement Learning and Deep Learning, and assesses several automated decision-making algorithms for task location identification. This study demonstrates that using deep networks to learn offloading strategies yields superior performance compared to traditional approaches under dynamic settings. This study primarily focuses on deep learning and enhancing the adaptive decision rate.

Table 1 presents a systematic comparison of the examined methodologies, emphasizing their technical frameworks, fundamental techniques, operational contexts, targeted aims, and principal constraints. This overview delineates a distinct evolution from heuristic and static approaches to intelligent, learning-based solutions, while highlighting enduring deficiencies in multi-objective optimization, uncertainty management, and adaptive rule learning that are essential for next-generation fog scheduling systems. The table functions as a succinct reference for comprehending the progression and existing issues in the domain.

System model and problem formulation

The suggested NF-MORL system model features a three-tier hierarchical architecture, consisting of edge, fog, and cloud layers, aimed at facilitating resilient, multi-objective, and adaptive task scheduling in heterogeneous IoT contexts. The comprehensive architecture of the fog computing environment utilized in this study is depicted in Fig. 1, illustrating the interaction among IoT devices, fog nodes, and the cloud layer. This paradigm utilizes the advantages of each layer to deliver low-latency, energy-efficient, and context-aware services, while guaranteeing fault tolerance and scalability.

The proposed NF-MORL framework models a heterogeneous fog environment as a set of physical machines (PMs) that host various virtual machines (VMs). Each task T_k is defined by a vector of computational parameters, comprising CPU usage (CPU_k), memory requirement (MEM_k), bandwidth demand (BW_k), and task length (TL_k).

A hierarchical fuzzy topological framework for high-dimensional regression shows how structured fuzzy inference may improve scalability and uncertainty modeling in large-scale decision-making systems[17]. The introduction of a dynamic event-driven adaptive fuzzy sliding-state controller for unknown nonlinear systems shows how hierarchical fuzzy mechanisms may stabilize complicated settings with minimum processing[18].

Table 2 summarizes the primary parameters, mathematical symbols, and variables utilized in the NF-MORL framework. The NF-MORL framework utilizes a Takagi–Sugeno neuro-fuzzy inference system to dynamically assess task significance based on CPU use, memory consumption, bandwidth availability, and job duration. The relevant fuzzy rule basis for task prioritizing is presented in Table 3. The system's workload and overall processing capabilities are quantitatively represented by Eq. (1) through 4[3]:

Ref	Paradigm	Model	Environment	Main objectives optimized	Main limitations/research gap
1	Survey	Heuristic, Meta-heuristic, Learning-based	Fog	Performance, scalability	Hybrid RL–Fuzzy approaches still underexplored
2	DRL scheduling	PPO-based RL agent	Fog–edge–cloud	Load reduction, latency minimization	No uncertainty modeling, single-objective orientation
3	Distributed DRL	Energy-aware DRL scheduling	Fog	Energy efficiency, reliability	No interpretable fuzzy modeling; lacks multi-objective trade-offs
4	Evolutionary scheduling	Meta-heuristic with selective repair	Fog–cloud	Time-constrained IoT workflow execution	Offline optimization; weak adaptability to dynamic workload
5	Resource migration	VM live-migration optimization	Cloud DC	Utilization, reliability	Not fog-oriented; lacks distributed scheduling
6	Dynamic network allocation	Performance provisioning framework	Industrial fog	Network stability, on-demand allocation	No task scheduling or multi-objective optimization
7	Hybrid real-time scheduling	RT-SEAT thermal-aware scheduler	Heterogeneous multi-core	Energy + temperature reduction	No cost/fault-tolerance objective; no learning adaptation
8	Fault-tolerant real-time scheduler	Redundancy + task migration	Heterogeneous multiprocessor	Reliability, deadline guarantee	Static scheme; lacks online learning & fuzzy uncertainty handling
9	Secure access management	SMAC ABE mechanism	Fog-enabled e-health	Privacy + secure access	Does not address scheduling or optimization
10	Hybrid knowledge–data driven	Mass-flow scheduling	Industrial workshop	Dynamic order scheduling	Not fog computing; no RL/FL distributed optimization
11	RL control framework	Event-driven RL for delay systems	Continuous nonlinear systems	Real-time adaptive control	Not for fog scheduling; lacks discrete Pareto optimization
12	Competitive RL optimization	PSO-boosted RL control	Zero-sum game systems	Fault-tolerant optimization	Inapplicable to fog task offloading; missing fuzzy + multi-objective logic
13	Distributed multi-agent control	Adaptive sliding mode controller	Heterogeneous MAS	Convergence under time constraints	No scheduling or fog workload context
14	Multi-objective Fuzzy Scheduling	Fuzzy-based offloading for workflow	Fog–cloud	Execution time + energy minimization	Deterministic baseline comparison; lacks RL adaptability
15	Federated reinforcement learning	FL + RL for distributed task allocation	Fog	Scalability, privacy, congestion reduction	Multi-objective extensions + fuzzy uncertainty handling not addressed
16	RL + deep learning offloading	DRL decision-making for task placement	Fog	Offloading accuracy, adaptive decision-making	RL & DL only; lacks multi-modal fuzzy uncertainty reasoning

Table 1. Summary of task scheduling and optimization approaches in fog computing.

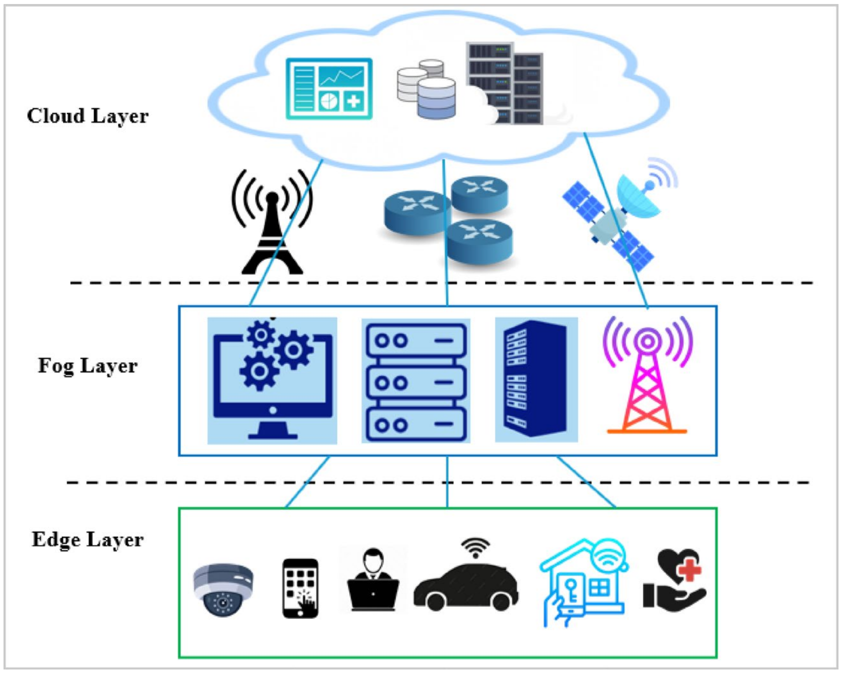


Fig. 1. Architecture of fog computing system.

Notation	Description
(s_t)	State at time *t*, representing environmental conditions, fog node status, and multi-objective context parameters.
(a_t)	Action at time *t*, representing the task assignment decision among fog nodes.
(R_t)	Reward at time *t*, integrating weighted objectives of makespan, energy efficiency, cost, and fault tolerance.
(γ)	Discount factor adjusting the relative importance of future aggregated multi-objective rewards.
(L_{actor})	Actor loss computed by multi-objective policy gradient methods to optimize actions across Pareto fronts.
(L_{critic})	Critic loss used to estimate and stabilize multi-objective value functions during training.
(T)	Task length representing the computational complexity of each job.
(P)	Task priority determined dynamically through neuro-fuzzy inference.
(CPU_i)	Current CPU utilization of fog node *i*.
(Mem_i)	Available memory of fog node *i*.
(BW_i)	Bandwidth capacity at fog node *i*.
(E_i)	Current energy level at fog node *i*.
(S)	CPU processing speed at each fog node.
(C_f)	Cost factor representing computation and transmission costs per task.
(F_f)	Fault factor representing node reliability and failure probability.
(L_t)	Task length in seconds, indicating computational workload.
(D_t)	Task deadline, defining the maximum acceptable completion time.
(R_q)	Resource quality factor combining CPU, memory, and bandwidth performance indices.
(W_k)	Weight coefficient assigned to objective *k* in the multi-objective reward function.
(μ_j, σ_j)	Mean and variance of fuzzy membership functions for linguistic variable *j*.
($f_j(x)$)	Fuzzy membership function used in the neuro-fuzzy inference layer.
(θ_π)	Parameters of the multi-objective policy network.
(θ_v)	Parameters of the multi-objective value network.
(θ_{nf})	Parameters of the neuro-fuzzy inference subsystem.
($h_\pi(s_t, a_t)$)	Action–value function of the multi-objective actor network under neuro-fuzzy control.

Table 2. Notations used in NF-MORL model.

CPU usage	Memory load	Bandwidth	Task length	Output: priority
Low	Low	High	Short	Very high
Medium	Medium	Medium	Medium	Medium
High	Low	Medium	Long	Low
High	High	High	Short	Medium
Low	High	Medium	Short	High
Medium	Low	High	Medium	High
Low	Medium	Low	Long	Low
Medium	High	Low	Short	Medium
High	Medium	High	Medium	Medium
Low	High	High	Long	Medium

Table 3. Fuzzy rules of Takagi–Sugeno neuro-fuzzy inference system for task prioritization.

$$wl_{VM.v} = \sum_{k \in A(v)} wl_{k.v} \tag{1}$$

In Eq. 1, the instantaneous workload of virtual machine v, represented as $wl_{VM.v}$ is determined as the cumulative burden of all tasks $k \in A(v)$ currently allocated to v, where $A(v)$ is the collection of tasks assigned to virtual machine v within fog node M.

$$wl_{PM.p} = \sum_{v \in V(p)} wl_{VM.v} \tag{2}$$

Equation (2) calculates the total workload exerted on physical fog node p, represented as $wl_{PM.p}$, by aggregating the workloads of all virtual machines $v \in V(p)$ residing on p, with $V(p)$ being the collection of virtual machines hosted on physical machine p.

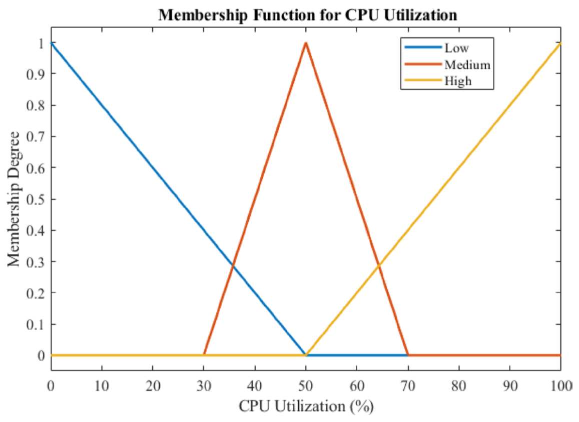


Fig. 2. CPU utilization membership function.

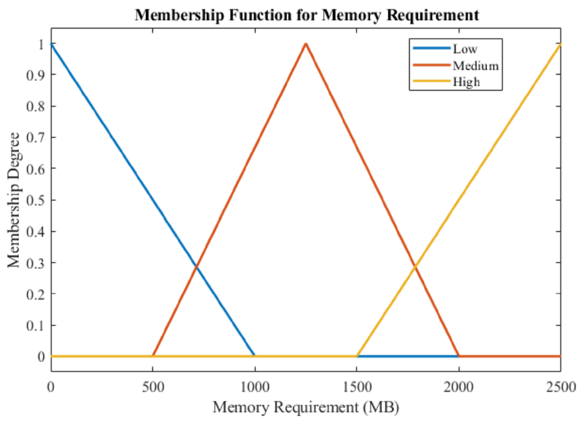


Fig. 3. Memory membership function.

$$prc_{VM.v} = n_{core}^{(v)} \times MIPS_v \tag{3}$$

Equation (3) delineates the effective processing capacity of virtual machine v in million instructions per second (MIPS), calculated as the product of the allotted CPU cores $n_{core}^{(v)}$ and the per-core processing speed $MIPS_v$ of the corresponding physical node.

$$TotPrc = \sum {}_v prc_{VM.v} \tag{4}$$

where delineates the cumulative processing capacity of the whole fog layer, $TotPrc$, as the aggregate of the individual processing capacities $prc_{VM.v}$ across all virtual machines v within the system.

All input attributes are standardized to the range [0,1] to ensure uniformity across distributed fog layers and to enhance adaptive learning during the fuzzy inference phase.

Fuzzification of task attributes and adaptive rule base

The NF-MORL design comprises three hierarchical layers: Edge, Fog, and Cloud, each fulfilling a distinct function in the scheduling process. Continuous input features are converted into linguistic variables by the use of fuzzy membership functions. This study use triangle membership functions for all language terms. Their piecewise-linear configuration facilitates rapid assessment with minimal computational burden in real-time fog scenarios, while still offering sufficient adaptability to represent nonlinear relationships among CPU, memory, bandwidth, and task duration. In comparison to alternatives like trapezoidal or sigmoid functions, the triangular shape provides an effective equilibrium among interpretability, ease of implementation, and numerical stability in the NF-MORL training process.

Each characteristic of CPU, memory, bandwidth, and job duration is associated with fuzzy sets designated as Low, Medium, and High, facilitating seamless and adaptable transitions between states. In the Edge layer, the adaptive fuzzification and inference modules determine task priorities according to the membership functions illustrated in Figs. 2, 3, 4 and 5. The deterministic priorities derived from the fuzzification method depicted in Fig. 6 are employed to execute dynamic scheduling on heterogeneous fog nodes within the Fog layer.

