Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

Cloud-edge hybrid deep learning framework for scalable IoT resource optimization

Umesh Kumar Lilhore[1,2*], Sarita Simaiya[1,3*], Yogesh Kumar Sharma[4], Anjani Kumar Rai[5], S. M. Padmaja[6], Khan Vajid Nabilal[7], Vimal Kumar[1], Roobaea Alroobaea[8] and Hamed Alsufyani[9]

Abstract

In the dynamic environment of the Internet of Things (IoT), edge and cloud computing play critical roles in analysing and storing data from numerous connected devices to produce valuable insights. Efficient resource allocation and workload distribution are vital to ensuring continuous and reliable service in growing IoT ecosystems with increasing data volumes and changing application demands. This study proposes a novel optimisation approach utilising deep learning to tackle these challenges. The integration of Deep Q-Networks (DQN) and Proximal Policy Optimization (PPO) offers a practical approach to addressing the dynamic characteristics of IoT applications. The hybrid algorithm's primary characteristic is its capacity to simultaneously fulfil multiple objectives, including reducing response times, enhancing resource efficiency, and decreasing operational costs. DQN facilitates the formulation of optimal resource allocation strategies in intricate and unpredictable environments. PPO enhances policies in continuous action spaces to guarantee reliable performance in real-time, dynamic IoT settings. This method achieves an optimal equilibrium between policy learning and optimisation, rendering it suitable for contemporary IoT systems. This method improves numerous IoT applications, including smart cities, industrial automation, and healthcare. The hybrid DQN-PPO-GNN-RL model addresses bottlenecks by dynamically managing computing and network resources, allowing for efficient operations in low-latency, high-demand environments such as autonomous systems, sensor networks, and real-time monitoring. The use of Graph Neural Networks (GNNs) improves the accuracy of resource representation, while reinforcement learning-based scheduling allows for seamless adaptation to changing workloads. Simulations using real-world IoT data on the iFogSim platform showed significant improvements: task scheduling time was reduced by 21%, operational costs by 17%, and energy consumption by 22%. The method reliably provided equitable resource distribution, with values between 0.93 and 0.99, guaranteeing efficient allocation throughout the network. This hybrid methodology establishes a novel benchmark for scalable, real-time resource management in extensive, data-centric IoT ecosystems, consequently enhancing system performance and operational efficiency.

Keywords Cloud load balancing, IoT edge networks, Deep Q-Networks, Proximal policy optimization, GNN, Reinforcement learning

*Correspondence:
Umesh Kumar Lilhore
umeshlilhore@gmail.com
Sarita Simaiya
saritasimaiya@gmail.com
Full list of author information is available at the end of the article

Introduction

In the current swiftly evolving technological environment, the Internet of Things (IoT) is revolutionising numerous industries by linking a growing array of devices across diverse sectors [1]. The capacity to effectively analyse and process the vast quantities of data generated by these interconnected devices can produce valuable insights and foster innovation. To proficiently handle the intricacies of data-driven applications, a robust cloud and edge computing infrastructure is essential. These systems mitigate the expenses associated with delivering real-time insights and services by supplying the necessary processing power and storage capacity to manage and analyse data from numerous IoT sensors [2].

As IoT networks expand exponentially, they present considerable challenges, especially regarding resource allocation and task distribution. Given the heightened accessibility of devices and the surge in data volume, it is imperative to devise adaptable and efficient strategies for managing computing and network resources. Due to the unpredictable characteristics of IoT applications, sophisticated strategies for workload management and resource allocation are essential for guaranteeing reliable and consistent service delivery [3].

Conventional resource management methods often inadequately address the evolving requirements of contemporary IoT systems. Traditional methods may have limitations in scalability, responsiveness, and real-time performance, resulting in inefficiencies and bottlenecks. To address these challenges, more advanced optimisation protocols are required. Deep learning offers a promising solution because of its ability to model complex, multidimensional data. Deep learning allows for more informed decision-making by identifying patterns and trends in the data, paving the way for more effective and adaptable resource management solutions [4].

Research questions and objectives

This study aims to look into the potential of advanced deep learning methodologies, such as DQN, PPO, GNN, and RL, to improve load balancing and resource allocation in cloud-based IoT edge ecosystems [5]. The main research questions are as follows.

- ➢ How can this hybrid methodology improve the system's capacity for real-time responsiveness, efficient scaling, and optimal resource allocation in dynamic IoT environments?
- ➢ How accurately tasks are assigned to the best available resources?
- ➢ How efficiently are resources allocated according to task demands and resource availability?
- ➢ How effectively are jobs executed without failures, considering all constraints?

The proliferation of IoT networks, characterised by a wide range of devices and massive data generation, presents significant challenges for resource optimization and real-time management. Conventional methods frequently overlook these systems' changing needs, resulting in inefficiencies and performance constraints [6]. The purpose of this study is to evaluate the efficacy of integrating DQN, PPO, GNN, and RL to improve resource management in this context. The main objectives of this research are:

- ➢ Determine how well the hybrid DQN, PPO, GNN, and RL approach can adapt resource allocation strategies to fluctuating workloads and data flows while minimising delays and preventing system overload [7].
- ➢ Examine how well the hybrid model scales with the number of IoT devices and data streams, ensuring consistent performance and responsiveness as the system expands.
- ➢ Investigate resource efficiency improvements, focusing on lowering operational costs and energy consumption while also ensuring equitable resource distribution across the network [8].
- ➢ Demonstrate the potential of this deep learning-based approach in addressing the challenges of modern IoT environments, ultimately improving system performance and operational efficiency.

Motivation

The growth of the Internet of Things (IoT) poses considerable challenges for the supporting systems, especially in handling the substantial data volumes generated by an increasing number of connected devices. Traditional resource management strategies, originally designed for stable and predictable environments, are now facing substantial challenges. Traditional approaches frequently fail to address the complexities inherent in modern IoT ecosystems, which are characterized by dynamic data flows and unpredictable device requirements. As a result, many systems experience inefficiencies, delays, and bottlenecks, reducing performance and increasing operational costs [9].

A major challenge in IoT involves real-time resource allocation across a distributed network of devices, ensuring system responsiveness and reliability as the network grows. The development of IoT applications, including smart cities and industrial automation, increases the demand for scalable and adaptive solutions. Current

approaches to resource management frequently demonstrate limitations in scalability and adaptability when faced with the challenges presented by complex and high-demand environments.

The present research seeks to address the shortcomings of existing resource management strategies through the development of a solution that is more efficient, scalable, and flexible. The aim is to create a system that enhances resource allocation in real-time, dynamically adjusting to the changing demands of a growing IoT network. This research seeks to improve the management of extensive IoT systems by optimizing resource utilization, minimizing operational costs, and enhancing overall system performance in response to the continuous evolution and expansion of these systems.

Key contribution

The key contribution of the complete work is as follows.

- *Hybrid Deep Learning Model:* The current work introduces an innovative hybrid deep learning model that integrates DQN, PPO, GNN and RL. This novel hybrid paradigm enhances the effective distribution of workload and resource utilisation in cloud-based IoT edge networks by incorporating the benefits of both methods to address complex issues in computing resource management.
- *Enhanced Resource Management:* To improve resource management, our model utilises DQN to acquire efficient strategies by engaging with the system. PPO enhances these techniques by maximising performance in continuous action spaces. This integration guarantees a higher level of adaptability and efficiency in resource management in dynamic settings.
- *Advanced Techniques Incorporated:* One sophisticated method is to integrate GNNs to enhance the precision of resource representation and comprehension within the system. Furthermore, scheduling based on reinforcement learning allows the model to adapt to changing workloads in real time, hence improving its capacity to coordinate resources efficiently.
- *Performance Enhancements:* Proposed model DQN-PPO-GNN-RL and existing model Adaptive Resource Management (ARM), Reinforcement Learning-based (RLB), and Ant Colony Optimization (ACO) are implemented on popular datasets. Google Cluster Data [10], Alibaba Cluster Trace [11] and Microsoft Azure Traces [12] using the iFogSim simulator. Our methodology has resulted in significant improvements in efficiency. More precisely, we have observed enhancements in the organisation of work schedules, a decrease in operating expenses, and a reduction in energy usage. These improvements lead to a resource management system that is both economically efficient and ecologically sustainable.
- *Balanced Resource Distribution:* This approach consistently achieves an optimal distribution of resources, resulting in equilibrium values between 0.93 and 0.99. Achieving an ideal allocation of resources across the network ensures the avoidance of congestion and the optimisation of operational efficiency.
- *Setting New Standards:* The present study sets a stringent benchmark for the effective and punctual management of resources in the IoT domain. The proposed method is an enhanced and effective strategy for resource management in data-intensive and dynamic scenarios, taking into account the evolving needs of modern IoT systems.

Structure of article

This paper starts with a thorough examination of the current research on the subject, establishing the context for our contributions by emphasising significant progress and areas that have not been adequately addressed. Subsequently, the text provides a comprehensive account of our materials and methodologies. The text begins by explaining the proposed hybrid deep learning model that integrates DQN, PPO, GNN and RL and then describes the specific methods and algorithms used in this model. We precisely define the parameters used to measure the performance of the model.

The experimental results section delineates the evaluation of numerous scenarios that were investigated, the addition of an ablation analysis to assess the impact of various model attributes, and our specially crafted simulation setup. Subsequently, a comprehensive examination of the results is provided, providing insightful assessments of the strategy's effectiveness. The essay concludes by giving a concise summary of our findings and suggesting potential lines of inquiry for future research to further our discovery and address any unresolved issues in the relevant scientific domains.

Literature review

This section covers the review of various existing research in the field of cloud computing.

Resource allocation in edge and IoT networks

Khani et al. [1] concentrate on employing deep reinforcement learning (DRL) to enhance resource allocation in multi-access edge computing (MEC). The research seeks to tackle the difficulties associated with dynamic and time-critical applications by presenting a novel Deep Reinforcement Learning (DRL) approach that improves

resource utilisation and minimises latency. Nonetheless, as the findings are derived from simulations, additional research is required to validate its efficacy in practical applications. Anbazhagan and Mugelan [13] investigate the application of reinforcement learning to enhance resource allocation in Narrowband IoT (NB-IoT) networks. Their methodology exhibits enhanced efficiency relative to conventional techniques. The study employs synthetic data to demonstrate significant enhancements in resource allocation; however, additional research in real-world contexts is essential to validate the practical efficacy of the method.

Bansal and colleagues [2] concentrate on service allocation in smart city IoT applications utilising deep reinforcement learning (DRL). Their objective is to minimise latency and enhance efficiency. Utilising actual smart city data, they illustrate that their methodology surpasses conventional techniques. Further investigation is required to evaluate its scalability and efficacy across various urban settings. Mangalampalli et al. [3] examine the potential of artificial intelligence to enhance resource allocation in cloud computing. They present a hybrid model that optimises scheduling and minimises operational expenses. Their assessment, utilising both simulated and accurate cloud data, demonstrates the model's efficacy; however, further investigation is required to evaluate its applicability across diverse cloud systems. Li et al. [4] provide an extensive analysis of the potential of reinforcement learning to revolutionise smart manufacturing. Their research underscores the potential for enhancing efficiency in manufacturing processes; however, the study lacks empirical validation, necessitating further investigation to evaluate its practical applications.

Resource management and optimization in iot and cloud environments

Aarthi et al. [5] investigate how combining neural networks and cloud computing improves social network modelling. Their hybrid approach improves scalability and performance by leveraging social network data. However, the study did not thoroughly investigate the model's application outside of social networks, indicating a need for additional research in broader contexts. Almudayni et al. [6] propose a new resource provisioning method for IoT networks based on bat algorithms. Their simulations show improved resource management efficiency, but more research is needed to determine how well this method performs in real IoT systems.

Mahapatra et al. [7] aim to improve energy efficiency and load distribution in fog and cloud computing settings. Their simulation outcomes demonstrate enhanced energy efficiency and load distribution. Nonetheless, additional validation is necessary to confirm the practicality of these models in real-world scenarios. Premalatha and Prakasam [8] examine energy-efficient resource allocation and fault tolerance in IoT fog computing networks. According to simulation data, their system enhances energy efficiency and reliability; nonetheless, further empirical research is required to corroborate these results. Rostami and Goli-Bidgoli [14] investigate load-balancing strategies for IoT networks, focusing on Software-Defined Networking (SDN). Their work combines various approaches and assesses their impact on quality of service (QoS). While their findings are insightful, they do not present new empirical models or breakthroughs in the field.

Trends and challenges in resource management for cloud, edge, and IoT networks

Aqeel et al. [15] examine methods to enhance task allocation in cloud-based Internet of Everything (IoE) systems within the healthcare sector. Load-balancing techniques enhance system efficiency. Their findings rely on IoE data pertinent to healthcare; however, additional research in actual healthcare environments is necessary to validate the results. Singhal et al. [16] propose a load-balancing solution for cloud computing utilising Rock Hyrax Optimization techniques. The methodology seeks to minimise energy consumption and enhance task allocation, supported by simulations that validate its effectiveness. Empirical testing is necessary to validate the method's efficacy and influence.

Muniswamy and Vignesh [17] examine the simultaneous optimisation of load balancing and resource allocation in cloud environments through container management. Their strategy enhances cloud resource management, as evidenced by simulations; however, further research is required to assess the model's applicability in various container environments. Recent studies by Ghorbian et al. [18], Ghobaei-Arani et al. [19], and Ebrahimi et al. [20] emphasise the challenges of resource management, especially within serverless computing environments. This study examines scheduling mechanisms, function placement, and cold start latency, which are essential elements of resource management in large-scale IoT networks. These studies highlight the need for more adaptable and pragmatic approaches to managing dynamic and diverse workloads in modern IoT environments.

This review discusses the ongoing advancements and challenges of resource management in IoT, edge, and cloud computing environments. Many studies have introduced novel methodologies, such as Deep Reinforcement Learning, Artificial Intelligence, and hybrid models; however, the majority rely on simulations, making it difficult to validate the practical efficacy of these techniques

thoroughly. Future research should concentrate on practical applications and additional empirical testing to confirm the scalability and efficiency of these methods in large, dynamic IoT ecosystems. Table 1 presents a comparative analysis of existing research studies with respect to the proposed method.