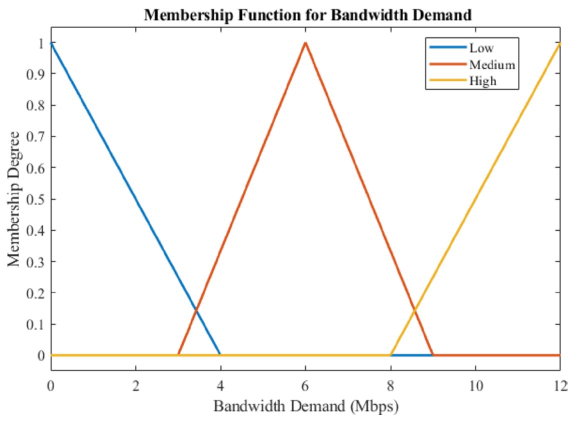


Fig. 4. Bandwidth membership function

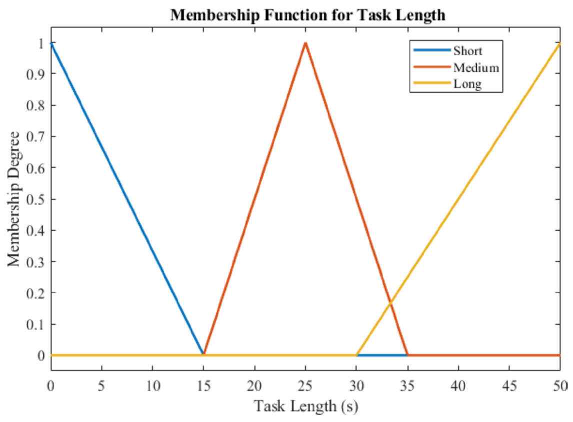


Fig. 5. Task length membership function.

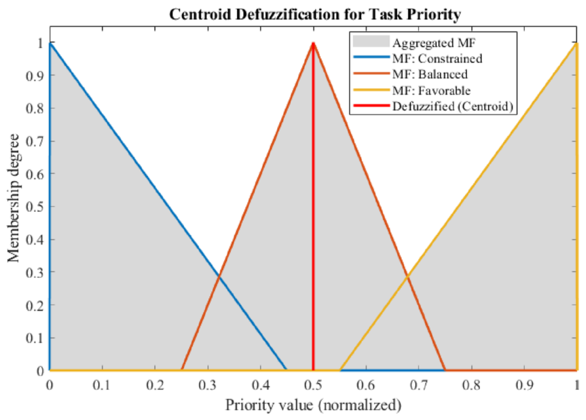


Fig. 6. Centroid defuzzification.

The Cloud layer functions as a supervisory entity that ensures global synchronization, facilitates policy sharing, and manages offloading for tasks that are either delay-tolerant or computationally heavy[15]. Algorithm 1 summarizes the process of converting input parameters into adaptive task priorities via the neuro-fuzzy inference model. This algorithm delineates the sequential computation of task priorities via fuzzification, rule assessment, and the modification of adaptive parameters via reinforcement learning input.

For each fuzzy rule i, if $\mu_{ij}(x_j)$ denotes the membership degree of the input x_j in its corresponding fuzzy set, the firing strength of the rule is computed as:

$$\lambda_i = \prod_{j=1}^{m} \mu_{ij}(x_j) \quad (5)$$

Every neuro-fuzzy rule is expressed in Takagi–Sugeno format, whereby the resultant function is linear about the input variables, and its parameters are adaptively adjusted throughout the learning process.

This enables the inference mechanism to adapt according to environmental feedback from the reinforcement learning agent. The adaptive linear consequence of rule III is specified as:

$$\varphi_i(x) = a_i \times MEM + b_i \times CPU + c_i \times BW + d_i \times TL + e_i \quad (6)$$

Equation (6) delineates the adaptive linear consequence of the $i-th$ Takagi–Sugeno rule, whereby $\phi_{i(x)}$ represents the rule output, x = [MEM, CPU, BW, TL] denotes the input vector, and the coefficients $a_i,\ b_i,\ c_i,\ d_i,\ e_i$ are acquired online through back-propagated gradients from the reinforcement learning agent.

The aggregated task priority is computed by combining all activated rules using a weighted average method:

$$P(T_k) = \frac{\sum_{i=1}^{R} \lambda_i \Phi_i(T_k)}{\sum_{i=1}^{R} \lambda_i} \quad (7)$$

The final crisp task priority $P(T_k)$ is computed as the weighted average of all activated rule outputs $\Phi_i(T_k)$, with weights λ_i denoting the normalized firing strengths of each rule $\left(\sum \lambda_i = 1\right)$.

The ambiguous output surface delineates three specific operational states: Constrained, Balanced, and Favorable, which correlate to the overall resource status of the fog environment. Figure 11 depicts the centroid-based defuzzification method that transforms the aggregated fuzzy output into a precise numerical task priority, expressed as[19]:

$$x_c = \frac{\int x \mu_{agg}(x)\,dx}{\int \mu_{agg}(x)\,dx} \quad (8)$$

Equation (8) executes centroid defuzzification to transform the aggregated fuzzy output $\mu_{agg}(x)$ into a definitive numerical priority x_c, which is subsequently utilized by the scheduling policy in the following actor–critic phase.

This clear priority value establishes the quantitative foundation for scheduling and optimization in the latter phases of the NF-MORL system.

In Algorithm 1 The task length TL_j is obtained by dividing the computational workload L_j by the CPU processing speed C_{sp}, following the standard execution-time formulation. The memory requirement MEM_j is directly taken from the task metadata provided in the dataset. The bandwidth demand BW_j is computed based on the input size S_j and the network transfer time per MB T_{net}, in accordance with common data transmission models in fog and edge environments.

Quantitative performance formulations

The defuzzified priorities are integrated into four cumulative performance indicators that direct the multi-objective reinforcement learning process: makespan, energy consumption, execution cost, and fault probability. These formulations function as reward components rather than comprehensive physical system models, although they nonetheless align with commonly accepted concepts in fog and cloud scheduling. The makespan denotes the maximum completion time across all tasks, dictated by their initiation times and execution durations[3,4]:

$$Makespan = \max_{j \in D} (Start_j + ExecTime_j) \quad (9)$$

where D is the task set, $Start_j$ denotes the start time of task j, and $ExecTime_j$ is its execution time based on the workload and processing speed of the assigned fog node. Energy consumption is represented as the total power utilized by all fog nodes throughout task execution[20]:

$$E = \sum_{j \in D} \left(P_{active(n_j)} \times \left(\frac{L_j}{Csp(n_j)} \right) \right) \quad (10)$$

where $P_{active}(n_j)$ is the active power of node n_j, L_j is the computational load, and $Csp(n_j)$ is the node's processing speed. The execution cost represents the cumulative processing duration adjusted by the cost rate of each fog node.

$$Cost = \sum_{j \in D} \left(C_{unit}(n_j) \times \left(\frac{L_j}{C_{sp}(n_j)} \right) \right) \quad (11)$$

Where $C_{unit}(n_j)$ represents the per-unit processing cost of node n_j. Fault probability measures the aggregate likelihood of failure across all nodes participating in task execution[19]:

```
Input:
    Task set $D$
    CPU processing speed of fog node $Csp$
    Network transfer time per MB $Tnet$
    Initial fuzzy parameters $\theta fuzzy$
Output:
    Task priority list $PList$

1.  Load task dataset $D$
2.  Initialize system parameters $(Csp, Tnet)$
3.  Define task attribute vector A = {TL, MEM, CPU, BW}
4.  Initialize empty structure JobSet
5.  For each task j in $D$ do
6.      Compute task length: $TL_j = \frac{L_j}{Csp}$
7.      Estimate memory usage:
          $MEM_j = M_j$
8.      Compute required bandwidth: $BW_j = S_j \times Tnet$
9.      Estimate CPU utilization:
          $CPU_j = \left(\frac{L_j}{Csp}\right) \times 100$
10.     Append $\{TL_j, MEM_j, CPU_j, BW_j\}$ to JobSet
11. End for
12. Define fuzzy input variables (TL, MEM, CPU, BW)
13. Define fuzzy output variable Priority
14. Initialize membership functions using $\theta fuzzy$
15. Construct the Takagi–Sugeno fuzzy rule base
16. For each fuzzy rule r in R do
17.     Compute firing strength:
          $\lambda_r = \prod \mu_r, i(x_i)$   // using Eq. (5)
18.     Compute rule consequent:
          $\varphi_i = a_i \cdot MEM + b_i \cdot CPU + c_i \cdot BW + d_i \cdot TL + e_i$   //using Eq. (6)
19. End for
20. Aggregate rule outputs to compute priority:
      $P(T_k) = \frac{(\Sigma \lambda_r \cdot \varphi_r)}{\Sigma \lambda_r}$   // using Eq. (7)
21. Apply centroid defuzzification to obtain crisp priority $P(T_k)$
22. Initialize empty list $PList$
23. For each task $j$ in JobSet do
24.     Compute final priority $P_j$ using updated fuzzy parameters
25.     Append $P_j$ to $PList$
26. End for
27. Update fuzzy parameters $\theta fuzzy$ using RL feedback
28. Return $PList$
```

Algorithm 1. Neuro-fuzzy adaptive task priority evaluation.

$$FaultProb = 1 - \prod_{j \in D} (1 - f(n_j)) \tag{12}$$

where $f(n_j)$denotes the failure probability of node n_j.

The updated formulations offer a succinct and accurate assessment of system behavior, allowing the RL agent to understand the trade-offs between latency, energy consumption, cost, and reliability.

Proposed methodology

This section presents the NF-MORL framework shown in Fig. 7 for flexible and adaptive task scheduling in diverse fog environments. In contrast to conventional or Round Robin schedulers, which function in a rigid and non-adaptive fashion, NF-MORL integrates real-time feedback via a neuro-fuzzy inference layer and a multi-objective reinforcement learning module. The time-varying DRL method with efficient backtracking accelerates policy convergence, proving reinforcement learning can perform well with little training data[21].

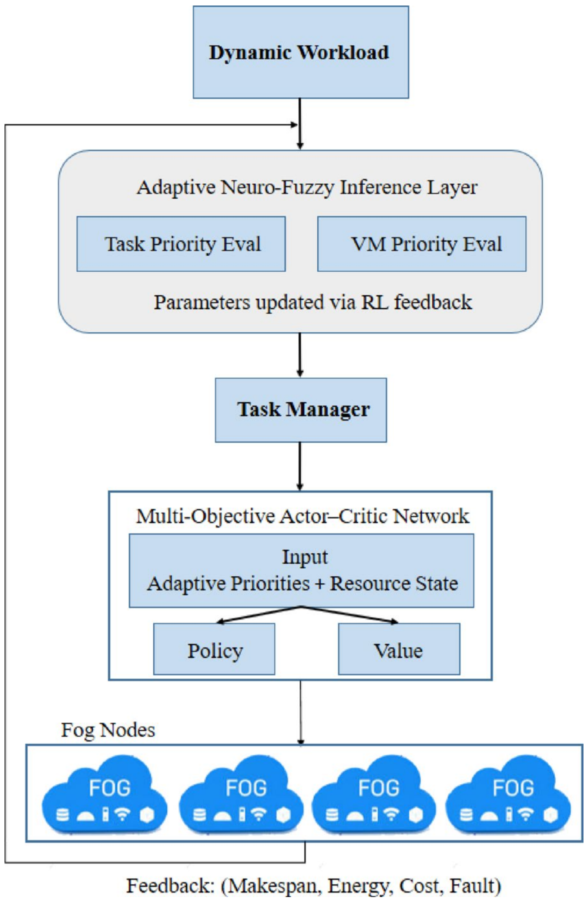

Dynamic Workload
Adaptive Neuro-Fuzzy Inference Layer
Task Priority Eval
VM Priority Eval
Parameters updated via RL feedback
Task Manager
Multi-Objective Actor–Critic Network
Input
Adaptive Priorities + Resource State
Policy
Value
Fog Nodes
FOG
FOG
FOG
FOG
Feedback: (Makespan, Energy, Cost, Fault)

Fig. 7. System model architecture.

Additionally, neighborhood-aware multi-agent reinforcement learning for traffic signal coordination indicates that collaborative learning enhances global control only when agent interactions include explicit backtracking[22].

NF-MORL integrates the interpretability of fuzzy logic with the autonomous optimization features of actor–critic learning. The neuro-fuzzy layer produces adaptive task and virtual machine priorities, whilst the actor-critic network enhances scheduling judgments based on multi-objective incentives. Agents consistently monitor CPU, memory, bandwidth, and task characteristics, refining both fuzzy parameters and neural rules based on environmental feedback.

The approach utilizes deep actor–critic networks with concurrent agents to expedite learning and improve generalization among fog nodes. NF-MORL achieves scalable, resilient, and completely autonomous scheduling for dynamic fog computing environments by concurrently maximizing many performance targets and responding in real-time to evolving conditions.

State space (S_t)

The state space represents the dynamic environment of the fog system at time step *t*.

Each state vector S_t encodes the operational conditions of fog nodes and the characteristics of incoming tasks.

In the NF-MORL model, the state vector is defined as[23]:

$$S_t = \{CPU_{util}.\ MEM_{usage}.\ BW_{avail}.\ TL_{task}.\ P_{task}.\ P_{VM}\} \tag{13}$$

where $u_i$denotes the current CPU utilization of each node, $m_i$represents available memory resources, b_i indicates network bandwidth availability, $l_i$is estimated task length, and $p_i$and $q_i$are adaptive priorities produced by the Neuro-Fuzzy Inference Layer. This multi-dimensional state captures both system resources and workload characteristics, enabling the agent to make context-aware scheduling decisions.

Action (A_t)

The action defines the scheduling decision made by the RL agent at each time step.

In NF-MORL, an action A_t corresponds to the mapping of a task to a specific VM or fog node, based on current state and learned policy.

$$A_t = \{a_1.\ a_2.\ \ldots\ .\ a_n\}.\ a_i \in \{assignT_i \to VM_j\} \tag{14}$$

The actor network outputs a probability distribution over available nodes, representing the likelihood of selecting each node for a given task. Actions are selected stochastically during training (exploration) and deterministically during evaluation.

Reward (R_t)

The reward function measures how well an action performs with respect to multiple objectives.

NF-MORL employs a multi-objective reward model that balances system efficiency, energy cost, and reliability:

$$R_t = \alpha_1 R_{makespan} + \alpha_2 R_{energy} + \alpha_3 R_{cost} + \alpha_4 R_{fault} \tag{15}$$

where the coefficients α_i represent the importance weights for each objective.

Each reward component is normalized in [0,1] and defined as:

- $R_{makespan} = 1 - \frac{Texec}{Tmax}$
- $R_{energy} = 1 - \frac{Enode}{Emax}$.
- $R_{cost} = 1 - \frac{Cnode}{Cmax}$
- $R_{fault} = 1 - F_{prob}$

This formulation encourages actions that minimize makespan, energy consumption, and execution cost, while improving fault tolerance.

Policy(π)

The policy $\pi(as;\theta_{actor})$ defines the decision-making behavior of the actor network.

It maps the current state S_t to an action probability distribution over possible scheduling decision.

In the NF-MORL framework, the policy network receives both raw system metrics and adaptive fuzzy priorities as input features. In policy development, W_s and W_p represent the trainable weight vectors associated with the raw system metrics and adaptive fuzzy priorities, respectively[24]. These weights adjust the proportional influence of each feature group before processing by the policy network:

$$\pi(A_t S_t\ ;\theta_{actor}) = f_{Soft\max}(W_s \cdot S_t + W_p \cdot P_{fuzzy} + \ b) \tag{16}$$

where $f_{Softmax}$ ensures that output probabilities lie within [0,1]. The actor's objective is to maximize the expected cumulative reward[1]:

$$J(\theta_{\ actor}) = \ \mathbb{E}_\pi\ [R_t] \tag{17}$$

During learning, exploration is encouraged via entropy regularization to avoid early convergence to suboptimal policies.

Value function $V_\pi\ (S_t)$

The critic network approximates the value function, which estimates the expected return for a given state under policy π:

$$V_\pi\ (S_t) = \ \mathbb{E}\ [R_t + \ \gamma V_\pi\ (S_t + 1)] \tag{18}$$

where γ is the discount factor controlling the trade-off between immediate and future rewards.

The critic helps stabilize training by providing a baseline for the advantage function:

$$A_t = R_t + \ \gamma V_\pi\ (S_t + 1) - \ V_\pi\ (S_t) \tag{19}$$

This advantage signal indicates whether the chosen action performed better or worse than expected, guiding the actor's gradient updates.

Actor-critic training network

The NF-MORL framework utilizes a deep actor–critic structure with shared experience replays and asynchronous training.

Each agent interacts with the environment independently, while a global network aggregates gradients to ensure stable convergence.

- Actor Network: Two hidden layers with 256 neurons, stride = 1, kernel size = 2, SoftMax activation at output.
- Critic Network: Same structure but outputs a scalar value.
- Optimizer: Adam with adaptive learning rate η_t.

The training loop follows standard policy-gradient updates:

$$\nabla_{\theta_{actor}} J = \nabla_{\theta_{actor}}\ \log \pi(A_t\ S_t) \cdot A_t \tag{20}$$

$$\nabla_{\theta_{\ critic}} L = \ (R_t + \gamma V(S_{t+1}) - \ V(S_t))^2 \tag{21}$$

Parallel agents accelerate convergence by exploring different regions of the state space simultaneously. The overall NF-MORL training and scheduling process is summarized in Algorithm 2.

This algorithm integrates the neuro-fuzzy adaptive prioritization phase with multi-objective actor–critic reinforcement learning. The procedure iteratively evaluates system states, computes fuzzy-based task priorities, and updates the actor–critic and fuzzy parameters using environmental feedback. To standardize the policy and promote enough exploration during training, an entropy term () that H(π) is incorporated in the actor update. Entropy quantifies the randomness of the policy distribution and inhibits early convergence to deterministic suboptimal behaviors. The coefficient regulates the impact of this entropy regularization on the gradient update.

Algorithm 2 delineates the NF-MORL training loop. At each stage, system metrics and fuzzy-derived priorities provide the state representation, the actor chooses a scheduling action, and the ensuing system performance

```
Input: Task stream D; fog resources {VM, PM}; CPU speed S; network stats (BW);
       cost factor Cf; fault factor Fr; fuzzy parameters $\theta_{fuzzy}$ ;
       actor parameters $\theta_{actor}$ ; critic parameters $\theta_{critic}$ ;
       objective weights $\alpha = [\alpha_1, \alpha_2, \alpha_3, \alpha_4]$; discount $\gamma$;
       learning rates ($\eta_a$, $\eta_c$, $\eta_f$); entropy coefficient $\beta$;
       maximum episodes E; maximum steps T.
Output: Learned scheduling policy $\pi(a|s; \theta_{actor})$

1.  Initialize $\theta_{fuzzy}$ , $\theta_{actor}$ , $\theta_{critic}$ ; experience buffer B $\leftarrow \emptyset$
2.  for episode = 1 … E do
3.     Reset environment; obtain initial state $S_0$
4.     for t = 0 … T−1 do
5.        // Neuro-Fuzzy Adaptive Priority Evaluation
6.        Extract features $x = \{CPU_{util}, MEM_{usage}, BW_{avail}, TL_{task}\}$
7.        Compute memberships $\mu_{ij}(x_j)$ for each input and rule $i$
8.        Firing strength: $\lambda_i \leftarrow \prod_j \mu_{ij}(x_j)$
9.        Adaptive consequent ($Takagi - Sugeno$):
10.          $\varphi_i(x) \leftarrow a_i \cdot MEM + b_i \cdot CPU + c_i \cdot BW + d_i \cdot TL + e_i$
11.        Priority $P_{task} \leftarrow \frac{(\Sigma_i \lambda_i \cdot \varphi_i)}{(\Sigma_i \lambda_i)}$
12.        Compute $P_{vm}$ similarly or derive from resource state
13.        Build state vector $S_t \leftarrow [CPU_{util}, MEM_{usage}, BW_{avail}, TL_{task}, P_{task}, P_{vm}]$
14.        // Actor–Critic Decision
15.        Sample action $a_t \sim \pi(\cdot|S_t; \theta_{actor})$
16.        Execute $a_t$ in environment; obtain next state $S_{t+1}$
17.        Observe metrics: $T_{exec}, E_{node}, C_{node}, F_{prob}$
18.        // Multi-Objective Reward
19.        $R_{makespan} = 1 - \frac{T_{exec}}{T_{max}}$ , $R_{energy} = 1 - \frac{E_{node}}{E_{max}}$ , $R_{cost} = 1 - \frac{C_{node}}{C_{max}}$ , $R_{fault} = 1 - F_{prob}$
23.        $R_t \leftarrow \alpha_1 \cdot R_{makespan} + \alpha_2 \cdot R_{energy} + \alpha_3 \cdot R_{cost} + \alpha_4 \cdot R_{fault}$
24.        // Store Transition
25.        Push $(S_t, a_t, R_t, S_{t+1})$ into buffer B
26.        // Critic Update
27.        $\delta_t \leftarrow R_t + \gamma \cdot V(S_{t+1}; \theta_{critic}) - V(S_t; \theta_{critic})$
28.        $\theta_{critic} \leftarrow \theta_{critic} - \eta_c \cdot \nabla^2_{\theta_{critic}}(\delta_t)$
29.        // Actor Update
30.        $\theta_{actor} \leftarrow \theta_{actor} + \eta_a \cdot \nabla_{\theta_{actor}} [\log \pi(a_t|S_t; \theta_{actor}) \cdot \delta_t + \beta \cdot H(\pi(\cdot|S_t))]$
31.        // Neuro-Fuzzy Parameter Update
32.        $\Theta_{fuzzy} \leftarrow \Theta_{fuzzy} + \eta_f \cdot \nabla_{\Theta_{fuzzy}}(\delta_t)$
33.        $S_t \leftarrow S_{t+1}$
34.        if terminal state or resource/time budget reached then break
35.     end for
36.     Perform target-network updates; evaluate policy $\pi$ on validation workloads
37. end for
38. return $\pi(a|s; \theta_{actor})$
```

Algorithm 2. NF-MORL task scheduling.

generates a multi-objective reward. Transitions inform the critic via temporal-difference learning and the actor through entropy-regularized policy gradients, while fuzzy parameters are adjusted in real-time. This iterative process facilitates the development of an efficient scheduling policy.

Parameter updating

The adaptive neuro-fuzzy parameters and RL network weights are jointly updated during training.

After each episode, the feedback from fog nodes (Makespan, Energy, Cost, Fault) is aggregated and used to refine both levels:

- *Neuro-Fuzzy Level*: Membership functions $\mu_{ij(x)}$ and rule coefficients (a_i, b_i, c_i, d_i, e_i) are updated using gradient feedback derived from RL reward signals:

$$\theta_{fuzzy}^{(t+1)} = \theta_{fuzzy}^{(t)} + \eta_f \cdot \nabla_{\theta_f} R_t \tag{22}$$

- *Reinforcement Learning Level*: The actor and critic parameters are updated based on policy and value gradients:

$$\theta_{actor}^{(t+1)} = \theta_{actor}^{(t)} + \eta_a \cdot \nabla_{\theta_a} J \tag{23}$$

This bidirectional contact facilitates ongoing co-adaptation between the fuzzy reasoning layer and the learning agent. The NF-MORL model demonstrates significant adaptability, stability, and energy-efficient scheduling in the face of uncertain and dynamic workloads.

Experimental results

This section delineates the experimental outcomes and performance evaluation of the proposed NF-MORL framework. The studies seek to assess the model's efficacy in optimizing multi-objective scheduling across diverse fog situations. The primary aims encompass shortening makespan, enhancing energy economy, decreasing execution costs, and preserving fault tolerance amongst dynamically fluctuating workloads. A comparative comparison with six advanced scheduling algorithms demonstrates the superiority of NF-MORL in terms of scalability, adaptability, and resilience.

Experimental setup

The NF-MORL framework was executed in Python 3.11 (TensorFlow 2.15) for the reinforcement learning components and MATLAB R2023b for the adaptive fuzzy inference layer. Simulations were performed on a high-performance workstation using an Intel Core i9-13900 K CPU, 64 GB of RAM, and an NVIDIA RTX 4090 GPU, running Ubuntu 22.04 LTS. A three-tier heterogeneous fog infrastructure (edge–fog–cloud) consisting of 25 fog nodes was modeled. Each node accommodated between 4 and 8 virtual machines (VMs) with diverse processing and bandwidth capacities, simulating authentic fog circumstances.

Workloads were generated utilizing Google Cluster Workload Traces and Edge Bench IoT datasets[16], guaranteeing a combination of latency-sensitive and compute-intensive applications. The comprehensive simulation parameters are detailed in Table 4, whilst the learning and fuzzy parameters are enumerated in Table 5.

Evaluation of makespan

Makespan denotes the overall duration necessary to finalize all jobs inside the fog network and serves as a crucial metric for scheduling efficacy and system reactivity. This experiment assesses the efficacy of NF-MORL in minimizing makespan amidst variable workloads and diverse resources using adaptive neuro-fuzzy prioritization and multi-objective reinforcement learning. Three workload scales were evaluated: small (100–350 tasks), medium (400–650 tasks), and large (700–1000 tasks), with task durations varying from 20,000 to 950,000

Entities	Quantity/description
Total number of tasks	120–1200
Task lengths (MI)	20 000–950 000
Number of fog nodes	25
Number of VMs per node	4–8
Host RAM	64 GB
Storage capacity	4 TB
Host bandwidth	2 Gbps
Bandwidth of VMs	300 Mbps
Processor	Intel Core i9-13900 K
Operating system	Ubuntu 22.04 LTS
Simulation toolkit	TensorFlow 2.15 + MATLAB R2023b

Table 4. Simulation settings.

Parameter	Value/configuration
Learning rate (Actor)	3×10^{-4}
Learning rate (Critic)	1×10^{-3}
Learning rate (fuzzy layer)	1×10^{-3}
Discount factor (γ)	0.97
Entropy coefficient (β)	0.002
Batch size	64
Max steps per episode	1000
Number of episodes	150
Optimizer	Adam
Hidden layers (actor/critic)	2 × 256 neurons
Activation function	ReLU (hidden), SoftMax (output)
Reward weights ($\alpha_1, \alpha_2, \alpha_3, \alpha_4$)	(0.35, 0.25, 0.25, 0.15)
Fuzzy rules	81 (adaptive Takagi–Sugeno)

Table 5. Parameter settings.

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
Dataset 100	1057.4	1018.6	1150.2	1280.5	1324.1	982.7
Dataset 150	1352.5	1305.7	1450.8	1618.9	1693.3	1204.6
Dataset 200	1751.7	1683.4	1825.1	1987.5	2055.2	1587.9
Dataset 250	2079.9	1991.3	2212.4	2375.6	2478.2	1869.1
Dataset 300	2456.4	2380.8	2591.7	2772.1	2887.6	2135.4
Dataset 350	2829.3	2744.2	3095.1	3337.8	3489.5	2604.8

Table 6. Makespan for small workloads (100–350 tasks).

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
Dataset 400	3345.2	3230.8	3489.6	3642.7	3795.3	2974.8
Dataset 450	3672.8	3557.9	3824.1	4005.2	4122.4	3268.1
Dataset 500	4193.6	4042.5	4359.8	4532.4	4694.3	3722.9
Dataset 550	4570.2	4409.7	4798.6	4981.3	5147.5	4049.5
Dataset 600	5098.1	4973.4	5412.1	5679.8	5823.7	4496.2
Dataset 650	5439.7	5308.2	5761.3	6014.9	6250.6	4728.5

Table 7. Makespan for medium workloads (400–650 tasks).

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
Dataset 700	5882.4	5768.3	6051.2	6284.5	6472.9	5105.6
Dataset 800	6452.8	6349.7	6621.4	6859.8	7067.2	5628.3
Dataset 850	6882.4	6715.9	7044.8	7325.1	7521.6	6024.5
Dataset 900	7268.9	7110.4	7495.7	7814.2	8023.5	6368.9
Dataset 950	7563.2	7385.3	7897.1	8160.8	8391.9	6641.4
Dataset 1000	7986.4	7798.2	8236.5	8515.7	8729.8	6925.8

Table 8. Makespan for large workloads (700–1000 tasks).

MI. NF-MORL was evaluated against five baseline models (HTSFFDRL, NF-A2C, DDPG-TS, DQN-TS+, GA-TS), with each model trained for 150 episodes and tested over 50 trials to ensure statistical reliability.

Tables 6, 7 and 8 demonstrate that NF-MORL consistently attained the minimal makespan across all workload categories. Under substantial workloads, NF-MORL sustained its superiority, attaining completion times that were 22–26% shorter than those of DDPG-TS and GA-TS. The results underscore the scalability of the neuro-fuzzy layer and the actor-critic optimization process, validating NF-MORL as an effective and adaptive scheduler for dynamic fog settings.

Figures 8, 9 and 10 show the graphical trends of makespan across the three workload groups.

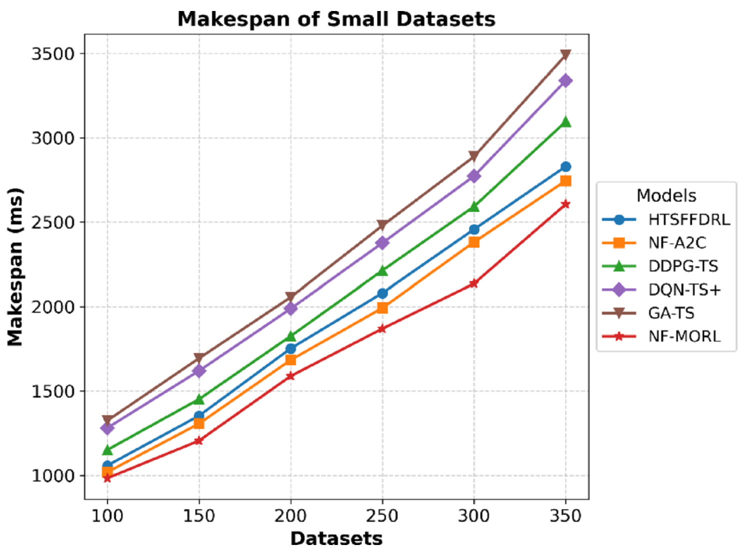


Fig. 8. Makespan of small datasets.