Materials and methods

This section covers the key methods used in the research, problem formulation, Solution by the proposed model, workings of the proposed model, and dataset details.

Problem formulation

Cloud-based IoT edge ecosystems are distinguished by their dynamic nature, demanding workloads, and decentralised foundation infrastructure [21, 22]. The features as mentioned earlier provide considerable difficulties in terms of resource allocation and capacity balancing. Conventional methods, such as static allocation and heuristic-based solutions, may be insufficient as they cannot adjust to sudden fluctuations in task demand promptly. Hence, the underutilisation of resources and wasteful energy use is widespread, resulting in substantial effects on system performance, scalability, and operational cost-effectiveness [14, 15, 23, 24].

A significant issue in cloud systems is the prevailing low utilisation rates of physical machines (PMs) in data centres, typically falling between 10 and 25%. Consequently, the majority of resources remain idle [16]. Although not in use, these inactive PMs nonetheless use energy, making up 70% of the overall energy consumption during peak periods. The lack of efficiency underscores the need for more sophisticated and adaptable resource management strategies that can enhance resource use while minimising energy usage and operational costs. Within cloud computing systems, multiple stakeholders have unique

Table 1 Comparative analysis of various existing research

Study	Techniques Used	Evaluation Tools	Performance Metrics	Datasets Used
Khani et al. (2024) [1]	Deep Reinforcement Learning (DRL)	Simulated Datasets	Resource Utilization Efficiency, Latency	Simulation-based IoT datasets
Anbazhagan & Mugelan (2024) [13]	Soft Actor-Critic RL	Synthetic Network Data	Network Efficiency, Resource Allocation	Synthetic NB-IoT Data
Bansal et al. (2022) [2]	Deep Reinforcement Learning (DRL)	Real-world Smart City Data	Latency, Service Quality	Smart city IoT datasets
Mangalampalli et al. (2023) [3]	Artificial Intelligence (AI)	Simulation, Cloud Data	Operational Costs, Resource Allocation Efficiency	Real-world cloud data
Li et al. (2023) [4]	Deep Reinforcement Learning (DRL)	Literature Review	Efficiency, Adaptability	Literature-based case studies
Aarthi et al. (2024) [5]	Neural Networks + Cloud Computing	Custom Cloud Setup	Scalability, Accuracy, Performance	Social Network Data
Almudayni et al. (2024) [6]	IoT-Mist Bat-Inspired Algorithm	Simulated IoT Data	Efficiency, Resource Consumption	Simulated IoT Data
Mahapatra et al. (2024) [7]	Energy-Aware Work Offloading	Simulation	Energy Efficiency, Load Distribution	Simulation Data (Fog-Cloud Systems)
Premalatha & Prakasam (2024) [8]	Energy-Efficient Resource Allocation, Fault Tolerance	Simulated Data	Energy Efficiency, Reliability	Simulated IoT-Fog Data
Rostami & Goli-Bidgoli (2024) [14]	SDN-Based Load Balancing	Literature Review	Quality of Service (QoS)	Literature-based IoT datasets
Aqeel et al. (2023) [15]	Artificial Intelligence for Load Balancing	Healthcare IoE Data	Task Allocation Efficiency	IoE Data in Healthcare
Singhal et al. (2024) [16]	Rock Hyrax Optimization	Simulation	Energy Efficiency, Task Distribution	Simulated Cloud Computing Data
Muniswamy & Vignesh (2024) [17]	Container Management Architecture	Simulation Data	Load Balancing, Resource Allocation	Cloud Container Data
Ghorbian et al. (2024) [18, 19]	Scheduling Mechanisms in Serverless Computing	Literature Review	Scheduling Efficiency, Performance	Literature-based Serverless Data
Ebrahimi et al. (2024) [20]	Cold Start Latency Mitigation	Literature Review	Latency Reduction	Literature-based Serverless Data
Proposed method	Deep Q-Networks (DQN), Proximal Policy Optimization (PPO), Graph Neural Networks (GNN), Reinforcement Learning (RL)	Simulation Platform (e.g., iFogSim, CloudSim), Real-world IoT data	Resource Allocation Efficiency, Latency, Operational Costs, Energy Consumption	Real-world IoT and Cloud datasets (Smart City, Healthcare, Industrial)

and sometimes conflicting objectives. Cost-effectiveness, performance, and timely job execution are of utmost significance to cloud customers (CCs). By contrast, cloud service providers (CSPs) give priority to the optimization of resource utilization, minimization of energy usage, and maintenance of profitability [17].

Key objectives

A practical approach to controlling resource allocation and load balancing in cloud IoT environments is proposed by combining two resilient deep reinforcement learning approaches, namely DQN and PPO, in a hybrid model. The incorporation of different methodologies allows the hybrid model to handle both discrete and continuous tasks suitably, hence effectively handling the complex and unpredictable demands of cloud IoT networks [25]. The hybrid DQN-PPO model suggested in this study introduces a multi-objective optimisation framework that efficiently handles the conflicting requirements of both groups by achieving a balance between optimal resource allocation and system performance.

Mathematical formulation

Our goal is to balance several objectives, such as optimising job scheduling efficiency, minimising task execution cost, optimal resource use, reducing energy usage, and minimising hardware cost [26–28]. The proposed model will incorporate the subsequent optimisation objectives, which are described below. Table 2 presents a detailed list of key symbols used in the mathematical formulation.

- Task scheduling efficiency

The schedule duration quantifies the overall time needed to carry out all tasks on their designated virtual machines. The goal is to minimise the most significant duration required to accomplish any given task, as defined in Eq. 1.

$$TEC = \sum_{j=1}^{N} \sum_{i=1}^{M} \left[\frac{TL_j}{CP_i^{Processor}} \right] \times [Cst_{i,k}] \quad (1)$$

- Total cost of task execution

Task execution cost pertains to expenses related to utilising virtual machine system resources, such as CPU, memory, and storage, for each specific task [29]. The objective is to reduce this spending, as defined in Eq. 2 substantially.

$$Sch_L = \underset{i=1}{\overset{M}{max}} \left[\sum_{j=1}^{N} \frac{TL_j}{CP_i^{Processor}} \right] \quad (2)$$

- Resource utilization efficiency

To enhance resource efficiency, our goal is to minimise idle time and enable the best possible use of resources

Table 2 Symbol and description used in the mathematical model

Symbol	Description
VM_i	Virtual Machine i, where i = 1, 2,……, m and m: Total m Virtual Machines
T_j	Task j, where j = 1,2,…,n and j: Total tasks
TL_j	Length for a particular Task j
CP_i^{cpu}	CPU Capacity for a particular virtual machine
CP_i^{Memory}	Memory capacity for a particular virtual machine
$CP_i^{Storage}$	Storage capacity for a particular virtual machine
$CP_i^{Processor}$	Processor capacity for a particular virtual machine
$Q_{i,k}$	Queue k, assigned to a virtual machine, Where: k∈(CPU, Memory, Storage)
H_i	Hardware cost for a particular virtual machine. It mainly includes cost of CPU, storage etc
EC_i	Energy consumption of VM iii
BC_i	Bandwidth capacity of a particular virtual machine
M	Total number of VMs
N	Total number of tasks
$Cst_{i,k}$	Total cost for a particular resource k, which includes CPU, memory and storage cost, on a virtual machine
Sch_L	Task schedule length
$R_{Utilization}$	Resource utilization
EN_{Cons}	Energy Consumption

[30]. The aim is to reduce the discrepancy between the total allocated resources and the resources that are being used, as defined in Eq. 3.

$$R_{Utilization} = [\sum_{i=1}^{N}\left(CP_i^{cpu} - \sum_{j=1}^{N}\frac{TL_j}{CP_i^{Processor}}\right)^2 + \left(CP_i^{Memory} - \sum_{j=1}^{N} Mem_j\right)^2 + \left(CP_i^{Storage} - \sum_{j=1}^{N} Storage\right)^2] \quad (3)$$

• Energy consumption

Estimating energy usage is a critical factor in cloud computing systems. Our study aims to minimise the total energy requirements of virtual machines [31]. A virtual machine's energy consumption is directly correlated with the length of time a job is being executed and the specific resources used, as defined in Eq. 4.

$$EN_{Cons} = \sum_{j=1}^{N}[\sum_{i=1}^{M}\left(En_i \times \frac{\text{TLj}}{CP_i^{Processor}}\right) + (Ideal_{En_i})] \quad (4)$$

• Hardware cost

Cloud service providers need to improve the efficiency with which they distribute their hardware resources [32]. As Eq. 5 defines, the expenses include the costs of purchasing and maintaining physical assets like the CPU, memory, storage, and network infrastructure.

$$Hardware_{Cost} = \sum_{i=1}^{M} HD_{Cost_i} \quad (5)$$

• Task-VM mapping efficiency (mapping function)

When assigning software tasks to virtual machines, certain constraints, such as compute power, memory, and storage needs, must be considered [33]. The task-to-VM mapping function is defined in Eq. 6.

$$Mapping(T_j, VM_j) = \begin{matrix} 1, if\ the\ task\ T_j, assigned\ to\ VM_j\ and\ all\ the\ constraints\ are\ satisfied\ relaetd\ to\ the\ capacity. \\ 0, else \end{matrix} \quad (6)$$

• Key constraints

This set of constraints guarantees that the optimization problem adheres to each virtual machine's resource capacities, task needs, and system restrictions [34].

➢ *CPU Capacity*: Its primary responsibility is to ensure that the total workload assigned to a virtual machine should not be higher than the CPU capacity CP_i^{cpu} As defined in Eq. 7 and 8.

$$\text{CPU Capacity} = (\text{Number of CPUs}) \times \left(\text{CPU Frequency}\right) \times (\text{Number of Cores per CPU}) \quad (7)$$

$$\sum_{j=1}^{N}\frac{\text{TLj}}{CP_i^{Processor}} \leq CP_i^{cpu} \quad (8)$$

Where $\forall_i \in (1,2,,3 \ldots\ldots\ldots\ldots\ldots\ldots M)$

➢ *Memory Capacity*:This restriction guarantees that the memory demands of all tasks allocated to a virtual machine (VM) do not exceed its available memory, as presented in Eq. 9.

$$\sum_{j=1}^{N} Memory_j \leq CP_i^{Memory} \quad (9)$$

➢ *Storage Capacity*: The limitation, as mentioned earlier, ensures that the aggregate storage needed by jobs remains within the storage capacity of the virtual machine as presented in Eq. 10.

$$\sum_{j=1}^{N} STO_j \leq CP_i^{Storage} \quad (10)$$

➢ *Bandwidth*: All tasks allocated to a virtual machine must not exceed the bandwidth capability of the specific VM, as presented in Eq. 11.

$$\sum_{j=1}^{N} \text{BCi} \leq \text{B}_i \quad (11)$$

Where $\forall_i \in (1,2,,3 \ldots\ldots\ldots\ldots\ldots\ldots M)$

➢ *Task Completion Time*: Each activity must be completed within a predetermined time limit to ensure timely completion as specified in Eq. 12.

$$\frac{\text{TLj}}{CP_i^{Processor}} \leq \text{T}_{Max} \forall_j \quad (12)$$

• Multi-objective optimization

The formulation of the ideal multi-objective optimisation [35] problem is given by Eq. 13.

$$MO_{opt} = Min(Sch_L, Hardware_{Cost}, EN_{Cons}) \quad (13)$$

The mathematical model introduced in this study aims to address the complex, dynamic, and diverse requirements of resource allocation and load balancing in cloud IoT environments. The goal is to deliver the best possible solutions for both cloud users and providers, considering several aspects such as job durations, resource capacities, execution costs, energy efficiency, and hardware requirements [23, 25].

Proposed model

The proposed approach combines DQN, PPO, GNN and RL to address the difficulties of resource allocation and load balancing in dynamic IoT contexts, as presented in Fig. 1.

DQN is used to learn optimal resource management methods by interacting with the environment and constantly refining decision-making using a reward-based system. This enables the model to adjust to the intricate and continually evolving characteristics of IoT ecosystems, where the amounts of data and corresponding resource needs can differ significantly [36]. PPO improves this process by optimising policies in continuous action spaces, leading to consistent and efficient policy updates, even in real-time. Combined with PPO's policy optimisation, the integration of DQN's ability to learn from complex contexts yields a robust system that dynamically adjusts resource allocation. The GNN implemented in the proposed model efficiently captures the relationships present in the underlying graph structures, rendering it highly suitable for the management of interconnected cloud and edge nodes. In contrast, Reinforcement Learning enables continuous learning and adjustment, therefore allowing the system to excel in ever-changing settings [18].

This combined approach enables the model to achieve three main goals: maximising resource use by avoiding both underutilisation and overexploitation, decreasing operational costs through intelligent energy consumption and minimising unnecessary resource expenditures, and ensuring rapid response times for real-time IoT applications such as smart city management, industrial automation, and healthcare monitoring. Collectively, these approaches form a comprehensive framework for handling the complexities of modern IoT systems, offering a firm answer for scalable and efficient resource allocation. Figure 1 presents the architecture of the proposed hybrid model.

Working of proposed model

The following section presents a detailed strategy for managing load balancing and resource allocation in Cloud IoT Edge environments using the hybrid paradigm. The efficiency and adaptability of the DQN architecture in resource allocation algorithms, together with the optimisation and stabilisation of load distribution rules in PPO, contribute to the successful handling of the dynamic characteristics of contemporary IoT configurations [19, 28, 36]. Figure 2 presents the workings of the proposed hybrid DQN and PPO model.

Resource allocation with DQN

DQN is precisely engineered to acquire the ability to efficiently distribute resources by engaging with the surroundings and optimising its tactics via the use of incentives. Figure 3 presents the process of DQN. It facilitates the identification of the most efficient allocation of resources according to the diverse requirements of various workloads and applications. DQN enables the adequate adjustment of resource allocations to meet the dynamic requirements of different jobs. The algorithm acquires knowledge of the allocations that result in optimal performance, therefore preventing both underutilisation and overexploitation of resources [16, 24]. Utilising the DQN, effective resource management strategies are developed through interaction with the environment. The algorithm considers factors such as resource availability, data loading volume, and network delay.

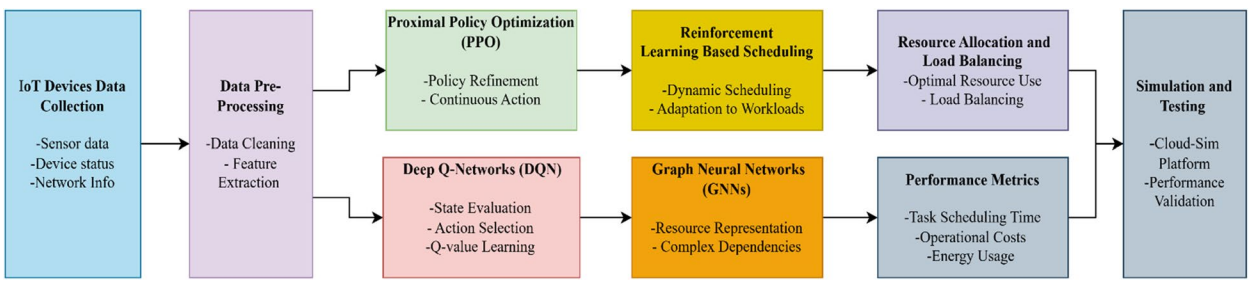

Fig. 1 Architecture of proposed hybrid model

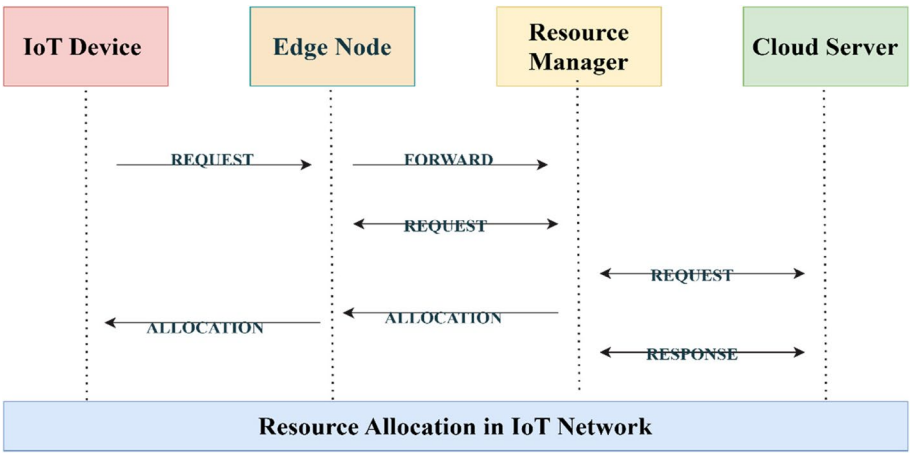

Fig. 2 Sequence diagram for working of proposed model

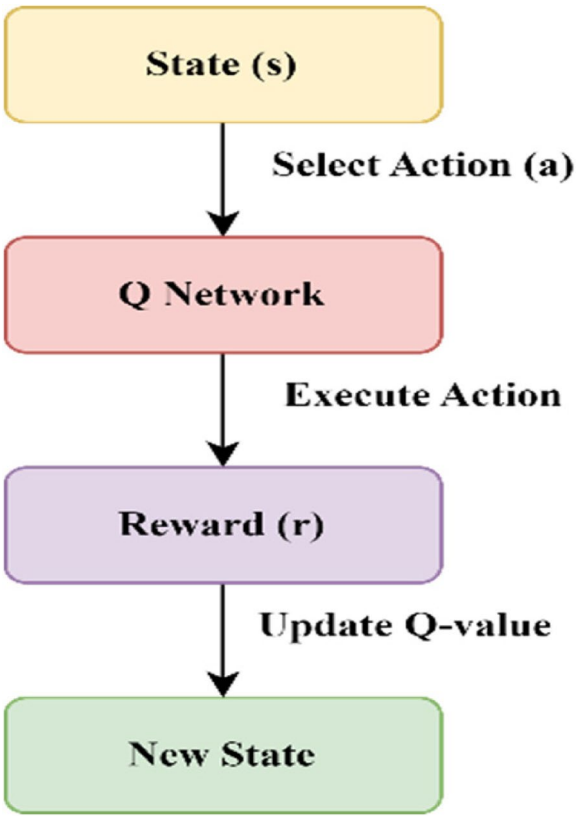

Fig. 3 DQN process

Furthermore, it produces actions that determine the distribution of resources between cloud and edge nodes.

The agent obtains information through repeated experimentation, enhancing its reasoning mechanism through a set of incentives that promote fair task allocation, efficient energy utilization, and minimum latency [8]. The complete workings of the DQN in load balancing are as follows.

➢ *Representation of State*: The current configuration of resources and workload needs across the system is represented by the state *s*.

➢ *Selection of Action*: Actions denote prospective distributions of resources. DQN employs an ε-greedy policy to achieve a balance between exploration.

- By using a probability Ω, choose a random action.
- Else, choose the action which has the highest Q value using Eq. 14.

$$A_{best} = Arg[Max_a Q(s, a, \Phi)] \tag{14}$$

➢ *Updating Q-Learning*: After performing an action and viewing the reward r and subsequent state *s*, adjust the Q-value by applying the Bellman Eq. 15.

$$y = [r + ¥ Max_a Q(s\prime, a\prime, \Phi^i)] \tag{15}$$

➢ *Calculation of Loss:* Optimize the loss function for efficient updating of the Q-network using Eq. 16.

$$Loss_{opt} = [(y - Q(s, a, \Phi))^2] \tag{16}$$

PPO for Policy optimization

PPO optimises policies in continuous action spaces. By circumventing substantial and disruptive alterations, this approach ensures consistent and efficient policy updates. An intrinsic characteristic of the method allows for equitable resource allocation in scenarios characterised by real-time fluctuations in workloads. PPO improves the model by adjusting rules to distribute resources and optimally distribute workloads, therefore ensuring that no edge or cloud node is either underutilised or overcommitted [19, 35]. Figure 4 presents the process for PPO. The complete work is as follows.

➢ *Representation of policy*: The likelihood of acting in a particular condition is ascertained by the policy $\pi_\theta(\mathrm{a}|\mathrm{s})$.
➢ *Collection of statistics*: Aggregate experiences by implementing the existing policy and saving the outcomes in a buffer.
➢ *Estimation benefits*: Calculate the advantage function $\widehat{\mathrm{A}}(s, \mathrm{a})$ Assess the relative effectiveness of activities in relation to the anticipated outcome by Eq. 17.

$$\widehat{\mathrm{A}}(s, \mathrm{a}) = [r + ¥ V_\Phi(\dot{s}) - V_\Phi(s)] \tag{17}$$

➢ Update policy: Achieve policy optimisation by utilizing the trimmed objective to avoid excessive updates using Eq. 18.

$$L^{PPO}(£_P) = E[min(r_t(£_P)A(s_t, a_t), clip(r_t(£_P), 1-\mathcal{E}, 1+\mathcal{E})A(s_t, a_t))] \tag{18}$$

➢ *Update Value Function:* Refine the value function to enhance the accuracy of predicting future rewards using Eq. 19.

$$L^V(\Phi) = E_t[(V_\Phi(s_t) - R_t] \tag{19}$$

GNNs

In the IoT ecosystem, GNNs are employed to enhance the depiction of resources. The ability to represent intricate connections among nodes (such as devices and servers) by considering them as a graph allows for the efficient capture of both local and global dependencies [5, 25]. GNNs offer a detailed perspective on the interconnections and interdependencies of resources, therefore improving the system's capacity to comprehend and forecast the availability and needs of resources inside the IoT architecture. Figure 5 presents the GNNs process. The complete details are as follows.

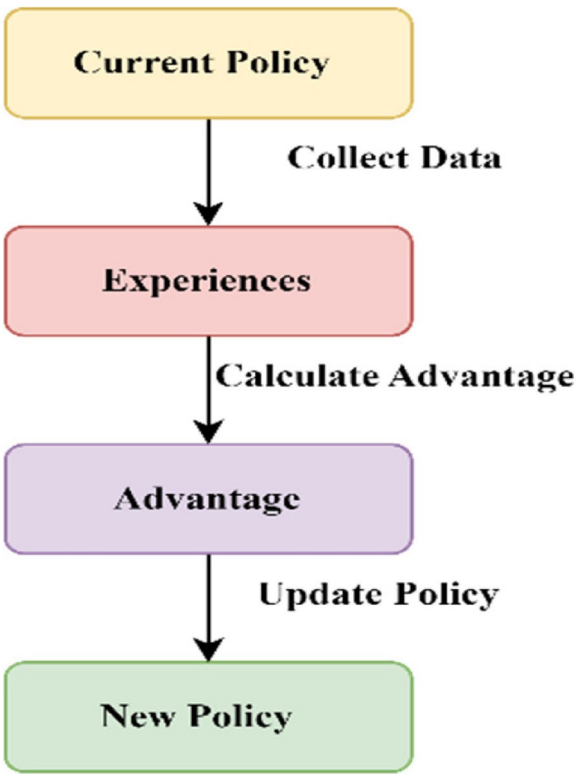

Fig. 4 PPO process

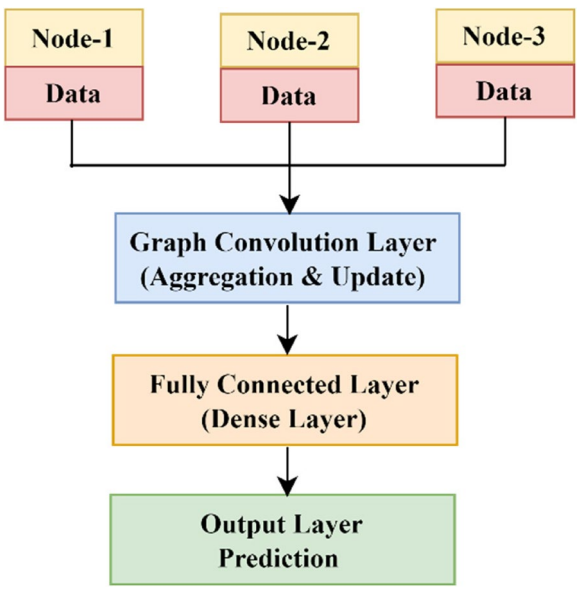

Fig. 5 The GNNs process

➢ *Representation of Graph*: Graphs are constructed by representing each device or server as a node. Given by G=(V, E), V represents the collection of nodes, and E represents the collection of edges that serve as representations of relationships or communication links.
➢ *The initialisation of Node features:* Initialise feature vectors FV_i^0 for each node V_i Dependent on their characteristics (e.g., capability of resources, present workload).
➢ *Message passing:* With each iteration k, nodes collect and consolidate information from their neighbouring nodes as presented in Eq. 20.

$$h_i^{k+1} = \sigma[(w) \times (\sum_{j\mathcal{E}Nb(i)} h_i^k + b)] \tag{20}$$

Here $Nb(i)$: set of neighbour's nodes i, w: Weigh matrix, b: bias vector, σ: Activation function.

➢ *Final Representation:* Following several iterations, every node V_i possesses a revised formulation h_k^i. This is the structural and feature information derived from the immediate graph neighbourhood.

Reinforcement learning

RL is the fundamental framework for both DQN and PPO. It enables the system to acquire knowledge from interactions with the environment by assimilating rewards obtained from performed actions [2, 7]. Figure 6 presents the complete process for reinforcement learning. The key elements are as follows.

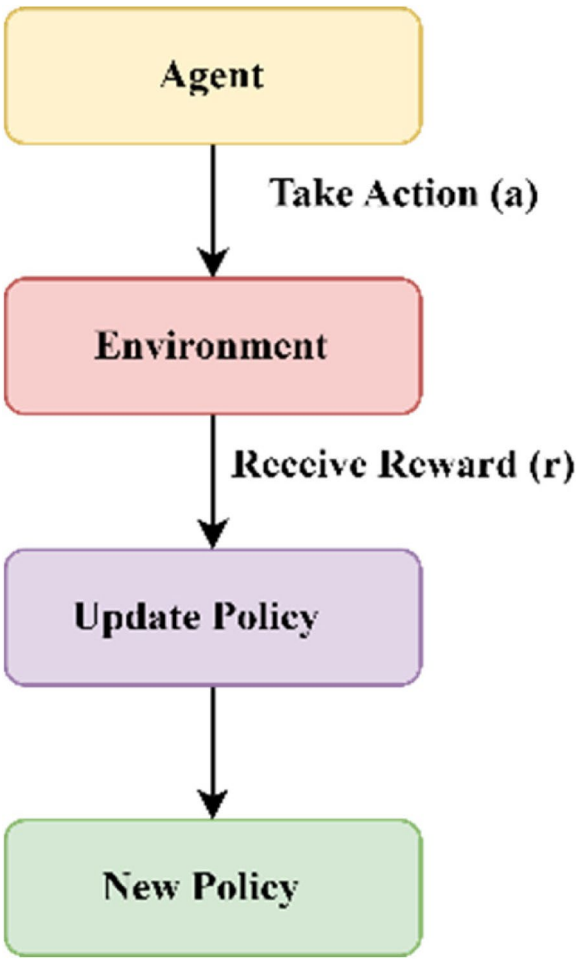

Fig. 6 The complete process for reinforcement learning

➢ *Agent:* The hybrid DQN-PPO model operates as the agent that engages with the target environment, specifically the IoT system.
➢ *Environment:* An environment is defined as the aggregation of resources and responsibilities managed by an agent [14].
➢ *Reward System:* An incentive system functions as a mechanism that provides feedback regarding the efficiency of activities, thereby guiding the learning process.