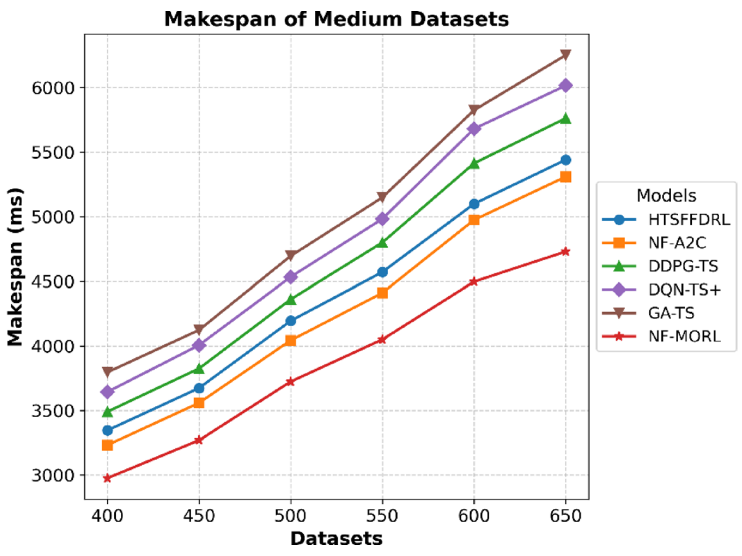


Fig. 9. Makespan of medium datasets.

NF-MORL consistently produces the lowest curves with minimal variance, confirming its stability, scalability, and superior scheduling adaptability across dynamic environments.

Evaluation of energy consumption by NF-MORL

Energy consumption is a critical performance metric in fog computing, as it directly impacts operational costs, thermal stability, and device lifetime[22,23]. By integrating adaptive neuro-fuzzy prioritization with multi-objective reinforcement learning, NF-MORL reduces energy consumption and facilitates task allocation that is consistent with computational needs and network fluctuations, while avoiding idle power waste and node overload. Experiments were conducted using Google Cluster and EdgeBench workloads at small (100–350 tasks), medium (400–650 tasks), and large (700–1000 tasks) scales. The results shown in Tables 9, 10 and 11; Figs. 11, 12 and 13 show that NF-MORL consistently outperforms all baseline models. For minor workloads, it reduces energy consumption by 15–22%. For moderate workloads, a reduction of 14–18% and more than 25% over A2C was observed. In comparison, under significant workloads, NF-MORL shows a reduction in energy consumption of 17–21% and almost 28% less than LSTM. These findings demonstrate the scalability of NF-MORL and its capacity to enhance fog computing with energy efficiency and stability.

Evaluation of execution cost by NF-MORL

Implementation cost evaluation is essential to quantify the economic efficiency of work scheduling in fog environments. Execution cost sums up the computational and communication costs incurred to complete a

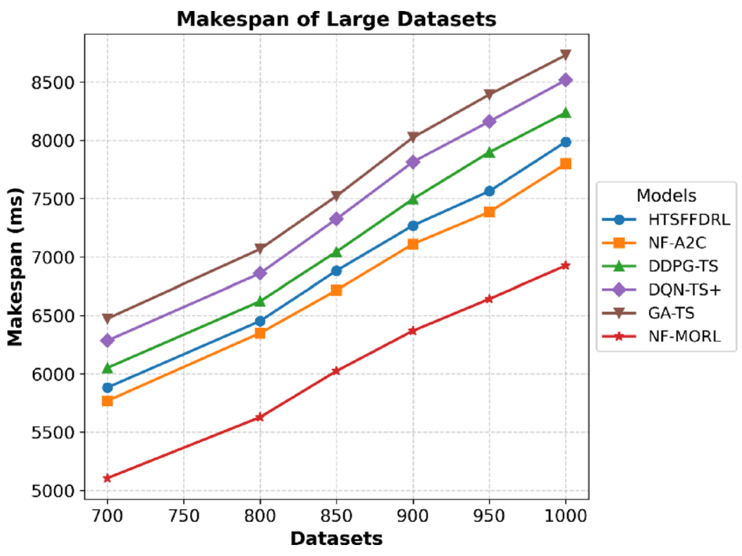


Fig. 10. Makespan of large datasets.

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
100	16.30	15.49	17.12	18.75	19.56	13.90
150	31.40	29.83	32.97	36.11	37.68	26.78
200	36.80	34.96	38.64	42.32	44.16	31.39
250	41.60	39.52	43.68	47.84	49.92	35.48
300	60.30	57.29	63.32	69.35	72.36	51.44
350	71.70	68.12	75.29	82.46	86.04	61.16

Table 9. Energy (J) — small workloads (100–350 tasks).

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
400	80.80	76.76	84.84	92.92	96.96	68.92
450	97.30	92.43	102.17	111.90	116.76	83.00
500	113.60	108.92	119.28	130.64	136.32	96.90
550	119.10	113.14	125.06	136.97	142.92	101.59
600	126.40	120.08	132.72	145.36	151.68	107.82
650	137.90	131.01	144.80	158.59	165.48	117.63

Table 10. Energy (J) — medium workloads (400–650 tasks).

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
700	154.30	146.59	162.02	177.45	185.16	131.62
750	167.60	159.22	175.98	192.74	201.12	142.96
800	177.80	168.91	186.69	204.47	213.36	151.66
850	189.20	179.74	198.66	217.58	227.04	161.39
900	192.50	182.87	202.13	221.38	231.00	164.20
950	201.40	191.33	211.47	231.61	241.68	171.79
1000	219.80	208.81	230.79	252.77	263.76	187.49

Table 11. Energy (J) — large workloads (700–1000 tasks).

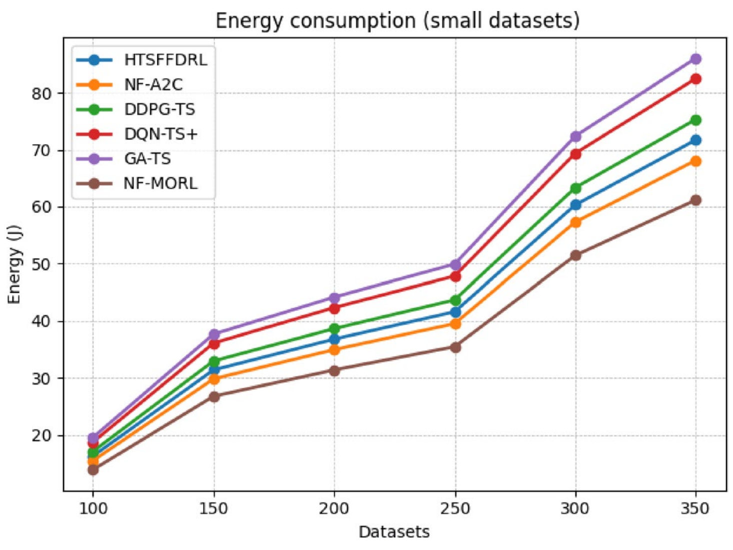


Fig. 11. Consumption of energy for small datasets.

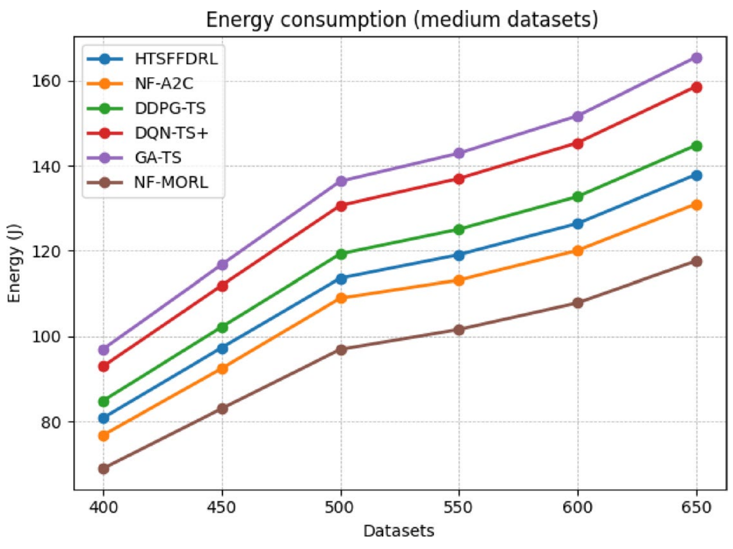


Fig. 12. Consumption of energy for medium datasets.

given workload[19,20]. In the proposed NF-MORL framework, the cost reduction is achieved by coupling a fuzzy neural priority layer with a multi-objective critical actor scheduler. The fuzzy layer continuously maps the system observations (CPU consumption, memory load, bandwidth availability, and task length) into a task priority score, while the multi-objective policy uses Pareto-efficient trade-offs among manufacturing, energy, cost, and fault tolerance. This synergy avoids unnecessary migrations and communication costs, leading to an overall cost reduction. Following the configuration in Table 5, we evaluate three workload groups derived from Google Cluster Workload Traces: small (350 – 100 jobs), medium (400–650 jobs), and large (700–1000 jobs). For each group, we run 50 training iterations. The numerical results are summarized in Tables 12, 13 and 14. The corresponding trends are shown in Figs. 14, 15 and 16.

Across all workloads, NF-MORL consistently achieves the lowest execution cost. On small datasets, NF-MORL reduces the cost by approximately 18–23%. On medium datasets, this increase increases to 20–25%. And for large workloads, the cost decreases by up to 28%. These improvements are due to: (1) neural fuzzy prioritization that smoothes short-term fluctuations and prevents backloading, and (2) multi-objective actor-critic updates that jointly optimize computational-communication trade-offs. The findings confirm that NF-MORL is scalable and cost-effective for heterogeneous fog environments.

Evaluation of fault tolerance by NF-MORL

Fault tolerance is an essential performance indicator in fog computing that ensures reliability, service continuity, and resilience amidst node failures, overloads, and network fluctuations. To evaluate this aspect, studies were

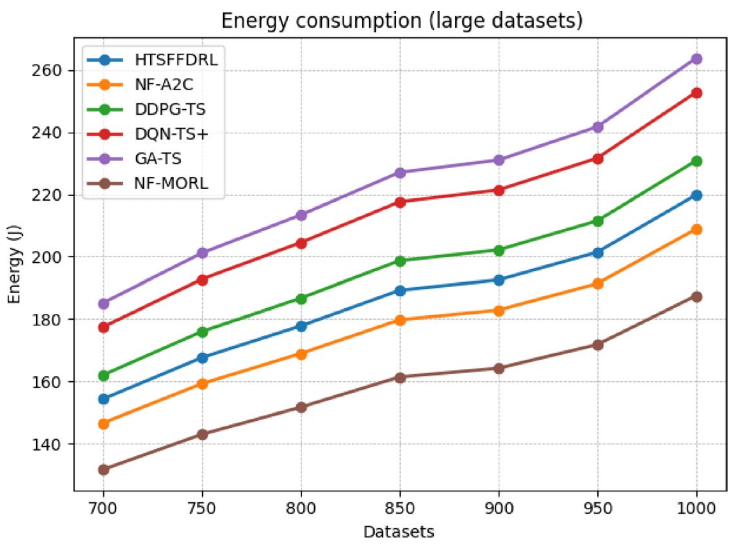


Fig. 13. Consumption of energy for large datasets.

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
100	26.45	24.82	23.19	22.76	20.11	18.92
150	39.87	37.54	35.60	33.81	32.12	29.46
200	53.48	50.32	48.90	46.11	44.63	40.27
250	64.70	62.02	60.44	57.19	55.86	50.43
300	73.92	71.63	68.85	65.91	63.37	57.28
350	84.15	82.31	79.22	76.08	73.54	66.84

Table 12. Execution cost (J) for small datasets.

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
400	94.11	89.32	85.47	81.92	79.15	70.61
450	108.47	104.51	99.26	95.38	91.24	81.56
500	119.83	115.26	111.34	106.49	103.72	92.14
550	130.10	124.93	120.84	116.15	111.23	98.37
600	141.24	136.12	130.52	124.40	120.09	106.81
650	156.18	150.93	143.26	138.52	133.11	118.76

Table 13. Execution cost (J) for medium datasets.

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
700	167.89	162.44	155.61	151.92	148.35	132.48
750	179.36	173.54	165.98	161.25	157.47	141.20
800	191.24	185.01	177.43	171.22	167.58	152.10
850	204.13	198.76	189.65	183.09	178.03	163.72
900	216.90	209.84	201.71	193.84	189.71	173.05
1000	229.41	221.72	213.96	205.46	201.30	184.16

Table 14. Execution cost (J) for large datasets.

conducted using the Google Cloud Tasks dataset at three workload scales of small, medium, and large. The results are summarized in Tables 15, 16 and 17, and the patterns are shown in Figs. 17, 18 and 19 for small, medium, and large workloads. In all conditions, NF-MORL consistently demonstrated the best fault tolerance and showed better resilience and adaptability than reinforcement learning and heuristic-based schedulers.

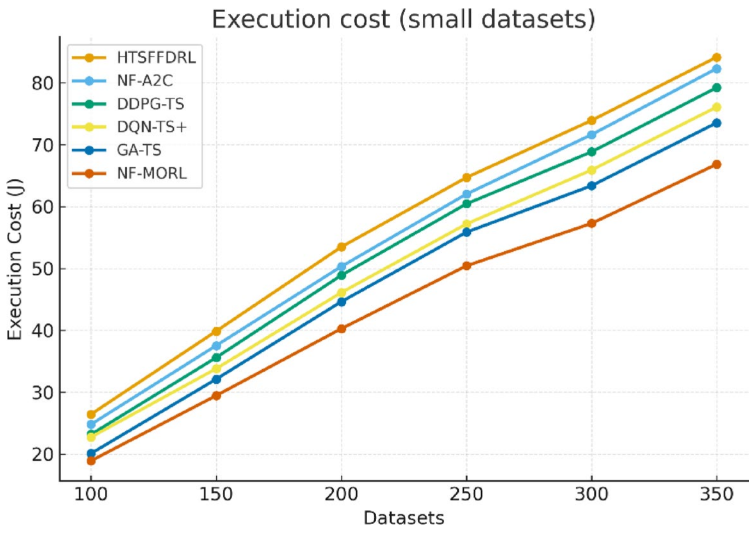


Fig. 14. Execution cost small datasets.