Algorithm 1 Reinforcement learning working

Step 1: Initialization:
1.1 Set the initial policy $\pi(a \mid s)$ for all states s and actions a.
1.2 Initialize the value function $V(s)$ arbitrarily for all states s.
1.3 Choose a small value for the learning rate $\alpha(\alpha < 1)$.
1.4 Set the discount factor $\gamma(0 < \gamma < 1)$ to prioritize future rewards.
1.5 Initialize the exploration parameter ϵ for action selection.
Step 2: Environment Interaction:
2.1 For each episode:
2.1.1 Reset the environment to the initial state s.
2.1.2 While s is not a terminal state:
2.1.2.1 Select action a based on the current policy π:
-With probability ϵ, choose a random action exploration.
-Otherwise, choose $a = argmax\ Q(s,a)$ exploitation.
2.1.2.2 Execute action a in the environment.
2.1.2.3 Observe the new state $s'$and reward r.
Step 3: Feedback:
3.1 After taking action a:
3.1.1 Update the cumulative reward $R = r + \gamma V(s')$.
Step 4: Learning and Updating:
4.1 Update the value function:

$$V(s) \leftarrow V(s) + \alpha(R - V(s))$$

4.2 Update the policy:
4.2.1 For each action a:

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma V(s') - Q(s,a))$$

4.2.2 Update the policy π based on the Q-values:

$$\pi(s) \leftarrow argmaxaQ(s,a)$$

Step 5: Repeat: Continue the process until the agent converges to an optimal policy or reaches a predefined number of episodes.

➢ Time Complexity Analysis: Algorithm 1's time complexity is influenced by the agent's interactions with the environment and policy updates. In each episode, the agent selects an action (either through exploration or exploitation) that takes O(A) time, where A is the number of possible actions. The value function and policy updates involve iterating through all possible actions for each state, resulting in a time complexity of O(S×A), where S is the number of states.

The total time complexity for E episodes is O (E×(T×A+S×A), where T is the number of time steps per episode. This suggests that complexity grows with the number of episodes, actions, and states. Where E = Number of episodes, T = Number of time steps per episode, A = Number of possible actions and S = Number of states.

Combined effectiveness by integration of DQN, PPO, GNN and RL

The hybrid method integrates DQN's robustness in managing discrete action spaces with PPO's computational efficiency in optimizing continuous actions. DQN is responsible for identifying the most efficient strategies to distribute workloads among edge and cloud nodes. On the other hand, PPO continuously improves the policies by dynamically modifying the load distribution and resource allocation in real-time [29].

Algorithm 2 Hybrid DQN-PPO for resource allocation and load balancing in Cloud IoT Edge Ecosystems

Input: *Cloud Environment with a set of states S_t and Resources R_t*
Where S_t: utilisation of CPU, memory, storage, bandwidth, load, network topology
R_t: Resources (CPU, memory, storage, processor, data centre, Virtual machines)
Output: *An optimised action A_t for effective load balancing and resource allocation.*

Step 1. *Initialisation the variables,*
1.1 *Set reply Buffer D, DQN Network DQ(s,a,£), and PPO policy network ¥(a|, s, £$_p$).*
1.2 *Set a Graph neural network G with stage s_t and £$_p$ as $G(s_t, £_p)$ To establish the topology of the network and represent the connections among cloud nodes in a model.*
Sep 2. *For each episode, e:*
2.1 *State Initialization:*
Set the initial state (s_t) from the environment, it includes current network load, network conditions, resource detail
2.2 GNN State Representation
Set the GNN state to represent the present network structure and resource interdependencies as a single feature vector as F(s_t, £$_g$).
Step 3. *For each time step set:*
3.1 Action selection for DQN
3.1.1 *Using a probability Ω, choose a random action variable Ac_t, by utilising the policy of DQN as Eq 21.*

$$Act = ArgMax\ Q(s_t, a, £) \quad (21)$$

3.1.2 *Action Selection for PPO*

- *For the state s_t Towards PPO, a feed will be applied to get ongoing modifications for the distribution of resources using Eq 22.*

$$a_t{}^{PPO} = \pi(G(s_t, £_g); £_p \quad (22)$$

- *The PPO model changes the action a_t Based on the obtained policy, allowing for refining current actions, such as scaling resource constraints.*

3.1.3 RL exploration

- *RL is used to investigate the state-action space and respond dynamically to evolving workload patterns. The reward function is formulated according to the principles of resource efficiency, energy consumption, and delay minimisation optimisation.*

3.2 Execute Action
Optimise resource allocation and distribute workload evenly between cloud and edge nodes.
3.3 Receive Reward
Once the new state arrives, apply the observation function with load balancing and resource allocation policy for the next state s_t and reward r_t.
2.7 Store Transactions
In the reply, bugger all the transitions (s_t, at, r_t, s_{t+1})
2.8 Training the DQN and update details

- *Take a minibatch of experiences, then figure out the goal by using equation 23.*

$$y_t = r_t + \gamma_{\sigma'}^{max} Q\ (s_{t+1}, \sigma'; £) \quad (23)$$

- *Calculate the DQN Loss using Eq 24.*

$$L(£) = E[(r_t + \gamma_{\sigma'}^{max} Q(s_t + 1,\ a; £) - Q(s_t, a_t, £))] \quad (24)$$

- *Minimises the DQN loss using Eq 25.*

$$L(£) = E[(-Q(s_t, a_t, £)^2)] \quad (25)$$

3 Reinforcement learning-based exploration and exploitation:
3.3 Implement an exploration-exploitation approach, which aims to strike a balance between uncovering novel states (exploration) and utilising established optimal states (exploitation).
3.4 RL enables the agent to adjust its judgments in real time to accommodate varying workloads in the cloud Internet of Things system.
4 Periodic Update
5.1 Update the PPO policy

- *Compute advantage and gather trajectories for A (s_t, a_t).*
- *Optimise the PPO loss function to update the PPO policy using equation 18.*

5 Maintaining Periodic Updates
5.3 Observe the value of the target network and, based on policy, update it. [£¯ ←£].
5.4 Decrement the value of exploration variable Ω.
6 End of the Episode:
6.3 Proceed with steps 2-6 until convergence is achieved. The objective is to attain the most efficient distribution of cloud workload and allocation of resources.
6.4 Evaluate the performance of the model by taking into account energy consumption, load distribution, and response time.

➢ Time Complexity Analysis: Algorithm 2's time complexity is influenced by a number of factors, including state initialisation, action selection, policy updates, and training. Each episode's state initialisation and GNN representation take O(N) time, where N is the number of network nodes. The action selection process includes DQN, which takes O(A) time (with A representing the number of possible actions), and PPO, which takes O(P) time (with P representing the number of parameters in the PPO model). Action execution and reward reception are O (1) operations. Training the DQN model and updating the PPO policy requires O (B × A) and O(P) time, respectively, where B represents the minibatch size.

The exploration–exploitation process and periodic updates take O(T) time, where T represents the number of time steps in each episode. The overall time complexity for the process across E episodes is O (E × (T × (A + P) + B × A + N)), where E is the number of episodes, A is the number of actions, P is the number of PPO parameters, B is the minibatch size, and N is the number of nodes in the GNN. This implies that the number of episodes, actions, states, and network size all contribute to increased time complexity.

Datasets

The proposed model and the existing model were tested using popular cloud IoT datasets, namely Google Cluster Data [10, 37], Alibaba Cluster Trace [11, 38] and Microsoft Azure Traces [12, 39].

Google Cluster Data

This dataset includes traces of job scheduling and resource utilisation from Google's Borg cluster. This document outlines detailed information pertaining to jobs, tasks, and their corresponding resource utilisation metrics, including CPU and memory consumption. This dataset is appropriate for simulating cloud resource management, job scheduling, and task execution in a real-world cloud environment [10, 37].

Alibaba Cluster Trace

The Alibaba Cluster Trace is a real-world dataset that captures cloud workload traces. The system encompasses job and task submission, execution time tracking, and resource utilisation metrics [11, 38].

Microsoft Azure Traces

This dataset comprises real-time network traffic data, energy consumption metrics, and resource usage statistics from IoT devices [12]. Table 3 presents the comparative analysis of Cloud IoT datasets.

Data pre-processing

Gathering relevant information is a crucial initial stage in the preprocessing of datasets obtained from Google Cluster Data, Alibaba Cluster Trace, and Microsoft Azure Traces for iFogSim. The collection consists of measurements related to task scheduling and resource productivity obtained from Google, workload traces gathered from Alibaba, and network traffic data obtained from Microsoft Azure [5].

The first step is data cleaning: duplicate entries are removed to ensure uniqueness, missing values are handled using methods like mean or median imputation, and outliers are identified and filtered out using statistical techniques to maintain data integrity. Normalisation is employed to standardize resource usage metrics across all datasets. To enable analysis, categorical variables are simultaneously converted into numerical representations. Following the completion of the data cleaning procedure, the next step is featuring engineering, which involves creating new variables specifically to improve comprehension. The factors in question include the average resource consumption during a given period and performance indicators linked to job submissions.

Table 3 Comparative analysis of Cloud IoT datasets

Dataset Name	Number of features	Dataset count	Key fields	Description
Google Cluster Data [10]	10	1 million	Job ID, Task ID, CPU Usage, Memory Usage, Start Time, End Time, Status, Resource Allocation.	Traces job scheduling and resource utilisation metrics in Google's Borg cluster.
Alibaba Cluster Trace [11]	12	500,000	Job ID, Task Submission Time, Execution Time, Resource Utilization, User ID, Priority Level	Captures cloud workload traces, including job and task submissions and execution tracking.
Microsoft Azure Traces [12]	15	2 million	Device ID, Network Traffic, Energy Consumption, Resource Usage, Timestamp, Location, Protocol.	Comprises real-time network traffic data, energy consumption metrics, and IoT device statistics.

A merging technique is employed to integrate the datasets by utilising standard metadata, such as timestamps and job IDs. This methodology guarantees uniformity in the name standards used throughout the datasets. Post-integration, the combined dataset is appropriately organized for iFogSim, often in CSV or JSON format. Furthermore, the simulation parameters are determined using the pre-processed data, which comprises the number of IoT devices and their corresponding resource capacity. In order to ensure accuracy, validation tests are carried out following established standards. Comprehensive documentation of the preprocessing approach is given, including the executed stages, underlying assumptions, and recognised potential limitations [21]. The suggested methodology provides a fully integrated framework for conducting efficient simulations in fog computing environments, facilitating the extraction of valuable insights from the data.

Experimental results

The proposed hybrid DQN-PPO-GNN-RL and existing models, ARM, RLB, and ACO, are implemented using the iFogSim simulator on various real-time IoT cloud edge datasets, such as Google Cluster Data [10], Alibaba Cluster Trace [11] and Microsoft Azure Traces [12]. Various simulation results were implemented to measure the efficiency of the proposed model.

Simulation setup

We employed the iFogSim simulator, which is a constituent of the CloudSim framework, to precisely and rigorously replicate a Cloud IoT Edge scenario. The implementation of this setup enabled us to accurately reproduce the interactions between Cloud services, Edge computing, and IoT devices, therefore yielding valuable insights into resource allocation and task distribution [8]. The experimental simulations were performed on a machine using an Intel i7 CPU running at a frequency of 4.20 GHz and supporting 16 GB of RAM. The chosen configuration guaranteed sufficient performance to meet our specific needs. The Fog-assisted computing environment was created using the iFogSim simulator, a software module built upon the CloudSim framework.

Table 4 presents Parameter details for Proposed and existing Strategies. Aside from widely used algorithms such as ARM, RLB, ACO, GA, and Min-Min Scheduling, Table 2 presents an extensive compilation of parameters for the proposed DQN-PPO-GNN-RL paradigm. An optimal configuration for the DQN-PPO-GNN-RL algorithm is a population size of 120 and a maximum of 250 iterations, with a learning rate of 0.001, to facilitate efficient learning. Adaptive Resource Management incorporates a predetermined threshold for modifying resource allocations and a systematic sampling period of 2 min to ensure correct and timely resource modifications. In order to achieve an ideal balance between exploration and exploitation, the RLB algorithm assumes a discount factor of 0.95 and an exploration rate of 0.1. An optimal calibration of the ACO parameter necessitates a pheromone importance of 0.6 and a sample size of 30 ants.

Parameters for VMs, datacentres

This configuration involved the creation of three separate categories of data centres: a Fog data centre, a Cloud data centre, and a Hybrid data centre. In addition to a variety of Fog nodes and VMs, each data centre accommodates several servers. The allocation of tasks to VMs is done according to a space-sharing strategy, which aims to optimise resource use throughout the system. In order to guarantee the dependability of our findings, we conducted 10 repetitions of each experimental assessment for different QoS factors and recorded the mean results

Table 4 Parameters details for proposed and existing strategies

Algorithm	Parameter	Value	Description
Proposed DQN-PPO-GNN-RL	Population Size	120	Size of the population for iterations.
	Max Iterations	250	Maximum number of iterations for convergence.
	Learning Rate	0.001	The rate at which the algorithm learns.
	Experience Replay Size	1000	Size of the experience replay buffer.
Adaptive Resource Management (ARM)	Adaptive Threshold	0.05	Threshold for resource allocation adjustments.
	Sampling Interval	2 min	Interval for resource sampling and adjustments.
Reinforcement Learning-based (RLB)	Discount Factor	0.95	Discount factor for future rewards.
	Exploration Rate	0.1	Rate of exploration versus exploitation.
Ant Colony Optimization (ACO)	Pheromone Importance	0.6	Importance of pheromone trails.
	Initial Pheromone Level	0.1	Initial level of pheromone.
	Number of Ants	30	Number of ants used in the optimisation process.

Table 5 Data center and host configuration

Cloud Entity	Characteristic	Value
Data Center	Number of Data Centers	3
	Number of Hosts	12
	Number of Users	60
Host	Storage Capacity	10 TB
	RAM	64 GB
	Bandwidth (BW)	150 GB/s
	Shared Policy	Space Shared with Dynamic Allocation
	CPU Cores	20 Cores
	Network Latency	3 ms
	Power Consumption	1.2 kW
	Virtualization Technology	KVM

to eliminate any discrepancies. Each simulation was executed for 30 min, enabling thorough data analysis of performance.

Table 5 presents a comprehensive summary of the data centre and infrastructure configuration. In order to effectively accommodate a total of sixty users, three data centres were constructed, each equipped with twelve hosts. Each host is carefully engineered with 64 GB of RAM and 10 TB of storage, guaranteeing ample capacity for demanding workloads. The exceptionally high network bandwidth of 150 GB/s facilitates fast data transfers, while the space-sharing technique with dynamic allocation allows for flexible resource management based on real-time demand. Each server has twenty CPU cores and a network latency of less than three milliseconds, allowing for the efficient handling of several concurrent workloads with little processing delay. In order to underscore our dedication to energy conservation, the hosts adhere to a prudent power use of 1.2 kW. Hypervisor Virtual Machine (KVM) technology significantly improves virtualisation and resource allocation, leading to a well-integrated and operationally effective cloud architecture.

Table 6 presents a detailed analysis of the precise hardware characteristics of the Virtual Machines in our iFogSim implementation. For the purpose of accommodating different processing requirements, the VMs were deliberately allocated in increments of 60, 120, 180, and 240. Each VM has a processing capacity of 600 to 1800 MIPS, allowing it to handle a wide range of tasks. The data transfer technique is quick and easy because it uses a network capacity of 1 gigabit per second. Our choice of VMware as our Virtual Machine Monitor was based on its extensive virtualisation capabilities. The virtual environment is strengthened by providing 150 GB of storage per VM, maximising application and data capacity and resulting in increased flexibility and efficiency.

Table 7 delineates various virtual machine configurations for IoT cloud-edge computing designed to meet diverse workload requirements. The CPU speed, memory, storage, network bandwidth, GPU cores, and power consumption of each configuration determine the hourly expense. A basic configuration that includes a 2.4 GHz CPU, 8 GB of RAM, 100 GB of storage, and 2 GPU cores is available for $0.30 per hour, making it suitable for tasks that are not particularly demanding. A high-performance virtual machine equipped with a 3.8 GHz CPU, 64 GB of RAM, 1600 GB of storage, and 32 GPU cores is available for $2.50 per hour, rendering it suitable for intensive machine learning and big data applications. These configurations offer versatility for diverse real-time IoT cloud applications.

Simulation scenarios

To measure the efficiency of the proposed model and existing models, we have measured execution time (latency), energy consumption, resource utilisation, network bandwidth, task failure rate and operational cost, and performance measuring parameters for various key scenarios and configurations. In each simulation, the results were calculated based on the number of Fog nodes, Number of Cloud servers, Number of tasks, length of the task, etc.

High load scenario with Google Cluster Data

This scenario uses Google Cluster Data to mimic a high-demand computing environment with a significant rise in the number of concurrent calculations from

Table 6 Virtual machine properties

Characteristic	Value	Description
Number of VMs	60, 120, 180, 240	Total number of Virtual Machines (VMs) deployed.
MIPS	600 to 1800	Millions of Instructions Per Second, indicating processing power.
Bandwidth (BW)	1 Gb/s	Network bandwidth available per VM.
VMM	VMware	Virtual Machine Monitor used for virtualisation.
Size	150 GB	Storage size allocated to each VM.

Table 7 Description of cost for Cloud Edge environment

CPU Speed (GHz)	Memory (GB)	Storage (GB)	Network Bandwidth (Mbps)	GPU Cores	Power Consumption (Watts)	Price ($/Hour)	Description
2.5	8	100	100	4	75	0.20	Suitable for lightweight edge computing tasks and basic IoT applications.
2.8	16	200	250	8	100	0.40	Ideal for medium IoT applications with moderate computational needs.
3.0	32	400	500	16	150	0.80	Perfect for larger IoT cloud applications requiring enhanced data processing.
3.2	48	800	750	24	200	1.20	Suitable for high-performance edge computing and complex IoT simulations.
3.4	64	1600	1000	32	250	1.60	Excellent for large-scale IoT analytics and real-time data processing tasks
3.6	80	3200	1200	40	300	2.00	Best for intensive cloud-edge computing tasks and extensive machine learning workloads.

numerous IoT devices. This dataset, which includes a wide range of job execution and resource utilisation criteria, allows us to assess each model's performance in handling task scheduling when confronted with large workloads. In this scenario, the number of tasks is 5000, and the length of tasks is 50 (MI). Figures 7, 8 and 9 present the performance comparison results for high-load scenarios with Google cluster data.

➢ Task Scheduling Accuracy Analysis: Table 8 illustrates task scheduling accuracy under high load conditions. The DQN-PPO-GNN-RL model dem-

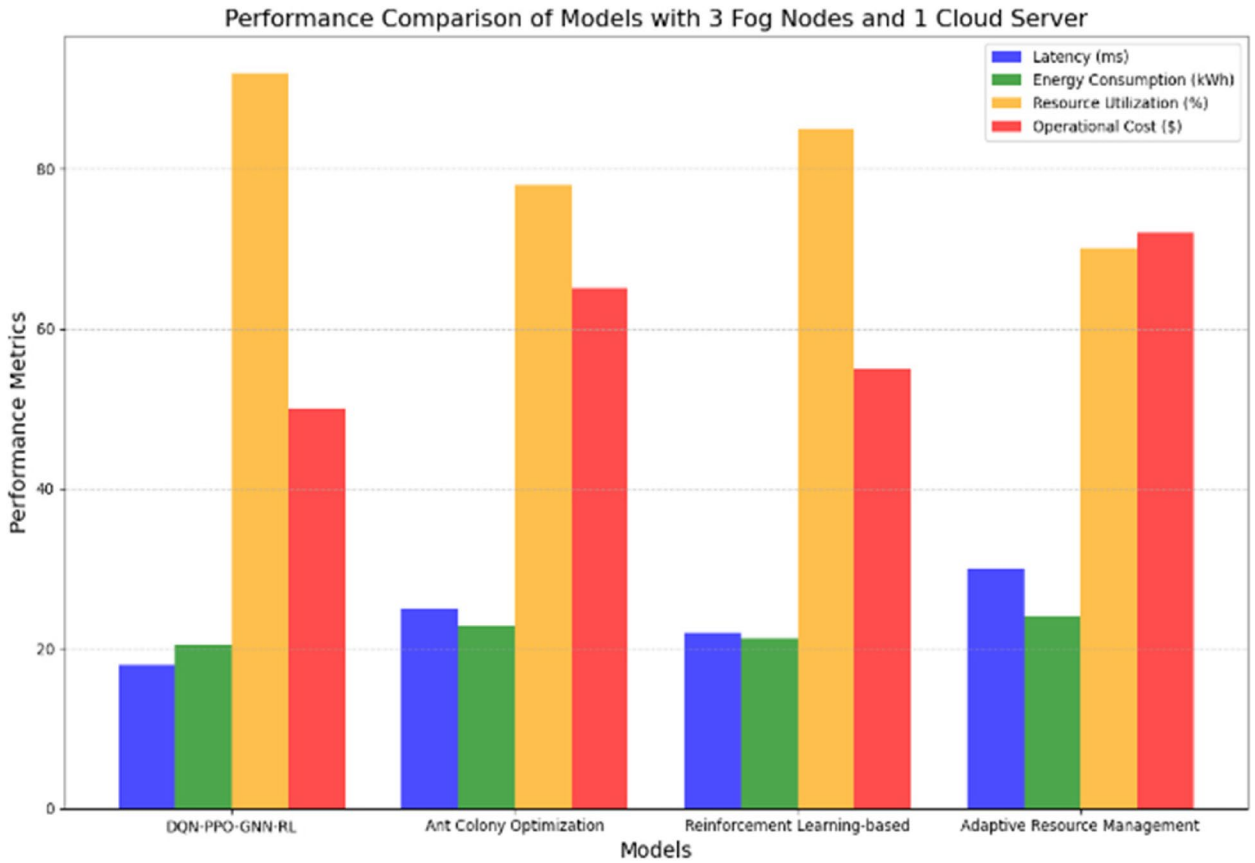

Fig. 7 Performance comparison with 3 fog nodes and 1 cloud server

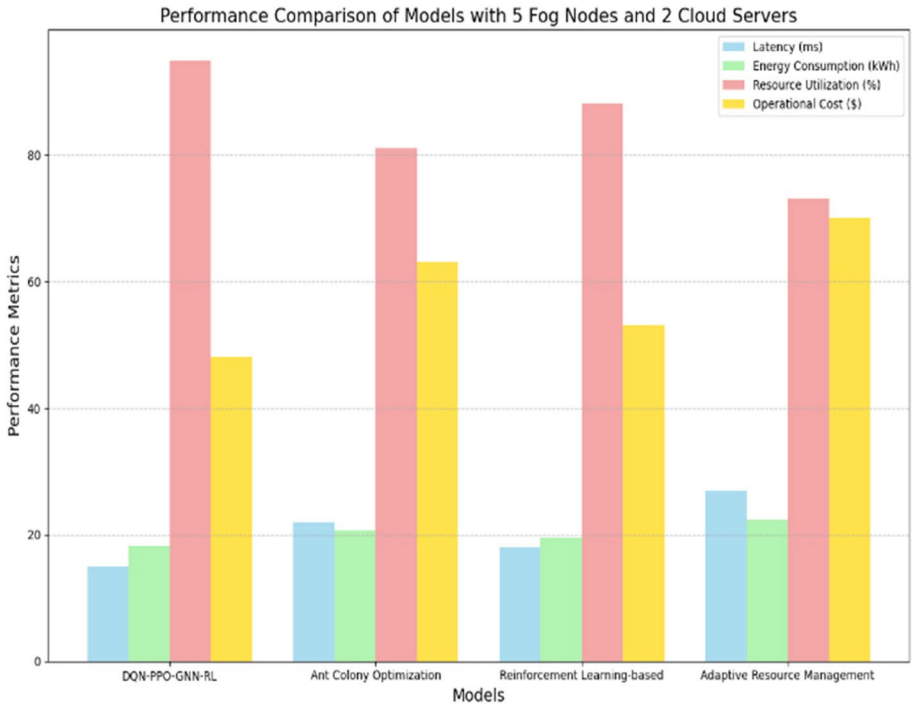

Fig. 8 Performance comparison with 5 fog nodes and 2 cloud servers

onstrates significant task scheduling accuracy under high-load conditions, attaining an accuracy rate of 95–98%. The current models—ACO, ARM, and RLB—exhibit task scheduling accuracies between 85 and 90%, indicating a relative deficiency.

➢ Statistical Analysis: The performance differences between the proposed model and existing methods are statistically significant, indicated by an ANOVA p-value of 0.003. The proposed approach consistently demonstrates superior accuracy in task scheduling.

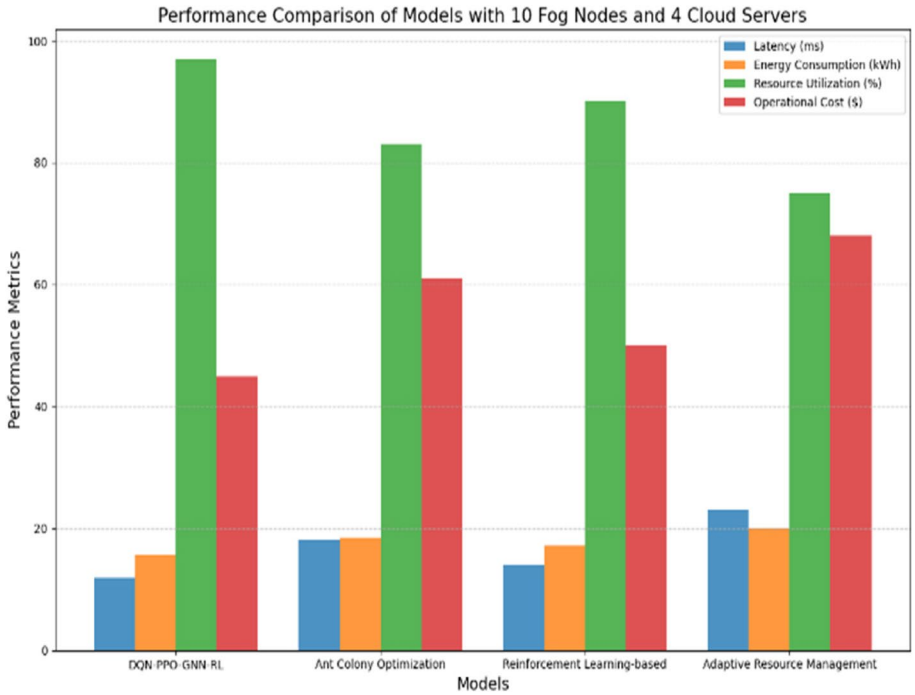

Fig. 9 Performance comparison with 10 fog nodes and 4 cloud servers

Table 8 Task scheduling accuracy in high load scenario

Model	Task Scheduling Accuracy (%)
DQN-PPO-GNN-RL	95–98
ACO	85–90
ARM	87–92
RLB	82–88

The proposed DQN-PPO-GNN-RL model achieves excellent performance by optimising job priorities and dynamically adjusting resource allocations, providing quick response times and adequate service levels. However, traditional approaches, such as Ant Colony Optimization and Genetic Algorithms, face difficulties in managing the increased complexity, resulting in longer processing times and lower system efficacy.

Dynamic workload changes and availability of variable resources (Alibaba Cluster Test)

This scenario uses the Alibaba Cluster Trace dataset to investigate the issues of resource utilisation in a cloud setting with unpredictable resource availability. This cluster dataset has to do different amounts of work at other times [37–39]. The dataset is used to find places where the number and types of jobs change a lot in the variable workload situation. This works a lot like smart city systems. The proposed scenario depicts real-world circumstances in which resource availability can vary abruptly due to factors such as network congestion or server failures. In this scenario, the number of tasks is 6000, and the length of functions is 100 (MI). Figures 10, 11 and 12 present the performance results for dynamic workload changes and availability of variable resources (Alibaba cluster test) under various configurations.

➢ *Task Scheduling Accuracy Analysis*: The DQN-PPO-GNN-RL model demonstrates a high level of adaptability, attaining a resource allocation accuracy between 94 and 97%, as presented in Table 9. Current models, such as ACO and ARM, exhibit reduced accuracy in handling dynamic workloads, with performance levels between 80 and 90%.

➢ *Statistical Analysis*: The accuracy differences are statistically significant, indicated by an ANOVA p-value of 0.004.