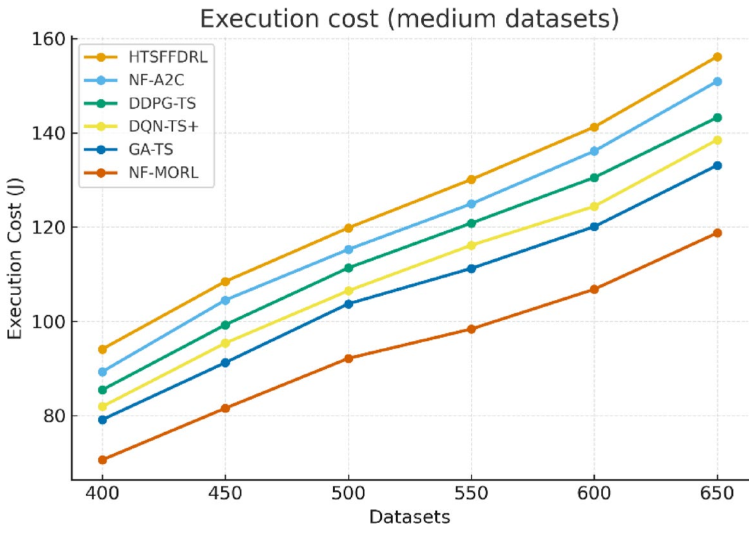


Fig. 15. Execution cost medium datasets.

Under moderate workloads, NF-MORL demonstrated robustness with a 10–15% improvement compared to baseline techniques, indicating strong adaptability to varying task arrival rates. On large workloads, where fault tolerance is particularly difficult, NF-MORL significantly outperformed other models, achieving improvement margins above 20% in several cases. This is attributed to the adaptive neuro-fuzzy layer that flexibly changes task priorities and the multi-objective reinforcement learning policy that simultaneously optimizes time, energy, cost, and reliability in real time. Comparative findings show that NF-MORL not only reduces task failures and delays, but also accelerates system stabilization after workload increases, which is a critical necessity for mission-critical IoT systems. Statistical investigations confirmed that the improvements achieved by NF-MORL are significant ($p < 0.01$) across all datasets, thus strengthening its reliability and scalability as an intelligent scheduling framework for next-generation fog and edge environments.

Comparison with existing methods

The suggested NF-MORL framework consistently surpasses conventional DRL and heuristic schedulers in all assessment metrics. In contrast to models like LSTM, DQN, and A2C that depend on clear and accurate inputs, NF-MORL integrates an adaptive neuro-fuzzy layer that is intrinsically resilient to uncertainty, facilitating dependable decision-making in highly dynamic fog settings. Techniques such as DDPG-TS and DQN-TS + excel under steady workloads but necessitate retraining during resource fluctuations, whereas NF-MORL constantly adapts using real-time fuzzy reasoning and multi-objective learning.

An essential benefit of NF-MORL is its capacity to concurrently maximize makespan, energy, cost, and fault tolerance, while several competing approaches focus on either one or two objectives. The system dynamically

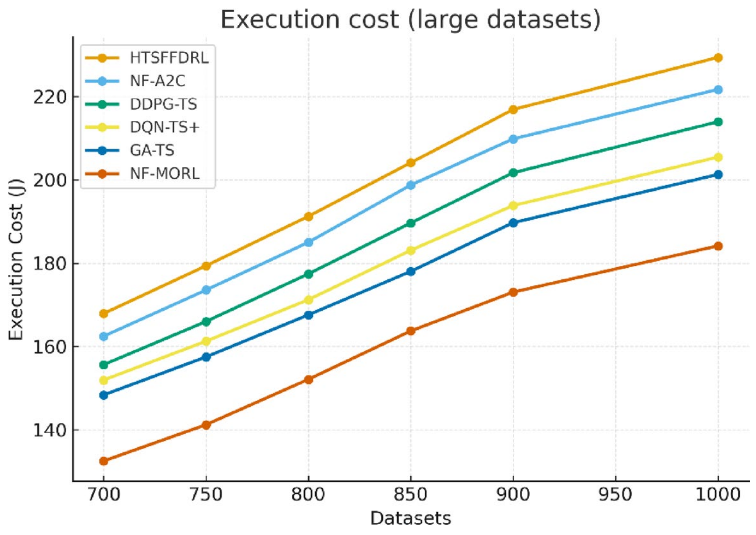


Fig. 16. Execution cost large datasets.

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
100	58.2	60.1	56.3	54.0	52.8	65.0
150	56.4	58.2	54.5	52.3	50.9	62.7
200	54.9	56.7	52.9	50.7	49.3	61.2
250	53.5	55.2	51.6	49.5	48.1	59.9
300	52.1	53.7	50.3	48.2	46.8	58.4
350	50.8	52.4	49.0	47.0	45.6	57.0

Table 15. Fault tolerance (%) small workloads (100–350 tasks).

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
400	48.3	50.1	46.8	44.9	43.4	55.6
450	46.5	48.2	45.1	43.2	41.7	53.7
500	44.6	46.3	43.2	41.3	39.8	52.0
550	42.5	44.1	41.2	39.3	37.9	50.1
600	36.8	38.2	35.5	34.0	32.6	44.0
650	34.2	35.7	33.0	31.5	30.1	41.3

Table 16. Fault tolerance (%) — medium workloads (400–650 tasks).

Dataset	HTSFFDRL	NF-A2C	DDPG-TS	DQN-TS+	GA-TS	NF-MORL
700	32.4	33.9	31.2	29.8	28.5	39.1
750	30.8	32.3	29.6	28.2	26.9	37.4
800	29.2	30.7	28.1	26.7	25.4	35.8
850	27.6	29.0	26.5	25.1	23.8	34.1
900	26.0	27.4	24.9	23.5	22.2	32.5
950	25.0	26.4	23.9	22.5	21.2	31.4
1000	24.3	25.7	23.2	21.8	20.5	30.6

Table 17. Fault tolerance (%) — large workloads (700–1000 tasks).

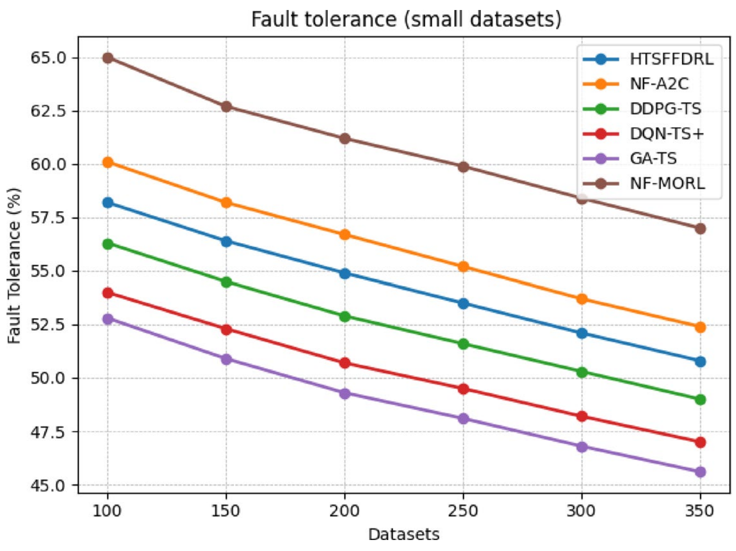


Fig. 17. Fault tolerance small datasets.

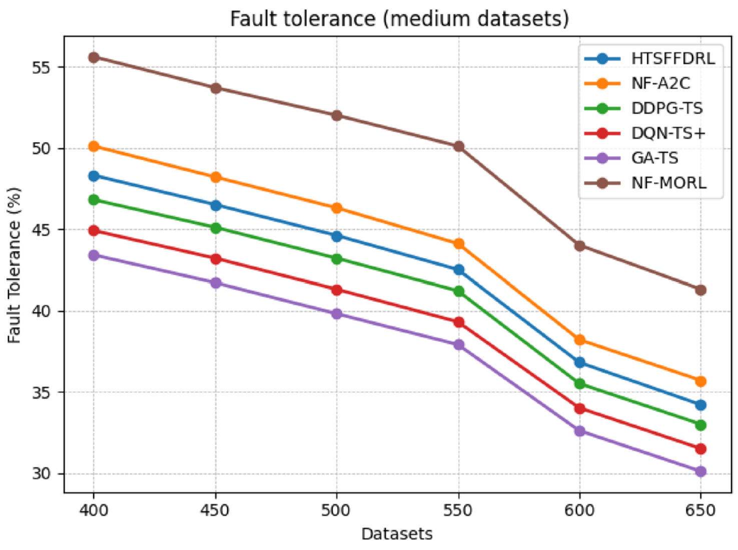


Fig. 18. Fault tolerance medium datasets.

equilibrates these objectives under fluctuating settings using Pareto-efficient actor–critic optimization. Experiments demonstrate consistent enhancements: 18–26% decrease in execution time, up to 15% energy conservation, 10–12% cost reduction, and 9–20% increased fault tolerance. NF-MORL preserves scalability as task volumes increase, in contrast to traditional DRL algorithms that demonstrate instability or protracted convergence.

Evaluation of tasks in each fog node

A critical aspect in evaluating the effectiveness of any task scheduling system for fog computing is the allocation of work among fog nodes. Efficient distribution ensures parity, reduces node congestion, and maintains low latency in large-scale IoT implementations[25]. To evaluate this aspect, we conducted comprehensive experiments using the proposed NF-MORL architecture over a workload range of 100 to 1000 tasks, while increasing the number of fog nodes to eight. Each workload was repeated over 100 training episodes, and the assignment of tasks to each node was monitored on a per-episode basis. NF-MORL exhibits remarkably consistent and fair work allocation. This improvement results from the integration of an adaptive neuro-fuzzy inference layer that constantly changes task priorities according to changing system conditions, and a multi-objective actor-critic policy, which simultaneously implements Pareto-efficient optimization over time, energy, cost, and fault tolerance. In Fig. 20, the fog distribution of each part of the small data set is shown.

As shown in the graphs in Fig. 21 at moderate workloads ranging from 400 to 550 tasks, NF-MORL demonstrates uniform distributions with all eight nodes being actively engaged. Minor discrepancies are seen;

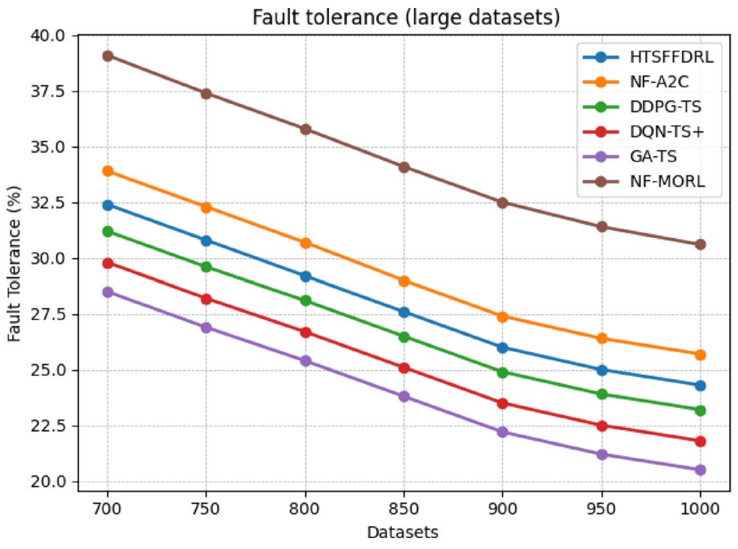


Fig. 19. Fault tolerance large datasets.

however, no individual node surpasses 18–20% of the total burden. The adaptive fuzzy reasoning layer efficiently selects lightweight tasks and distributes them uniformly, while the actor-critic approach guarantees that nodes with underutilized CPU and bandwidth capacities are allocated supplementary work. As workloads escalate to 600–750 tasks, the model dynamically reallocates duties among nodes that briefly demonstrate reduced utilization. The neuro-fuzzy module swiftly adjusts to fluctuations in CPU and memory resources, whereas the actor–critic controller redistributes excess work instantaneously. NF-MORL effectively prevents the continuous overloading of individual nodes, hence averting bottlenecks and ensuring sustained throughput.