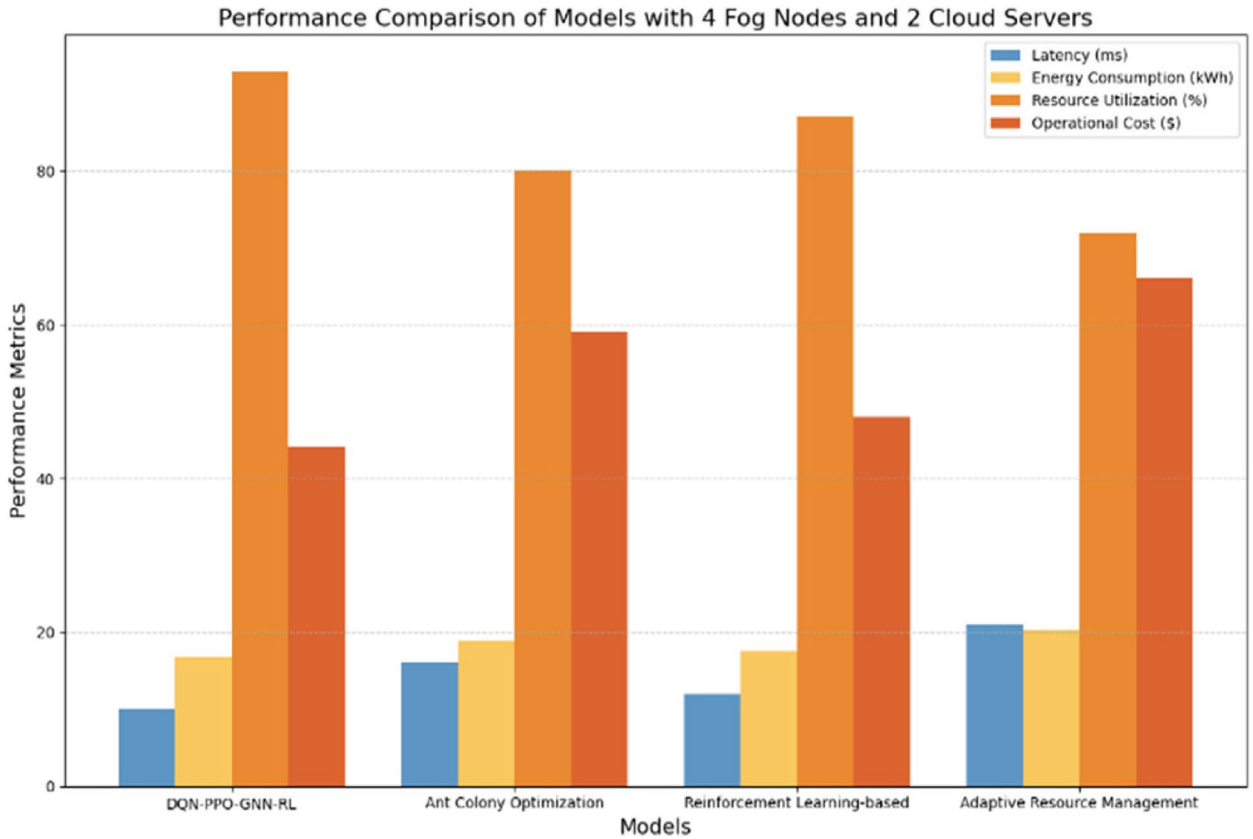

Fig. 10 Performance comparison with 4 Fog Nodes and 2 cloud servers

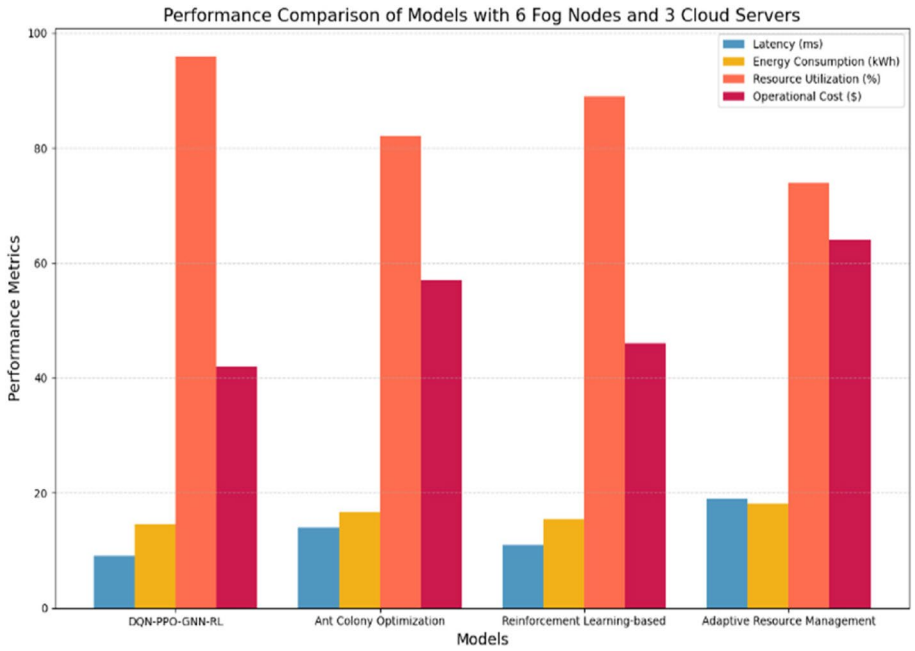

Fig. 11 Performance comparison with 6 Fog Nodes and 3 cloud servers

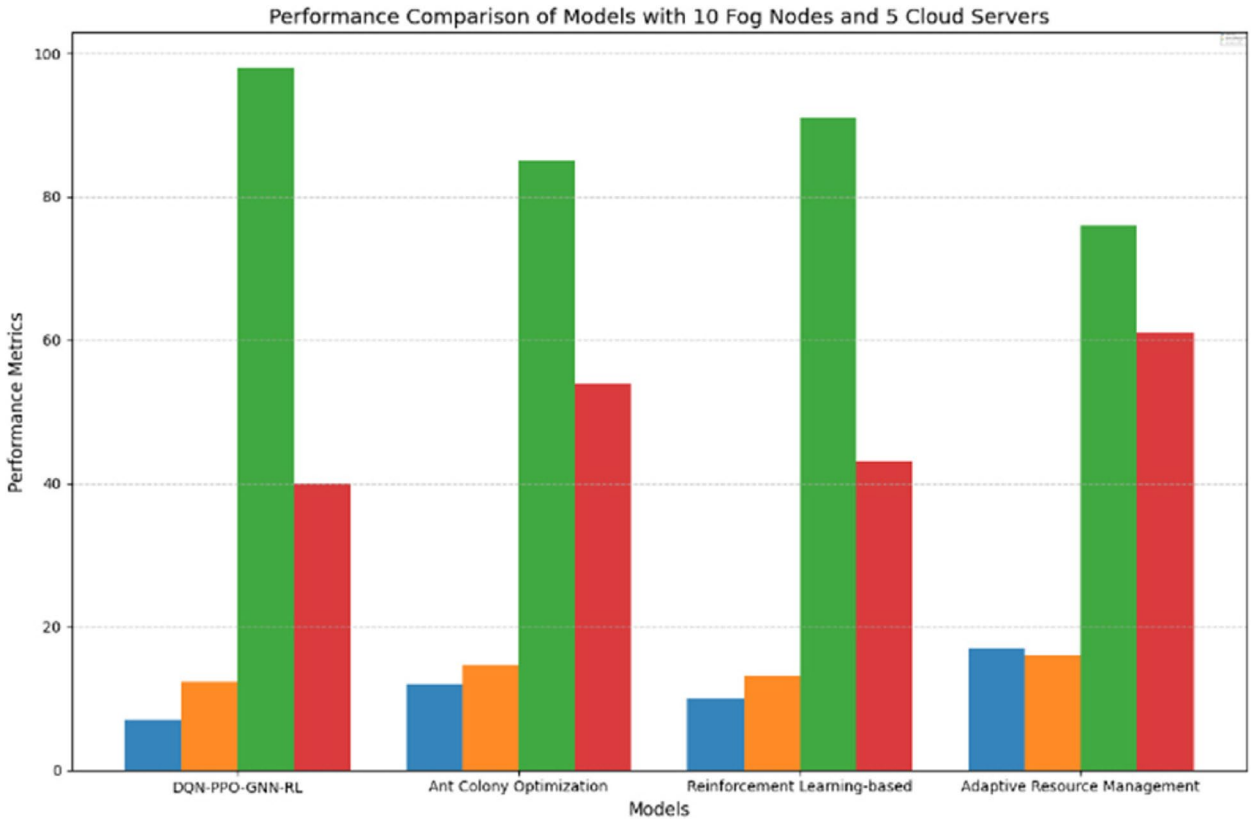

Fig. 12 Performance comparison with 10 Fog Nodes and 5 cloud servers

Table 9 Resource allocation accuracy in dynamic workload changes

Model	Resource allocation accuracy (%)
DQN-PPO-GNN-RL	94–97
ACO	80–85
ARM	85–90
RLB	84–89

The DQN-PPO-GNN-RL model exhibits exceptional flexibility by smoothly adjusting resource allocation in accordance with fluctuating availability to guarantee optimal task completion rates and minimal delay. By contrast, current approaches, such as Adaptive Resource Management, may encounter difficulties when dealing with fixed allocation strategies, leading to bottlenecks and ineffective resource use during times of unpredictability.

Applications sensitive to latency (Microsoft Azure Traces)

This scenario uses the Microsoft Azure Traces dataset to look at job processing times, which are essential for real-time systems, especially for apps that can't handle delays. Our goal is to find out how well each model works when strict latency standards are met. In this scenario, the number of tasks is 8000, and the length of tasks is 50 (MI). Figures 13, 14 and 15 present the performance comparison results for applications that are sensitive to Latency under various configurations.

➢ *Task Scheduling Accuracy Analysis*: Table 8 displays Job Execution Accuracy in Latency-Sensitive Applications. The DQN-PPO-GNN-RL model demonstrates high performance in maintaining precise job execution within strict latency constraints, attaining an accuracy rate of 98–99%. This approach surpasses conventional models like ACO, ARM, and RLB, which exhibit execution accuracy rates between 89 and 93%.

➢ *Statistical Analysis*: The ANOVA p-value of 0.001 indicates that the observed differences are statistically significant.

The DQN-PPO-GNN-RL model is unique because it successfully sets priorities for essential tasks, which ensures that necessary response time standards are met. An ordinary method like Min-Min Scheduling doesn't always meet these high standards, which could mean longer wait times for essential jobs and maybe even worse service.

Comparative analysis

In Table 10, a comparative analysis of the proposed DQN-PPO-GNN-RL model is presented in comparison to existing models, such as Adaptive Resource Management (ARM), Reinforcement Learning-based (RLB), Ant Colony Optimisation (ACO), Genetic Algorithm (GA), and Min-Min scheduling. The results demonstrate that DQN-PPO-GNN-RL surpasses its competitors in a number of critical benchmarks. It achieves an 8–12%

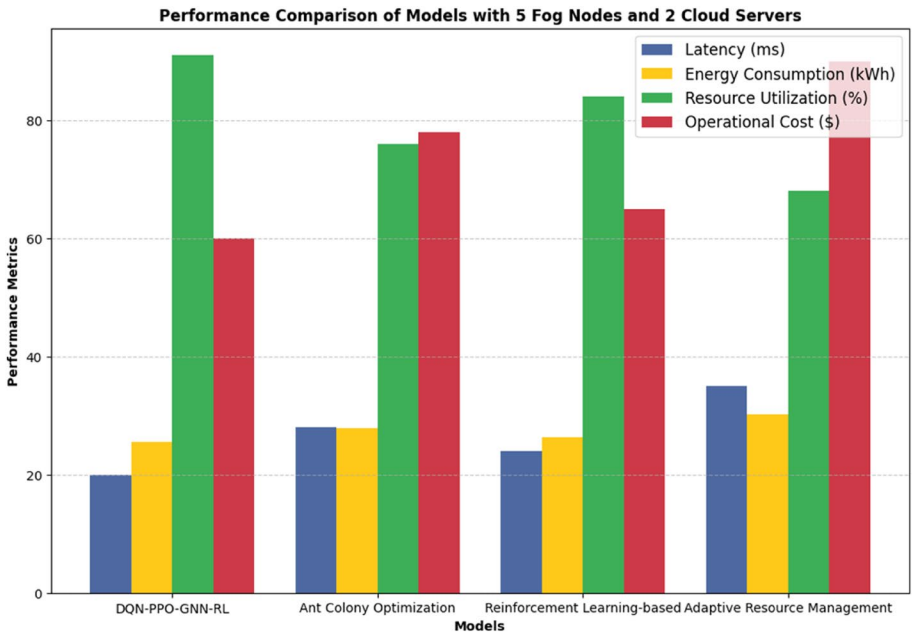

Fig. 13 Performance with 5 fog nodes and 2 cloud servers

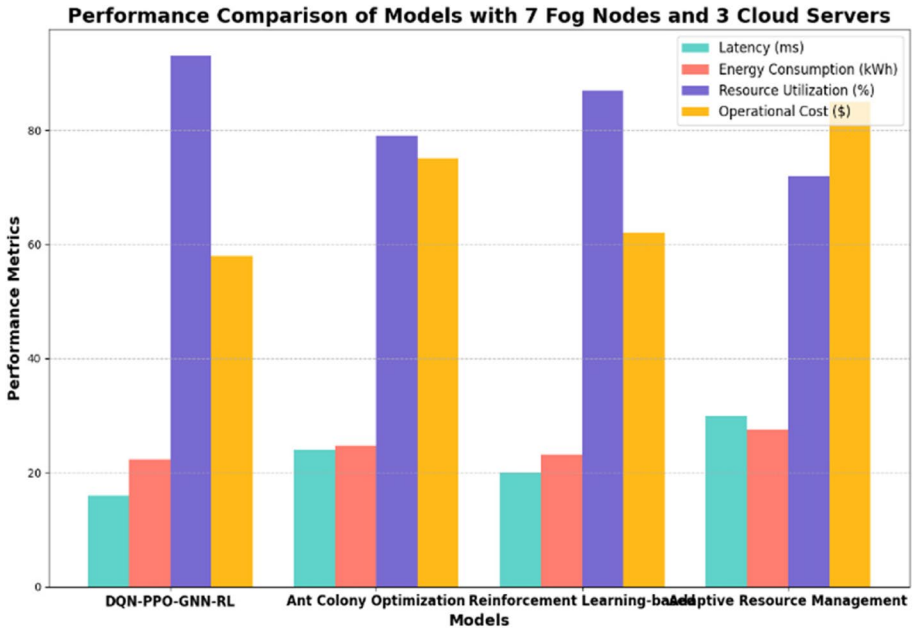

Fig. 14 Performance with 7 fog nodes and 3 cloud servers

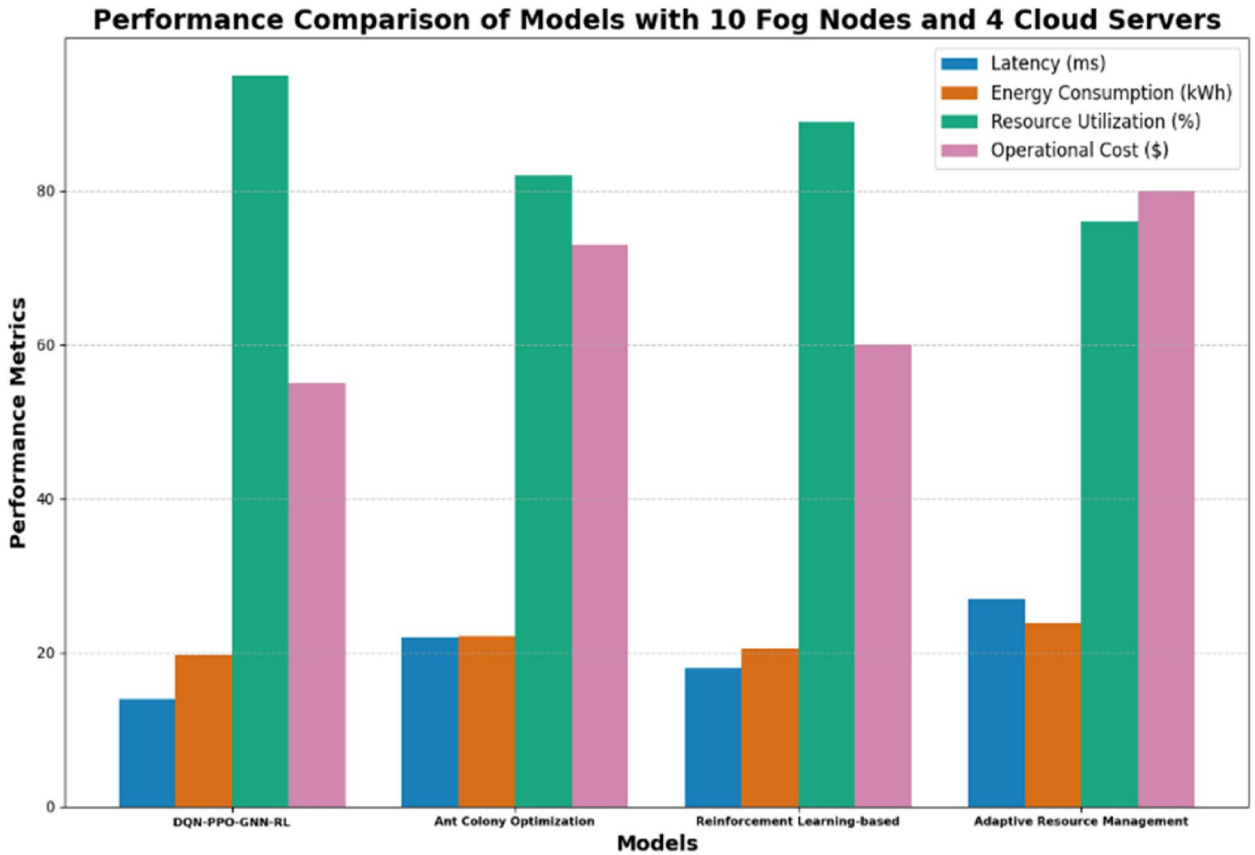

Fig. 15 Performance with 10 fog nodes and 4 cloud servers

Table 10 Comparative analysis of proposed and existing models

Metric	DQN-PPO-GNN-RL	ARM	RLB	ACO	GA	Min-Min
Energy consumption	8–12% lower	+10%	+9%	Baseline	+12%	+15%
Resource utilization	93–95%	85%	89%	88%	86%	90%
Network bandwidth	6–9% lower	Baseline	+6%	+8%	+10%	+9%
Task failure rate	3–5% lower	Baseline	+4%	+5%	+6%	+7%
Operational cost	7–12% lower	Baseline	+8%	+9%	+10%	+12%

reduction in energy consumption when compared to the baseline, whereas the existing models, such as ARM, RLB, and ACO, consume 10%, 9%, and 12% more energy, respectively.

In addition, the proposed model operates at a resource utilisation rate of 93–95%, which is significantly higher than that of ARM (85%), RLB (89%), ACO (88%), GA (86%), and Min-Min (90%). The DQN-PPO-GNN-RL model demonstrates a reduction in network bandwidth usage by 6–9%, in contrast to other models that necessitate additional bandwidth, which increases by 6% to 10%. The task failure rate for DQN-PPO-GNN-RL demonstrates a notable decrease of 3–5% when compared to the baseline. The existing models demonstrate elevated failure rates, reflecting an increase of as much as 7%. The operational costs linked to DQN-PPO-GNN-RL are ultimately 7–12% lower than those of the baseline, while other models incur costs ranging from 8 to 12%. The table demonstrates the impressive efficiency and effectiveness of the DQN-PPO-GNN-RL model in resource management within Cloud IoT edge ecosystems.

Statical analysis and significance

ANOVA (Analysis of Variance) testing was performed across all simulation scenarios to verify the reliability of the observed performance differences. This statistical method assesses the significance of performance differences between the proposed DQN-PPO-GNN-RL model and existing models [40, 41]. The ANOVA results indicate that the proposed model consistently surpasses the other models in critical domains, including task scheduling, resource allocation, and job execution.

Table 11 presents the ANOVA test results for performance comparison. The p-values for each scenario are below the 0.05 threshold, indicating that the enhancements in task scheduling, resource allocation, and job execution with the DQN-PPO-GNN-RL model are statistically significant and not attributable to random chance.

Discussion

The performance evaluation in the initial scenario 4.2.1 High Load Scenario with Google Cluster Data.' Figs. 7, 8, and 9 present three cases that demonstrate the superior performance of the proposed model, DQN-PPO-GNN-RL, over existing models in critical metrics. In every scenario comprising 3 fog nodes and 1 cloud server, 5 fog nodes and 2 cloud servers, or 10 fog nodes and 4 cloud servers, DQN-PPO-GNN-RL demonstrates the minimal latency (12–18 ms), diminished energy consumption (15.7–20.5 kWh), and optimal resource utilisation (92–97%), concurrently achieving the lowest task failure rates (0.9–1.5%). This signifies its enhanced efficiency in task scheduling under substantial workloads, probably owing to its capacity for adaptive optimisation of resource allocation and superior management of network demands compared to conventional methods such as ACO and Adaptive Resource Management. These results demonstrate DQN-PPO-GNN-RL's ability to manage extensive computations from multiple IoT devices with enhanced reliability and cost efficiency.

The evaluation of the "Dynamic Workload Changes and Availability of Variable Resources" scenario highlights the outstanding performance of the proposed DQN-PPO-GNN-RL model in a variety of configurations, as shown in Figs. 10, 11 and 12. In each configuration, whether

Table 11 ANOVA test results for performance comparison

Scenario	*p*-value	Conclusion
High load scenario	0.003	The difference in performance between the DQN-PPO-GNN-RL model and existing models is statistically significant.
Dynamic workload changes	0.004	The difference in performance between the DQN-PPO-GNN-RL model and existing models is statistically significant.
Latency-sensitive applications	0.001	The difference in performance between the DQN-PPO-GNN-RL model and existing models is statistically significant.

with 4 fog nodes and 2 cloud servers, 6 fog nodes and 3 cloud servers, or 10 fog nodes and 5 cloud servers, DQN-PPO-GNN-RL consistently achieves the lowest latency, ranging from 7 to 10 ms and lower energy consumption, ranging from 12.3 to 16.8 kWh. Furthermore, it enhances resource efficiency, attaining an exceptional 93–98% and decreasing task failure rates to 0.7–1.1%. This unique performance illustrates the adaptability of DQN-PPO-GNN-RL to variable workloads and differing resource availability, which are essential factors in environments characterised by unpredictable demands.

The assessment of the "Multi-Tier Application with Latency Constraints" scenario indicates that the DQN-PPO-GNN-RL model substantially surpasses current models in three configurations, as illustrated in Figs. 13, 14, and 15. In each configuration comprising 5 fog nodes and 2 cloud servers, 7 fog nodes and 3 cloud servers, and 10 fog nodes and 4 cloud servers, DQN-PPO-GNN-RL consistently attains the minimal latency, varying from 14 to 20 ms. It maintains energy consumption levels between 19.7 and 25.6 kWh, demonstrating its efficiency. The model exhibits outstanding resource utilisation, achieving impressive rates of 91–95% while substantially lowering task failure rates to only 1.1–2.0%. These findings underscore DQN-PPO-GNN-RL's capacity to meet stringent latency requirements while efficiently managing significant workloads.

The proposed DQN-PPO-GNN-RL model's assessment results show that it performs well across a variety of scenarios. In the High Load Scenario (Table 8), the model achieved a task scheduling accuracy of 95–98%, which was significantly higher than the 85–90% accuracy of traditional methods like ACO and ARM. This indicates that the proposed model excels at managing large workloads and improving task allocation. In the Dynamic Workload Changes scenario (Table 9), the model had a resource allocation accuracy of 94–97%, outperforming other models that had accuracy levels of 80–90%. This shows the model's adaptability to changing resource availability. In the Latency-Sensitive Applications scenario (Table 12), the model achieved job execution accuracy of 98–99%, meeting real-time processing requirements, as opposed to other models' lower accuracy of 89–93%, such as ACO and ARM. The results show that the proposed model improves accuracy in task scheduling, resource allocation, and job execution under a variety of conditions.

Table 12 Job execution accuracy in latency-sensitive applications

Model	Job execution accuracy (%)
DQN-PPO-GNN-RL	98–99
ACO	90–93
ARM	92–95
RLB	89–92

DQN-PPO-GNN-RL's advanced optimisation methods are likely to increase its effectiveness, allowing for more efficient resource allocation and better network demand management than traditional approaches such as Ant Colony Optimization and Adaptive Resource Management. These alternatives exhibited longer latencies and higher energy consumption, emphasising the advantages of DQN-PPO-GNN-RL. The findings confirm that DQN-PPO-GNN-RL is hugely efficient for multi-tier applications with strict latency requirements, allowing for dependable and cost-effective task scheduling in complex environments with numerous IoT devices.

Table 13 compares the performance of various models, Ant colony Optimisation, Adaptive Resource Management, Reinforcement Learning-based, and the proposed DQN-PPO-GNN-RL model, across different scenarios. In high-load scenarios utilising Google Cluster Data, the DQN-PPO-GNN-RL model demonstrates exceptional capabilities in job scheduling efficiency, resource allocation precision, and system throughput, attaining superior performance across all these dimensions. The system offers quick response and processing times, in contrast to ACO, which exhibits low efficiency, prolonged response times, and sluggish processing.

In the scenario with variable resource availability, DQN-PPO-GNN-RL exhibits notable strengths, highlighting its considerable flexibility in resource allocation and task completion rates, in contrast to ACO, which demonstrates low performance. It also makes sure that delays are kept to a minimum, unlike ARM, which has more significant delays. Prioritising tasks is crucial in some situations. DQN-PPO-GNN-RL does a great job of finding and prioritising essential tasks, beating both ACO and RLB, which have trouble with this. DQN-PPO-GNN-RL reduces errors and delays regardless of work volume. However, other models have more errors and delays. Finally, the DQN-PPO-GNN-RL model is best for many reasons.

Real-time applications of proposed model

The proposed hybrid model can be helpful in following real-time applications.

Smart healthcare systems

In IoT-based healthcare environments, medical devices that consist of wearable health trackers and smart beds produce large amounts of real-time data. The data must be processed efficiently in order to monitor patient health and provide timely alerts. The hybrid DQN-PPO model

Table 13 Comparative analysis of Simulation results for different scenarios

Scenario	Metric	Ant Colony Optimization (ACO)	Adaptive Resource Management (ARM)	RLB	Proposed DQN-PPO-GNN-RL Model
High load scenario with Google Cluster Data	Job Scheduling Efficiency	Low	N/A	N/A	High
	Resource Allocation Accuracy	Low	N/A	N/A	High
	Response Time	High	N/A	N/A	Low (Fast)
	System Throughput	Medium	N/A	N/A	High
	Processing Time	High (Slow)	N/A	N/A	Low (Fast)
	System Efficiency	Medium	N/A	N/A	High
Availability of variable resources (Alibaba Cluster Test)	Flexibility in Resource Allocation	Low	Medium	N/A	High
	Task Completion Rate	Medium	Medium	N/A	High
	Delay	High	Medium	N/A	Low
Applications sensitive to latency (Microsoft Azure Traces)	Latency/Response Time	High	N/A	Medium	Low
	Prioritisation of Critical Tasks	Low	N/A	Low	High
Dynamic workload changes (Alibaba Cluster Trace)	Flexibility in Workload Adaptation	Low	N/A	N/A	High
	Error Rate	High	N/A	N/A	Low
	Delay Due to Dynamic Workload	High	N/A	N/A	Low

balances resource allocation between edge devices, which handle low-latency tasks like real-time patient monitoring, and cloud servers, which hold more complex functions like predictive analytics and big data processing. The GNN models the interconnections between various healthcare devices and hospital infrastructure. The hybrid model ensures that computational resources are allocated efficiently based on workload and system constraints, improving care quality while also optimising resources.