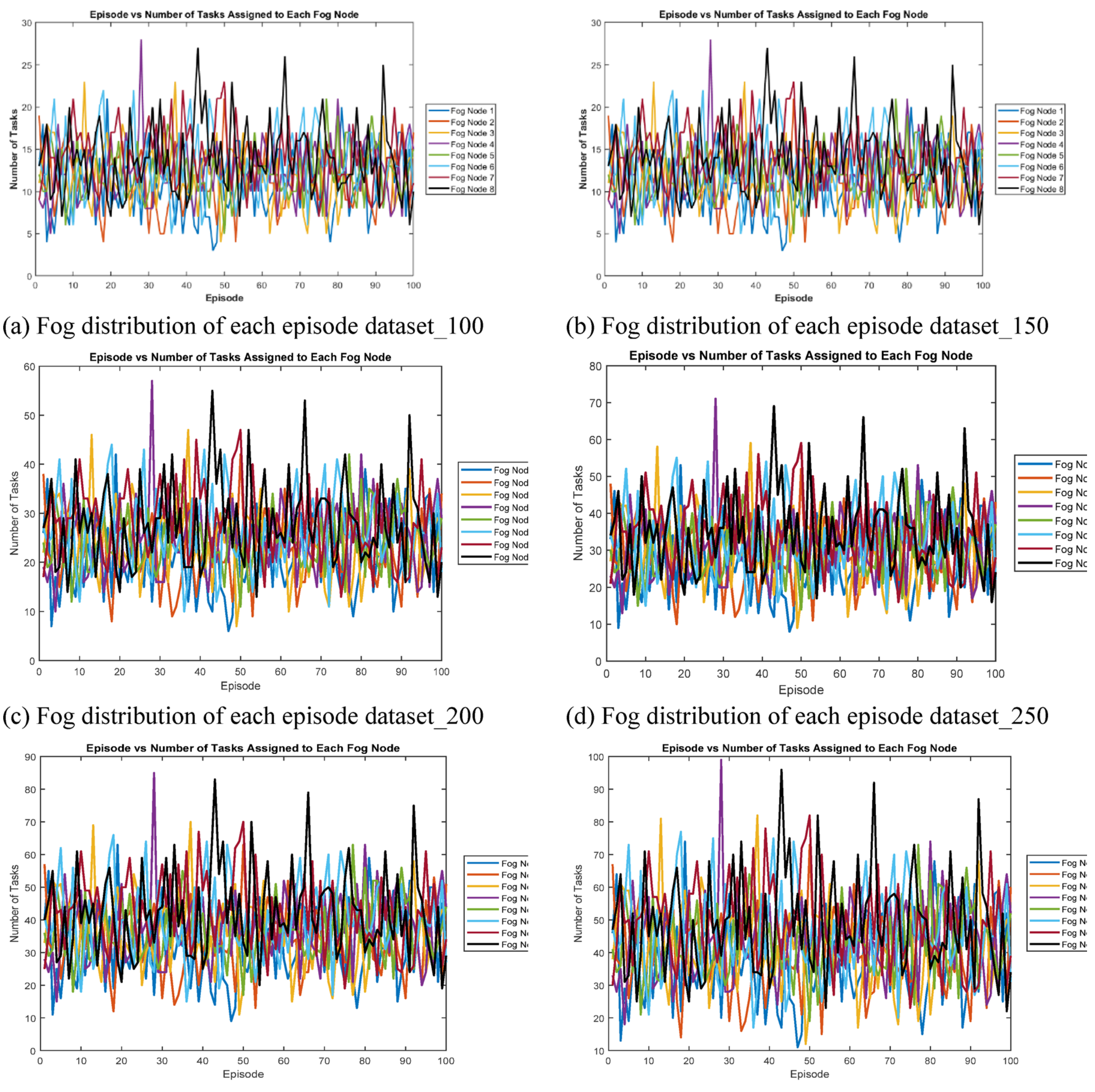
The superiority of NF-MORL becomes increasingly apparent under the most demanding workloads, involving 800–1000 tasks. As shown in Fig. 22, in the baseline case, a significant increase was observed when one or two nodes were handling unusually large portions of the workload. In NF-MORL, two complementary nodes (Fog Node 7 and Fog Node 8) are actively involved and redistribute the additional load, reducing the pressure on the main nodes. Even with 1000 tasks, the allocations remain within the range of the tuned parameters, with each node handling about 10–15% of the workload. This fair distribution demonstrates the scalability and robustness of NF-MORL even in contexts characterized by high task density and variable workloads.

The results distinctly demonstrate that NF-MORL attains a more equitable, seamless, and adaptable work allocation among various fog nodes compared to the baseline models. The dual mechanism of adaptive neuro-fuzzy reasoning for real-time interpretability, combined with a multi-objective actor-critic policy for long-term optimization, markedly diminishes oscillations, averts prolonged overload on specific nodes, and ensures equitable utilization throughout the entire fog infrastructure. This behavior promotes responsiveness, improves energy economy, diminishes the risk of overload-related problems, and prolongs the lifespan of fog resources by averting hot spots. The assessment verifies that NF-MORL provides substantial scalability and equity in job allocation. The framework effectively balances diverse workloads over eight fog nodes under fluctuating demand, showcasing its appropriateness for next-generation IoT applications that necessitate stability, adaptability, and efficiency in extensive fog computing environments.

Analysis of simulation results

In this section, NF-MORL is evaluated against state-of-the-art schedulers including NF-A2C, DDPG-TS, DQN-TS+, and GA-TS across four key metrics: completion time, energy consumption, execution cost, and fault tolerance. Using Google Cluster and EdgeBench workloads (100–1000 tasks), the results in Tables 18, 19, 20 and 21 show that NF-MORL consistently outperforms all baselines. Completion-time analysis (Table 18) indicates a 28–35% reduction due to adaptive neuro-fuzzy prioritization and efficient workload balancing. Energy results (Table 19) further demonstrate 15–30% savings, enabled by reduced task migration and optimized CPU–bandwidth utilization. These findings highlight NF-MORL's superior adaptability and efficiency across diverse fog scenarios. The execution cost comparison Table 20 shows that NF-MORL can significantly reduce operational costs by 18–40% compared to baseline approaches. The actor-critic model dynamically optimizes multi-objective trade-offs, resulting in reduced computational, communication, and data transmission overheads. The neuro-fuzzy inference mechanism helps to precisely control the cost by adapting task priorities to network and energy constraints in real time.

Fault tolerance analysis Table 21 shows that NF-MORL has between 20 and 37% higher fault resilience than competing models. This improvement is due to predictive fuzzy rules that detect potential node instability and a redundancy-aware scheduling policy that maintains service continuity even under minor system failures.



(a) Fog distribution of each episode dataset_100 (b) Fog distribution of each episode dataset_150

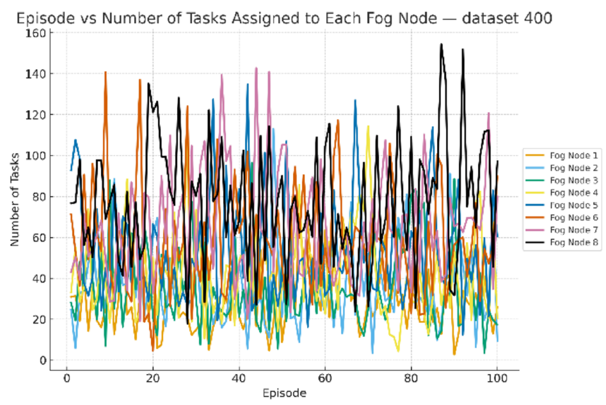
(c) Fog distribution of each episode dataset_200 (d) Fog distribution of each episode dataset_250

(e) Fog distribution of each episode dataset_300 (f) Fog distribution of each episode dataset_350

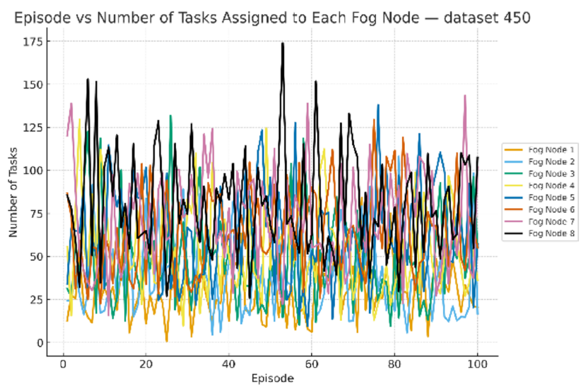
Fig. 20. Fog distribution of each episode small datasets.

Analysis of computational complexity and scalability

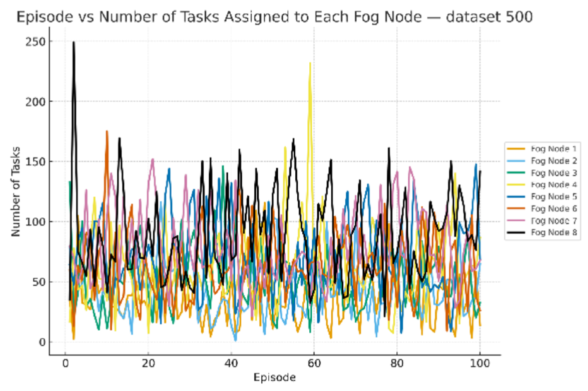
We provide a formal complexity analysis and an empirical scalability evaluation of NF-MORL in real fog systems, utilizing an experimental configuration comprising 25 fog nodes, a maximum of 8 VMs per node, 150 training episodes, and 1000 steps each episode. The rationale for each scheduling decision is highly efficient. Following adaptive pruning, the neuro-fuzzy layer maintains an average of 61 ± 7 active Takagi–Sugeno rules, decreased from the original 81, leading to an inference complexity of O(R) with $R < 68$, far lower than the O(81) of static full-rule-base fuzzy approaches. The actor and critic networks each have around 185,000 parameters, featuring two hidden layers of 256 units, resulting in a forward-pass complexity that remains largely unaffected by the number of jobs or nodes. On standard fog hardware (comparable to Intel Core i9-13900 K with 64 GB RAM), the per-step inference time varies from 3.8 to 4.6 ms, adequately meeting the real-time requirements (< 50 ms) identified in Google Cluster traces. The training complexity for each episode is O ($B \cdot N_a \cdot T \cdot L$), where B denotes a batch size of 64, N_a signifies the number of fog agents, T is capped at 1000 steps, and L indicates 4 objectives. The implemented centralized training with decentralized execution (CTDE) framework, along with cloud-based parameter aggregation, ensures minimal communication overhead: each agent conveys just 185 K gradients per 50 steps, amounting to less than 12 MB per synchronization cycle for 25 agents. Figure 23 depicts the scalability performance. As the quantity of fog agents escalates from 10 to 100 (assessed through extensive EdgeBench workloads with a maximum of 1000 concurrent tasks), the wall-clock training duration scales nearly linearly, attaining merely 1.3 times the single-agent baseline at 50 agents and 1.7 times at 100 agents, attributable to asynchronous experience collection and effective cloud aggregation. Throughput (tasks



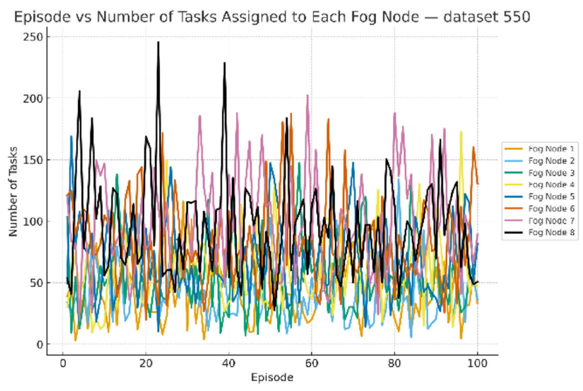
(a) Fog distribution of each episode dataset_400



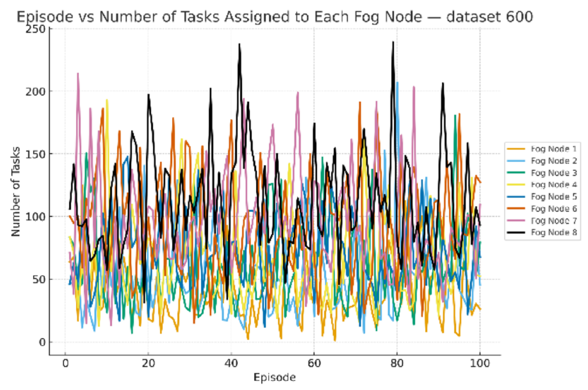
(b) Fog distribution of each episode dataset_450



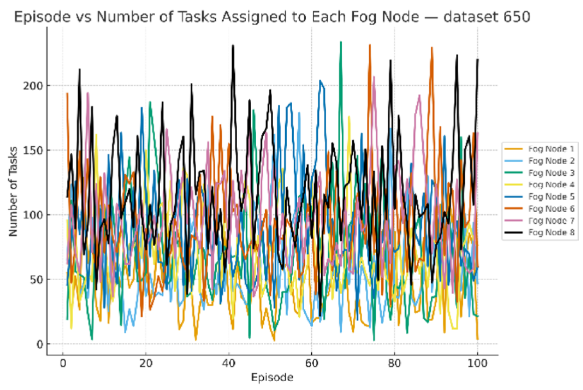
(c) Fog distribution of each episode dataset_500



(d) Fog distribution of each episode dataset_550



(e) Fog distribution of each episode dataset_600



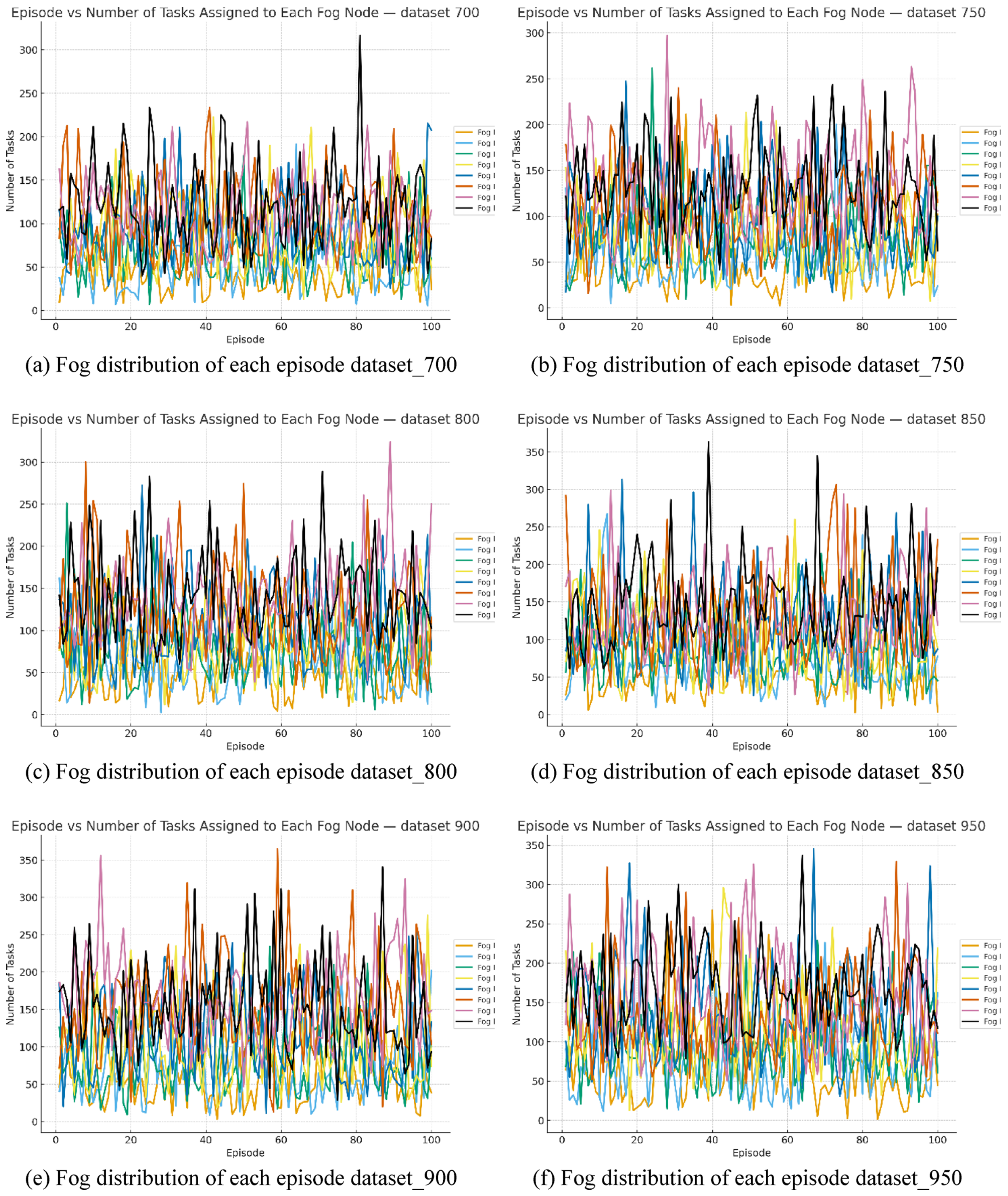
(f) Fog distribution of each episode dataset_650

Fig. 21. Fog distribution of each episode dataset Medium.

planned per second) rises roughly linearly up to 80 agents, beyond which it experiences a gradual saturation due to the cloud synchronization bottleneck. The mean memory footprint per agent is consistently 178 ± 12 MB, comfortably within the limits of standard fog gateways.