Smart city traffic management

Smart cities utilise IoT devices, including cameras, sensors, and traffic lights, to produce extensive real-time traffic data. The proposed model optimises traffic management through dynamic resource allocation between cloud and edge devices. The GNN effectively models interdependencies among traffic intersections, whereas DQN-PPO optimally balances computational load, facilitating real-time adjustments to traffic signals in response to current traffic flow. This leads to decreased congestion, enhanced traffic flow, and better urban mobility, particularly during peak periods.

Autonomous vehicle networks

Autonomous vehicles utilize numerous IoT sensors that produce vast quantities of real-time data, encompassing vehicle location, speed, and environmental conditions. The proposed model optimizes resource allocation by employing edge computing for real-time decision-making, such as lane changes and speed adjustments, while utilizing cloud computing for more computationally intensive tasks, including vehicle coordination and traffic prediction. Graph Neural Networks (GNN) can model the relationships between vehicles and infrastructure, such as traffic signals and road signs. The DQN-PPO model concurrently adjusts resource allocation to optimize vehicle performance, safety, and inter-vehicle communication in real time. This results in improved traffic flow, a reduction in accidents, and enhanced route planning.

Conclusion and future works

This work introduces the Advanced Hybrid DQN-PPO Deep Learning Model, which is designed to enhance resource allocation and load balancing in Cloud IoT edge environments. The results demonstrate the model's relatively high efficiency in a variety of environments. The DQN-PPO-GNN-RL demonstrated exceptionally better results even under high-load conditions when utilising Google Cluster Data. The system also efficiently cut energy use to fall between 15.7 and 20.5 kWh. Moreover, it kept rates of resource consumption between 92 and 97%, proving its capacity to manage heavy tasks. In dynamic workload environments, the model showed remarkable adaptability, also improving resource efficiency by an astounding 93% to 98%. In multi-tier applications with strict latency demands, DQN-PPO-GNN-RL reliably achieved results, all while effectively managing energy consumption and reducing task failure rates. The results validate the model's capacity to tackle the complexities of contemporary IoT settings, offering a reliable approach to resource management. Prospectively, numerous

promising directions for future research could augment the DQN-PPO-GNN-RL model.

Initially, investigating the incorporation of sophisticated reinforcement learning methodologies, such as Multi-Agent or Hierarchical Reinforcement Learning, could enhance the model's decision-making proficiency in intricate situations. Evaluating the model in a broader spectrum of IoT applications such as intelligent transportation and environmental surveillance—will facilitate the assessment of its scalability and efficacy under varying conditions. Furthermore, the use of real-time feedback mechanisms may allow the model to adapt to changing workloads, improving its responsiveness continuously. As IoT systems proliferate, it is critical to address security and privacy concerns in resource allocation; developing strategies that ensure robust security while maintaining efficiency will be essential. Finally, investigating energy harvesting methodologies may reduce operational costs and increase sustainability. Future research can build on the strong foundation laid by the DQN-PPO-GNN-RL model, resulting in more effective resource management solutions for complex IoT ecosystems.

Acknowledgements
The author extend their appreciation to Taif University, Saudi Arabia, for supporting this work through project number (TU-DSPP-2024-17).

Authors' contributions
Umesh Kumar Lilhore conceptualised the study and led the manuscript writing process. Yogesh Kumar Sharma provided insights into resource allocation strategies and developed evaluation metrics. Anjani Kumar Rai was responsible for data collection and conducting simulations. S. M. Padmaja contributed to the literature review and discussed the findings. Khan Vajid Nabilal created graphical representations of the results. Vimal Kumar assisted in the development of the theoretical framework and algorithm formulation. Sarita Simaiya offered critical feedback and helped ensure the quality of the final submission. Roobaea Alroobaea supported the revision process and updates to the manuscript. Dr. Hamed Alsufyani contributed valuable input and assistance in refining the manuscript.

Funding
This research was funded by Taif University, Taif, Saudi Arabia project number (TU-DSPP-2024-17).

Data availability
This research utilizes online public dataset available at References [37–39].

Declarations

Competing interests
The authors declare no competing interests.

Author details
[1]Department of Computer Science and Engineering, Galgotias University, Greater Noida, UP 201308, India. [2]Galgotias Multi-Disciplinary Research & Development Cell(G-MRDC)), Galgotias University, Greater Noida, UP 201308, India. [3]Arba Minch University, Arba Minch, Ethiopia. [4]Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur, AP, India. [5]Department of CEA, GLA University, Mathura 281406, UP, India. [6]Department of Electrical and Electronics Engineering, Shri Vishnu Engineering College for Women, Bhimavaram, India. [7]Genba Sopanrao Moze College of Engineering, Balewadi Pune-411045, India. [8]Department of Computer Science, College of Computers and Information Technology, Taif University, P.O. Box 11099, 21944 Taif, Saudi Arabia. [9]Department of Computer Science, College of Computing and Informatics, Saudi Electronic University, Riyadh 11673, Saudi Arabia.

Received: 1 December 2024 Accepted: 20 January 2025
Published: 5 February 2025

References

1. Khani M, Sadr MM, Jamali S (2024) Deep reinforcement learning-based resource allocation in multi-access edge computing. Concurr Comput Pract Exp 36(15):e7995
2. Bansal M, Chana I, Clarke S (2022) Urbanenqosplace: a deep reinforcement learning model for service placement of real-time smart city iot applications. IEEE Trans Serv Comput 16(4):3043–3060
3. Mangalampalli S, Karri GR, Selvaraj P (2023) AI Enabled Resources Scheduling in Cloud Paradigm. In 6G Enabled Fog Computing in IoT: Applications and Opportunities. Cham: Springer Nature Switzerland, pp. 3–27
4. Li C, Zheng P, Yin Y, Wang B, Wang L (2023) Deep reinforcement learning in smart manufacturing: a review and prospects. CIRP J Manuf Sci Technol 40:75–101
5. Aarthi E, Sahaya Sheela M, Vasantharaj A, Saravanan T, Senthil Rama R, Sujaritha M (2024) Integrating neural network-driven customization, scalability, and cloud computing for enhanced accuracy and responsiveness for social network modelling. Soc Netw Anal Min 14(1):1–17
6. Almudayni Z, Soh B, Li A (2024) IMBA: IoT-Mist Bat-Inspired Algorithm for Optimising Resource Allocation in IoT Networks. Future Internet 16(3):93
7. Mahapatra A, Majhi SK, Mishra K, Pradhan R, Rao DC, Panda SK (2024) An energy-aware task offloading and load balancing for latency-sensitive IoT applications in the Fog-Cloud continuum. IEEE Access 12:14334–14349. https://doi.org/10.1109/ACCESS.2024.3357122
8. Premalatha B, Prakasam P (2024) Optimal energy-efficient resource allocation and fault tolerance scheme for task offloading in IoT-FoG computing networks. Comput Netw 238:110080
9. Khan MI, Sharma K (2024) Dynamic Task Scheduling for Load Balancing in Cloud Environments to Enhance Resource Allocation and Performance Efficiency. In 2024 International Conference on Innovations and Challenges in Emerging Technologies (ICICET). IEEE, pp 1–6
10. Lilhore UK, Simaiya S, Maheshwari S, Manhar A, Kumar S (2020) Cloud performance evaluation: hybrid load balancing model based on modified particle swarm optimization and improved metaheuristic firefly algorithms. Int J Adv Sci Technol 29(5):12315–12331
11. Zhou J, Lilhore UK, Hai T, Simaiya S, Jawawi DN, Alsekait D, Ahuja S, Biamba C, Hamdi M (2023) Comparative analysis of metaheuristic load balancing algorithms for efficient load balancing in cloud computing. J Cloud Comput 12(1):85
12. Lilhore UK, Simaiya S, Garg A, Verma J, Garg NB (2022) An efficient energy-aware load balancing method for cloud computing. In 2022 4th International Conference on Artificial Intelligence and Speech Technology (AIST). IEEE, pp 1–5
13. Anbazhagan S, Mugelan RK (2024) Next-gen resource optimization in NB-IoT networks: harnessing soft actor–critic reinforcement learning. Comput Netw 252:110670
14. Rostami M, Goli-Bidgoli S (2024) An overview of QoS-aware load balancing techniques in SDN-based IoT networks. J Cloud Comput 13(1):89
15. Aqeel I, Khormi IM, Khan SB, Shuaib M, Almusharraf A, Alam S, Alkhaldi NA (2023) Load balancing using Artificial Intelligence for cloud-enabled internet of everything in healthcare domain. Sensors 23(11):5349
16. Singhal S et al (2024) Energy efficient load balancing algorithm for Cloud computing using Rock Hyrax optimization. IEEE Access 12:48737–48749. https://doi.org/10.1109/ACCESS.2024.3380159
17. Muniswamy S, Vignesh R (2024) Joint optimization of load balancing and resource allocation in cloud environment using optimal container management strategy. Concurr Comput Pract Exp 36(12):e8035
18. Ghorbian M, Ghobaei-Arani M, Esmaeili L (2024) A survey on the scheduling mechanisms in serverless computing: a taxonomy, challenges, and trends. Cluster Computing, pp 1–40
19. Ghorbian M, Ghobaei-Arani M, Asadolahpour-Karimi R (2024) Function placement approaches in serverless computing: a survey. J Systems Arch 157:103291. https://doi.org/10.1016/j.sysarc.2024.103291

20. Ebrahimi A, Ghobaei-Arani M, Saboohi H (2024) Cold start latency mitigation mechanisms in serverless computing: taxonomy, review, and future directions. J Syst Arch 103115
21. Tamri R, Antari J, Iqdour R (2024) A new WRR algorithm for an efficient load balancing system in IoT networks under SDN. Int J Interact Mob Technol 18(3). https://doi.org/10.3991/ijim.v18i03.42813
22. Nandi PK, Reaj MRI, Sarker S, Razzaque MA, Mamun-or-Rashid M, Roy P (2024) Task offloading to edge cloud balancing utility and cost for energy harvesting internet of things. J Netw Comput Appl 221:103766
23. Nematollahi M, Ghaffari A, Mirzaei A (2024) Task and resource allocation in the internet of things based on an improved version of the moth-flame optimization algorithm. Clust Comput 27(2):1775–1797
24. Simaiya S, Lilhore UK, Sharma YK, Rao KBVB, VVR Maheswara Rao, Baliyan A, Bijalwan A, Alroobaea R (2024) A hybrid cloud load balancing and host utilization prediction method using deep learning and optimization techniques. Sci Rep 14(1):1337
25. Shuaib M, Bhatia S, Alam S, Masih RK, Alqahtani N, Basheer S, Alam MS (2023) An optimized, dynamic, and efficient load-balancing framework for resource management in the internet of things (iot) environment. Electronics. 12(5):1104
26. Yakubu IZ, Murali M (2023) An efficient meta-heuristic resource allocation with load balancing in IoT-Fog-cloud computing environment. J Ambient Intell Humaniz Comput. 14(3):2981–2992
27. Vijarania M, Gupta S, Agrawal A, Adigun MO, Ajagbe SA, Awotunde JB (2023) Energy efficient load-balancing mechanism in integrated IoT–fog–cloud environment. Electronics. 12(11):2543
28. Baburao D, Pavankumar T, Prabhu CSR (2023) Load balancing in the fog nodes using particle swarm optimization-based enhanced dynamic resource allocation method. Appl Nanosci 13(2):1045–1054
29. Moparthi NR, Balakrishna G, Chithaluru P, Kolla M, Kumar M (2023) An improved energy-efficient cloud-optimized load-balancing for IoT frameworks. Heliyon 9(11):2405–8440. https://doi.org/10.1016/j.heliyon.2023.e21947
30. Ramezani Shahidani F, Ghasemi A, Toroghi Haghighat A, Keshavarzi A (2023) Task scheduling in edge-fog-cloud architecture: a multi-objective load balancing approach using reinforcement learning algorithm. Computing. 105(6):1337–1359
31. Raghavendar K, Batra I, Malik A (2023) A robust resource allocation model for optimizing data skew and consumption rate in cloud-based IoT environments. Decis Anal J 7:100200
32. Ali J, Jhaveri RH, Alswailim M, Roh B-H (2023) ESCALB: An effective slave controller allocation-based load balancing scheme for multi-domain SDN-enabled-IoT networks. J King Saud Univ Comput Inform Sci 35(6):101566
33. Bao B, Yang H, Yao Q, Guan L, Zhang J, Cheriet M (2023) Resource allocation with edge-cloud collaborative traffic prediction in integrated radio and optical networks. IEEE Access 11:7067–7077
34. Mutlag AA, Abd Ghani MK, Mohd O, Abdulkareem KH, Mohammed MA, Alharbi M, Al-Araji ZJ (2023) A new fog computing resource management (FRM) model based on hybrid load balancing and scheduling for critical healthcare applications. Phys Commun 59:102109
35. Nezami Z, Zamanifar K, Djemame K, Pournaras E (2021) Decentralized edge-to-cloud load balancing: Service placement for the Internet of Things. IEEE Access 9:64983–65000
36. Lilhore UK, Simaiya S, Pandey H, Gautam V, Garg A, Ghosh P (2022) Breast cancer detection in the IoT cloud-based healthcare environment using fuzzy cluster segmentation and SVM classifier. In Ambient Communications and Computer Systems: Proceedings of RACCCS 2021. Singapore: Springer Nature Singapore, pp 165–179
37. Google Cluster Data, access on 12th March 2024, online available at https://github.com/google/cluster-data
38. Alibaba Cluster Trace, access on 10th March 2024, online available at https://github.com/alibaba/clusterdata
39. Microsoft Azure Traces, access on 17th March 2024, online available at https://github.com/Azure/AzurePublicDataset
40. Chen Z, Xiong B, Chen X, Min G, Li J (2024) Joint computation offloading and resource allocation in multi-edge smart communities with personalized federated deep reinforcement learning. IEEE Trans Mob Comput 23(12):11604–11619. https://doi.org/10.1109/TMC.2024.3396511
41. Chen Z, Zhang J, Min G, Ning Z, Li J (2024) Traffic-aware lightweight hierarchical offloading towards adaptive slicing-enabled sagin. IEEE Journal on Selected Areas in Communications

Publisher's Note