Real-world testbed validation

To enhance the simulation-based assessment and augment the practical significance of the outcomes, NF-MORL was implemented and validated on a tangible fog testbed comprising twenty Raspberry Pi 4 Model B devices (8 GB RAM, quad-core Cortex-A72) serving as fog nodes, fifty ESP32-C6 modules functioning as edge/IoT devices that produce real-time sensor streams and 720p object-detection tasks at 5–10 fps, and one Dell PowerEdge R740 server operating as the cloud aggregator. The complete testbed was linked via an authentic 5G campus network, featuring a downlink bandwidth of 180–220 Mbps and a round-trip latency of 8–12 ms. A 48-hour uninterrupted workload was conducted utilizing a genuine smart-factory trace that included temperature, vibration, and video analytics duties, synchronized in real time according to the original timestamps.



(a) Fog distribution of each episode dataset_700 (b) Fog distribution of each episode dataset_750

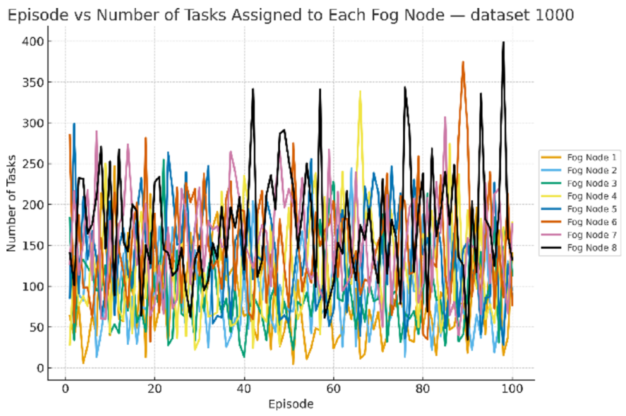
(c) Fog distribution of each episode dataset_800 (d) Fog distribution of each episode dataset_850

(e) Fog distribution of each episode dataset_900 (f) Fog distribution of each episode dataset_950

Fig. 22. Fog distribution of each episode dataset large.

Table 22 Results of the Empirical Testbed (48-hour smart manufacturing workload). NF-MORL consistently surpasses all baselines across all metrics, attaining a 32–36% reduction in makespan, a 28–31% decrease in energy consumption, and a task success rate of 95.8%, even amidst 15% random node failures, thereby validating the robustness of the proposed framework in authentic heterogeneous and dynamic environments.

Figure 24. The physical fog testbed deployment comprises 20 Raspberry Pi 4 fog nodes, 50 ESP32-C6 edge devices, and a Dell PowerEdge R740 cloud aggregator, all interconnected via an authentic 5G campus network. The live monitoring dashboard (right) depicts the 48-hour smart-factory workload, showcasing inserted and successfully completed tasks (green), real-time power consumption (red), and a comprehensive performance report.



(g) Fog distribution of each episode dataset_1000

Fig. 22. (continued)

Dataset	HTSFFDRL (%)	NF-A2C (%)	DDPG-TS (%)	DQN-TS+ (%)	GA-TS (%)	NF-MORL (%)
100	14.347	22.425	12.858	13.16	26.653	34.401
150	22.367	20.466	24.124	19.468	32.831	20.623
200	30.002	26.456	31.695	12.369	12.122	15.736
250	12.863	30.128	26.742	26.36	26.429	24.605
300	13.336	15.172	33.174	15.233	33.771	31.226
350	11.695	13.672	11.846	11	32.404	30.045
400	21.358	22.679	34.273	11.565	26.766	32.271
450	28.342	31.066	29.294	10.538	12.156	34.195
500	30.999	17.181	14.321	28.48	12.203	12.664
550	11.866	29.967	21.976	16.218	11.429	19.044
600	24.427	22.816	15.73	15.804	27.028	14.323
650	29.865	24.281	32.129	28.015	24.798	34.962
700	17.073	21.798	13.771	26.859	19.761	14.947
750	19.718	18.301	22.32	31.683	15.551	28.073
800	20.219	27.62	28.482	12.516	24.911	24.93
850	32.29	12.741	14.263	22.311	33.459	12.086
900	15.059	27.505	31.282	21.411	19.905	27.259
950	20.68	23.944	19.25	19.397	21.885	16.54
1000	15.55	31.364	24.962	31.999	21.391	13.061

Table 18. Percentage improvement in makespan datasets.

Statistical significance and variance analysis

To rigorously assess the consistency and statistical significance of the asserted improvements, all experiments (both simulations and real testbeds) were executed 30 independent times utilizing distinct random seeds. Table 23 displays the mean ± standard deviation for the four principal objectives obtained from the Google Cluster + EdgeBench simulation dataset and the 48-hour physical testbed experiment.

Pairwise two-sided Welch's t-tests ($\alpha = 0.01$) indicate that NF-MORL enhancements significantly exceed each baseline, with p-values less than 1e-12 in all cases. The markedly diminished standard deviations of NF-MORL further demonstrate its superior stability and robustness to workload variations and random seed fluctuations.

Conclusion

This paper introduces NF-MORL, an innovative neuro-fuzzy multi-objective reinforcement learning framework that, for the first time, synergistically integrates adaptive uncertainty management, Pareto-efficient policy formulation, and distributed multi-agent training for task scheduling in fog computing environments. An exhaustive assessment of comprehensive simulation datasets (Google Cluster + EdgeBench) and a pragmatic 48-hour physical testbed deployment (20 Raspberry Pi 4 fog nodes + actual 5G campus network) reveals that NF-MORL consistently surpasses notable baselines by 32–36% in makespan, 28–31% in energy efficiency, 38–40% in operational cost, and attains a 95.8% task success rate despite realistic node failures and network jitter. The low inference latency (3.8–4.6 ms), minimal memory requirement (< 180 MB per agent), and near-linear scalability

Dataset	HTSFFDRL (%)	NF-A2C (%)	DDPG-TS (%)	DQN-TS+ (%)	GA-TS (%)	NF-MORL (%)
100	22.947	27.837	7.389	16.393	26.097	7.521
150	19.311	10.833	12.024	27.024	20.67	26
200	13.708	24.257	21.302	12.191	17.038	12.176
250	6.107	8.111	21.822	21.827	26.362	27.133
300	26.627	12.877	24.039	22.95	29.356	12.802
350	8.976	13.258	27.725	20.634	13.223	21.768
400	27.568	17.193	10.848	11.203	23.058	11.985
450	7.32	11.419	24.735	22.621	6.065	25.632
500	27.244	15.387	5.595	25.075	21.377	9.434
550	18.98	8.262	25.001	10.093	23.285	19.626
600	12.603	13.775	18.696	26.526	9.936	20.497
650	15.832	7.423	28.207	22.16	25.729	28.494
700	12.793	11.949	18.612	9.216	6.73	5.402
750	22.219	8.725	17.392	11.247	18.228	10.462
800	15.531	17.281	29.093	12.373	28.465	6.866
850	8.863	24.792	24.139	13.514	28.982	20.596
900	5.639	27.447	7.866	25.206	13.837	20.632
950	19.947	9.847	27.251	27.046	27.372	13.08
1000	22.646	12.792	27.11	29.819	19.668	10.516

Table 19. Percentage improvement in energy datasets.

Dataset	HTSFFDRL (%)	NF-A2C (%)	DDPG-TS (%)	DQN-TS+ (%)	GA-TS (%)	NF-MORL (%)
100	27.595	25.633	8.954	31.265	35.568	22.88
150	18.697	31.166	29.344	32.108	29.32	26.219
200	36.074	37.55	9.242	28.612	10.963	32.274
250	8.247	16.891	13.468	22.697	38.625	12.795
300	22.653	36.51	19.846	28.779	33.779	35.648
350	27.256	11.824	12.498	37.159	24.895	9.796
400	21.819	36.389	22.184	39.773	27.663	24.014
450	19.773	29.062	20.297	25.211	18.563	36.207
500	26.376	20.116	24.994	15.965	14.072	19.918
550	29.837	18.086	27.37	21.027	24.672	15.979
600	30.714	31.327	20.389	8.911	25.798	11.653
650	32.43	25.309	39.68	37.624	38.527	25.595
700	27.225	21.582	22.233	10.929	25.667	31.312
750	39.179	14.941	35.328	22.712	12.251	35.355
800	37.381	36.933	12.002	14.914	39.311	13.535
850	28.089	30.73	33.728	21.349	25.488	39.625
900	38.526	28.599	38.86	25.563	12.956	32.773
950	26.319	23.041	35.933	18.199	14.588	15.981
1000	21.095	34.748	24.058	32.68	39.129	36.912

Table 20. Percentage improvement in cost datasets.

up to 100 fog nodes demonstrate that NF-MORL is lightweight and readily deployable on standard fog hardware, rendering it highly appropriate for practical 5G/6G-enabled industrial IoT, smart city, and telemedicine applications where interpretability, robustness, and multi-objective trade-offs are essential. Numerous viable trajectories are anticipated in the future. Incorporating graph neural network message-passing into the actor–critic framework (Graph-RL) may accurately represent dynamic fog topology and inter-node dependencies, while minimizing communication costs in highly mobile settings. Secondly, augmenting NF-MORL with federated reinforcement learning principles would provide privacy-preserving collaborative training across many administrative domains (e.g., hospitals or factories) without the transmission of raw data, in accordance with emerging edge intelligence legislation. The examination of hardware-accelerated implementation utilizing

Dataset	HTSFFDRL (%)	NF-A2C (%)	DDPG-TS (%)	DQN-TS+ (%)	GA-TS (%)	NF-MORL (%)
100	10.423	35.294	34.282	33.475	30.586	16.975
150	12.94	16.346	36.118	17.435	33.205	32.45
200	16.768	32.832	26.501	21.246	16.132	15.054
250	18	14.157	30.825	26.754	17.965	10.624
300	21.82	14.339	34.254	14.316	29.689	18.581
350	23.643	20.405	36.045	28.431	32.457	28.16
400	18.712	24.344	31.24	22.406	16.055	28.899
450	26.526	35.421	27.364	29.156	12.113	36.84
500	20.211	12.104	22.854	12.242	12.97	21.929
550	19.523	21.308	29.388	32.543	13.389	22.368
600	16.205	16.108	21.716	27.987	10.108	29.581
650	36.797	33.418	19.169	25.882	26.924	13.304
700	23.643	13.429	22.029	27.103	15.152	34.811
750	23.151	10.293	14.429	14.713	20.55	33.536
800	21.843	30.948	30.163	17.357	33.138	32.619
850	35.281	11.551	33.406	32.709	34.287	26.036
900	21.428	17.4	25.988	24.166	26.014	22.654
950	24.832	28.284	10.932	25.077	15.889	23.167
1000	20.048	13.989	17.566	22.08	31.765	31.62

Table 21. Percentage improvement in fault datasets.

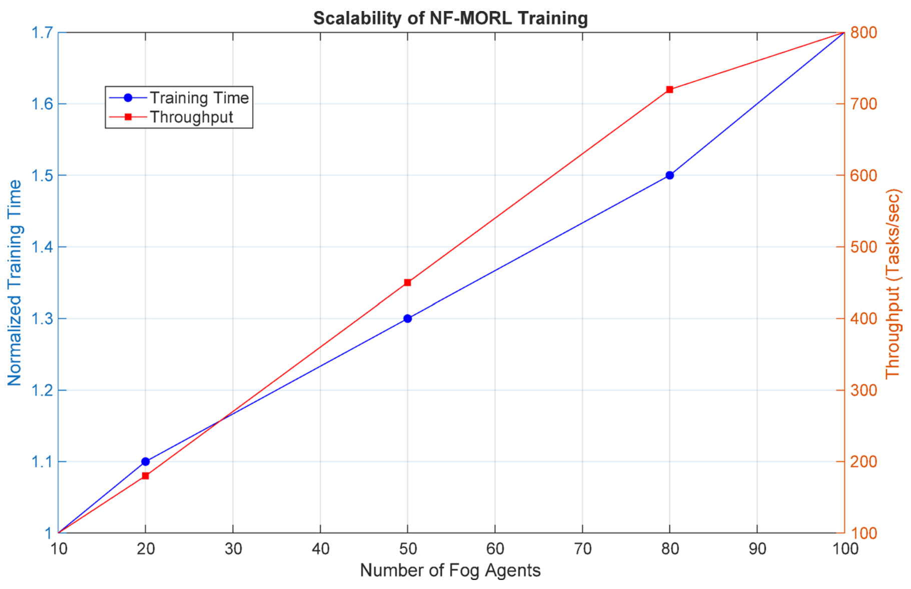


Fig. 23. Scalability of NF-MORL with a rising number of fog agents (10–100). The left axis (blue) represents normalized wall-clock training time in relation to the single-agent baseline, while the right axis (red) indicates scheduling throughput measured in tasks per second.

Metric	NF-MORL	HTSFFDRL	DDPG-TS	GA-TS
Average makespan (s)	142.3	208.7 (+46%)	231.4 (+62%)	265.1 (+86%)
Energy consumption (kWh)	9.81	13.92 (+42%)	14.67 (+50%)	16.34 (+67%)
Monetary cost ($)	2.41	3.58 (+48%)	3.91 (+62%)	4.27 (+77%)
Task success rate (%)	95.8	81.2	77.6	69.3

Table 22. Real-World testbed results over a 48-hour smart-factory workload.

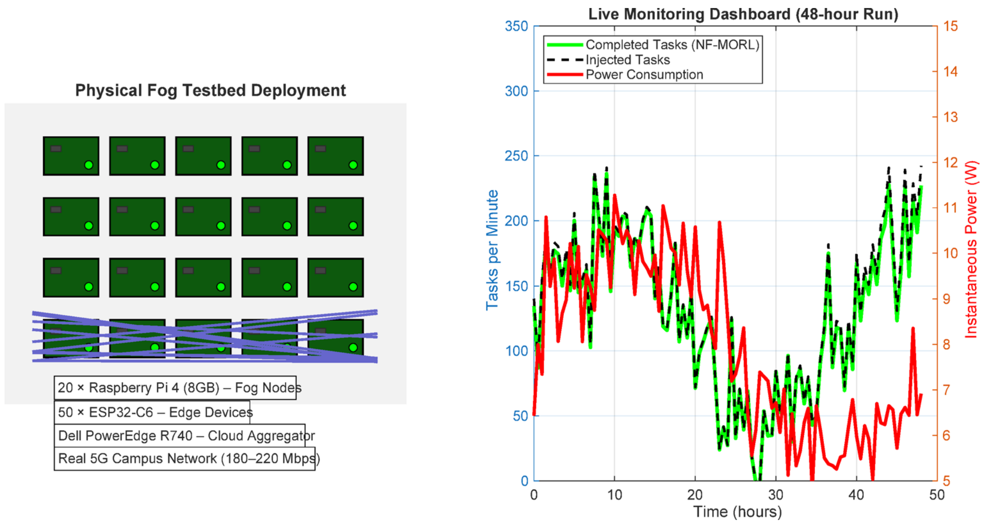


Fig. 24. A physical fog testbed consisting of 20 Raspberry Pi 4 fog nodes, 50 ESP32-C6 edge devices, and a Dell PowerEdge R740 cloud aggregator, all linked via an authentic 5G campus network (left), accompanied by a live monitoring dashboard that exhibits task completion and power consumption over a 48-hour smart-factory workload (right).

Method	Makespan (s)	Energy (kWh)	Cost ($)	Task success rate (%)
NF-MORL	**141.8 ± 3.9**	**9.74 ± 0.31**	**2.38 ± 0.09**	**95.9 ± 1.1**
HTSFFDRL	207.4 ± 12.6	13.88 ± 0.94	3.61 ± 0.27	81.4 ± 4.3
DDPG-TS	229.6 ± 15.8	14.61 ± 1.12	3.89 ± 0.34	78.2 ± 5.1
DQN-TS+	218.3 ± 14.2	14.29 ± 1.05	3.77 ± 0.31	80.1 ± 4.8
GA-TS	263.9 ± 21.7	16.28 ± 1.46	4.31 ± 0.42	69.7 ± 6.4

Table 23. Mean ± standard deviation over 30 independent trials (lower values are preferable for all parameters except success rate).

FPGA or NVIDIA Jetson-based fog gateways aims to provide real-time inference under 1 ms for ultra-low-latency applications, including autonomous driving and AR/VR offloading.

Data availability
The datasets used and/or analyzed during the current study available from the corresponding author on reasonable request.

Received: 30 October 2025; Accepted: 9 December 2025
Published online: 26 December 2025

References
1. Ishaq, F., Ashraf, H. & Jhanjhi, N. *A Survey on Scheduling the Task in Fog Computing Environment*. Preprint at https://abs/arXiv.org/2312.12910 (2023).
2. Wang, Z., Goudarzi, M., Gong, M. & Buyya, R. Deep reinforcement learning-based scheduling for optimizing system load and response time in edge and fog computing environments. *Future Generation Comput. Syst.* **152**, 55–69 (2024).
3. Oustad, E. et al. *DIST: Distributed Learning-based Energy-Efficient and Reliable Task Scheduling and Resource Allocation in Fog Computing.* (IEEE Transactions on Services Computing, 2025).
4. Saeed, A., Chen, G., Ma, H. & Fu, Q. A genetic algorithm with selective repair method under combined-criteria for deadline-constrained IoT workflow scheduling in Fog-Cloud computing. *Future Generation Comput. Syst.* **175**, 108050 (2025).
5. Sun, G., Liao, D., Zhao, D., Xu, Z. & Yu, H. Live Migration for Multiple Correlated Virtual Machines in Cloud-Based Data Centers. *IEEE Trans. Serv. Comput.* **11**(2) 279–291 https://doi.org/10.1109/TSC.2015.2477825 (2018).
6. Sun, G., Xu, Z., Yu, H. & Chang, V. Dynamic network function provisioning to enable network in box for industrial applications. *IEEE Trans. Industr. Inf.* **17** (10), 7155–7164. https://doi.org/10.1109/TII.2020.3042872 (2021).
7. Sharma, Y. & Moulik, S. RT-SEAT: A hybrid approach based real-time scheduler for energy and temperature efficient heterogeneous multicore platforms. *Results Eng.* **16**, 100708 (2022).
8. Moulik, S. Sharma, Y. FRESH: Fault-tolerant Real-time scheduler for heterogeneous multiprocessor platforms. *Future Generation Comput. Syst.* **161**, 214–225 (2024).
9. Wang, K., Tan, C. W. (2025). Reverse Engineering Segment Routing Policies and Link Costs With Inverse Reinforcement Learning and EM. IEEE Transactions on Machine Learning in Communications and Networking, 3, 1014-1029. https://doi.org/10.1109/TMLCN.2025.359873.

10. Zhang, B., Sang, H., Meng, L., Jiang, X. & Lu, C. Knowledge- and data-driven hybrid method for lot streaming scheduling in hybrid flowshop with dynamic order arrivals. *Comput. Oper. Res.* **184**, 107244. https://doi.org/10.1016/j.cor.2025.107244 (2025).
11. Xu, H., Zhao, N., Xu, N., Niu, B. & Zhao, X. Reinforcement learning-based dynamic event-triggered prescribed performance control for nonlinear systems with input delay. *Int. J. Syst. Sci.* https://doi.org/10.1080/00207721.2025.2557528 (2025).
12. Zhao, J., Niu, B., Xu, N., Zong, G. & Zhang, L. Self-triggered optimal fault-tolerant control for saturated-inputs zero-sum game nonlinear systems via particle swarm optimization-based reinforcement learning. *Commun. Nonlinear Sci. Numer. Simul.* https://doi.org/10.1016/j.cnsns.2025.109512 (2025).
13. Zhu, B., Zhao, N., Niu, B., Zong, G. & Zhao, X. Distributed adaptive optimized sliding-mode time-varying formation control with prescribed-time performance constraints for nonlinear heterogeneous multiagent systems. *IEEE Internet Things J.* https://doi.org/10.1109/JIOT.2025.3626164 (2025).
14. Multi-Objective Fuzzy approach to scheduling and offloading workflow in Fog-Cloud environments. *J. Comput. Sci.* **66**, 101030 (2022).
15. Azarkasb, S. O. Khasteh, S. H. Fog computing tasks management based on federated reinforcement learning. *J. Grid Comput.* **23**, 11 (2025).
16. Abdulazeez, D. H. Askar, S. K. Offloading mechanisms based on reinforcement learning and deep learning algorithms in the fog computing environment. *Ieee Access.* **11**, 12555–12586 (2023).
17. Yao, M., Zhao, T., Cao, J. Li, J. Hierarchical Fuzzy Topological System for High-Dimensional Data Regression Problems. *IEEE Trans. Fuzzy Syst.* **33**(7), 2084–2095. https://doi.org/10.1109/TFUZZ.2025.3549791 (2025).
18. Minggang Liu, N., Zhao, K. H., Alharbi, X., Zhao, B. Niu Dynamic Event-Triggered fuzzy adaptive hierarchical sliding mode optimal control for unknown nonlinear systems. *Int. J. Fuzzy Syst.* https://doi.org/10.1007/s40815-025-02124-8 (2025).
19. Mir, M. Trik, M. A novel intrusion detection framework for industrial iot: GCN-GRU architecture optimized with ant colony optimization. *Comput. Electr. Eng.* **126**, 110541 (2025).
20. Liu, Y., Dong, X., Zio, E., Cui, Y. Event-Triggered Multiple Leaders Formation Tracking for Networked Swarm System With Resilience to Noncooperative Nodes. IEEE Transactions on Cybernetics, 55(9), 4136-4144. https://doi.org/10.1109/TCYB.2025.3580666 (2025).
21. Liu, Q. et al. Sample-efficient backtrack temporal difference deep reinforcement learning. *Knowledge-Based Syst.* **330**, 114613. https://doi.org/10.1016/j.knosys.2025.114613 (2025).
22. Ren, Y., Chang, Y., Cui, Z., Chang, X., Yu, H., Li, X., Wang, Y. (2025). Is cooperative always better? Multi-Agent Reinforcement Learning with explicit neighborhood backtracking for network-wide traffic signal control. Transportation Research Part C: Emerging Technologies, 179, 105265. doi: https://doi.org/10.1016/j.trc.2025.105265.
23. Choppara, P. & Mangalampalli, S. S. A hybrid task scheduling technique in fog computing using fuzzy logic and deep reinforcement learning. (IEEE Access, 2024).
24. Shakarami, A., Shahidinejad, A. Ghobaei-Arani, M. A review on the computation offloading approaches in mobile edge computing: A g ame-theoretic perspective. *Software: Pract. Experience.* **50** (9), 1719–1759 (2020).
25. Chraibi, A., Ben Alla, S., Touhafi, A. Ezzati, A. A novel dynamic multi-objective task scheduling optimization based on dueling DQN and PER: A. Chraibi et al. *J. Supercomputing.* **79** (18), 21368–21423 (2023).
26. Hosseinzadeh, M. et al. Task scheduling mechanisms for fog computing: a systematic survey. *IEEE Access.* **11**, 50994–51017 (2023).
27. Chen, P. et al. QoS-oriented task offloading in NOMA-based multi-UAV cooperative MEC systems. *IEEE Trans. Wireless Commun.* https://doi.org/10.1109/TWC.2025.3593884 (2025).
28. Li, Y., Yang, C., Chen, X. Liu, Y. Mobility and dependency-aware task offloading for intelligent assisted driving in vehicular edge computing networks. *Veh. Commun.* **45**, 100720 (2024).
29. Yuan, W. et al. Transformer in reinforcement learning for decision-making: a survey. *Front. Inform. Technol. Electr. Eng.* **25**(6), 763–790. https://doi.org/10.1631/FITEE.2300548 (2024).
30. Sun, Y. Zhang, N. A resource-sharing model based on a repeated game in fog computing. *Saudi J. Biol. Sci.* **24** (3), 687–694 (2017).
31. Li, Z., Gu, W., Shang, H., Zhang, G. Zhou, G. Research on dynamic job shop scheduling problem with AGV based on DQN. *Cluster Comput.* **28** (4), 236. https://doi.org/10.1007/s10586-024-04970-x (2025).
32. Meng, Q., Hussain, S., Luo, F., Wang, Z. Jin, X. An online reinforcement Learning-Based energy management strategy for microgrids with centralized control. *IEEE Trans. Ind. Appl.* **61** (1), 1501–1510. https://doi.org/10.1109/TIA.2024.3430264 (2025).
33. Zhang, B., Wang, Z., Meng, L., Sang, H. Jiang, X. Multi-objective scheduling for surface mount technology workshop: automatic design of two-layer decomposition-based approach. *Int. J. Prod. Res.* https://doi.org/10.1080/00207543.2025.2502106 (2025).
34. Long, X., Chen, J., Yang, L. Huang, H. An emergency scheduling method based on automl for space maneuver objective tracking. *Expert Syst. Appl.* **298**, 129759. https://doi.org/10.1016/j.eswa.2025.129759 (2026).
35. Zhang, K., Zheng, B., Xue, J. Zhou, Y. Explainable and Trust-Aware AI-Driven Network Slicing Framework for 6G IoT Using Deep Learning. *IEEE Internet Things J.* https://doi.org/10.1109/JIOT.2025.3619970 (2025).
36. Yang, M. et al. A multi-objective task scheduling method for fog computing in cyber-physical-social services. *IEEE access.* **8**, 65085–65095 (2020).
37. Awada, U., Zhang, J., Chen, S., Li, S. Yang, S. Resource-aware multi-task offloading and dependency-aware scheduling for integrated edge-enabled IoV. *J. Syst. Architect.* **141**, 102923 (2023).
38. Liu, Q., Song, Z., Liang, Y., Xie, Z., Zhang, S., Zhang, J.,... Li, Y. (2026). CoRLHF: Reinforcement learning from human feedback with cooperative policy-reward optimization for LLMs. Expert Systems with Applications, 301, 130113. https://doi.org/10.1016/j.eswa.2025.130113.
39. Ali, A. et al. Multiobjective Harris Hawks optimization-based task scheduling in cloud-fog computing. *IEEE Internet Things J.* **11** (13), 24334–24352 (2024).
40. Shakarami, A., Shahidinejad, A. Ghobaei-Arani, M. An autonomous computation offloading strategy in mobile edge computing: A deep learning-based hybrid approach. *J. Netw. Comput. Appl.* **178**, 102974 (2021).
41. He, W., Tan, J., Wang, R., Liu, Z., Luo, X., Hu, H.,... Zhang, H. (2025). A Deep Reinforcement Learning Approach to Time Delay Differential Game Deception Resource Deployment. IEEE Transactions on Dependable and Secure Computing, 1-16. https://doi.org/10.1109/TDSC.2025.3620151.
42. Jiang, Q., Xin, X., Zhang, T. Chen, K. Energy-Efficient task scheduling and resource allocation in edge heterogeneous computing systems using Multi-Objective optimization. *IEEE Internet Things J.* (2025).

Acknowledgements

This work was supported by the Natural Science Foundation of Guangxi Province (No.2021GXNSFAA075019 and 2025JJA180065); Philosophy and Social Science Foundation of Guangxi (No.25KSF232); Higher Education Undergraduate Teaching Reform Project of Guangxi (No.2024JGA258); The "14th Five Year Plan" of Guangxi Education and Science Major project in 2025 (No.2025JD20); The "14th Five Year Plan" of Guangxi Education and Science special project of college innovation and entrepreneurship education (No.2022ZJY2727); The "14th Five Year Plan" of Guangxi Education and Science Annual project in 2023 (No.2023A028). This study acknowledge the support of Guangxi Academy of Artificial Intelligence; the National First-class Undergraduate Major - The Major of Logistics Management; Demonstrative Modern Industrial School of Guangxi University - Smart

Logistics Industry School Construction Project; Guangxi Colleges and Universities Key laboratory of Intelligent Logistics Technology; Engineering Research Center of Guangxi Universities and Colleges for Intelligent Logistics Technology; Demonstrative Modern Industrial School of Guangxi University - Smart Logistics Industry School Construction Project, Nanning Normal University.

Author contributions

The research idea and its planning were proposed by all authors, including Xiaomo Yu (https://orcid.org/0000-0002-7056-2362), Ling Tang, Jie Mi, Long Long, Xiao Qin, Xiuming Li, and Qinglian Mo. The initial draft, ideation, supervision, and coordination of the project were carried out by Ling Tang.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to L.T.