SvedbergOpen
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

# A Forecast-Driven Closed-Loop Framework for Proactive Cloud Load Balancing and Resource Optimization Using Multi-Agent Reinforcement Learning

Mr. Kirit C. Patel[1*], Dr. Bhavesh Patel[2], Dr. Ajay M. Patel[3], Mr. Ravi S. Patel[4], Prof. Deepika J. Patel[5], Prof. Sonal J. Patel[6], Prof. Nikita Modi[7], Prof. Parth T. Lakhatariya[8]

[1*]Research Scholar, DCS, Ganpat University. E-mail: kcp01@ganpatuniversity.ac.in
[2]Associate Professor, FCA, Ganpat University. E-mail: bhavesh.patel@ganpatuniversity.ac.in
[3]Associate Professor, FCA, Ganpat University. E-mail: ajaykumar.patel@ganpatuniversity.ac.in
[4]Assistant Professor, DCS, Ganpat University. E-mail: rsp01@ganpatuniversity.ac.in
[5]Assistant Professor, DCS, Ganpat University. E-mail: djp01@ganpatuniversity.ac.in
[6]Assistant Professor, FCA, Ganpat University. E-mail: sjp02@ganpatuniversity.ac.in
[7]Assistant Professor, DCS, Ganpat University. E-mail: npm01@ganpatuniversity.ac.in
[8]Assistant Professor, DCS, Ganpat University. E-mail: ptl01@ganpatuniversity.ac.in

**ABSTRACT**

Cloud data centers must efficiently manage dynamic and heterogeneous workloads while satisfying Quality of Service (QoS) requirements, minimizing energy consumption, and maximizing resource utilization. Traditional load balancing techniques primarily rely on reactive scheduling strategies that fail to anticipate workload fluctuations, resulting in resource imbalance, excessive virtual machine (VM) migrations, increased Service Level Agreement (SLA) violations, and higher operational costs. [8-10]Recent advances in machine learning and reinforcement learning have improved individual aspects of cloud resource management[14-16]; however, forecasting, scheduling, and resource optimization are generally treated as independent processes, limiting overall system adaptability.[18-20]

This paper proposes an Intelligent Closed-Loop Multi-Objective Load Balancing Framework that integrates Spatiotemporal Adaptive Ensemble Forecasting (SAEF), a Hybrid Adaptive Scheduling Engine (HASE), and Graph-Augmented Multi-Agent Deep Reinforcement Learning (GA-MADRL) within a unified optimization architecture. The SAEF module predicts future workload behavior by combining temporal attention mechanisms, graph-based spatial representations, adaptive ensemble learning, and uncertainty estimation[1-7]. The predicted workload is utilized by HASE to perform proactive multi-objective task scheduling using tenant-aware prioritization, metaheuristic optimization, and Pareto-based decision-making[9-13]. Subsequently, GA-MADRL models the cloud infrastructure as a heterogeneous graph and learns adaptive VM placement and migration policies through cooperative reinforcement learning agents[15-18]. A continuous feedback mechanism connects all three modules, enabling self-adaptive optimization as cloud conditions evolve.

The proposed framework is designed to improve resource utilization, reduce scheduling overhead, minimize energy consumption, and enhance QoS in heterogeneous cloud environments. Its modular architecture supports integration with modern cloud platforms and provides a scalable foundation for intelligent autonomous resource management.

**Keywords:** Cloud Computing, Load Balancing, Workload Forecasting, Resource Scheduling, Graph Neural Networks, Deep Reinforcement Learning, Multi-Agent Systems, Energy-Efficient Computing.

## Introduction

Cloud computing has become the backbone of modern digital infrastructure by providing scalable, on-demand computing resources through virtualization and distributed resource sharing. Enterprises, research organizations, and public service providers increasingly rely on cloud platforms to host data-intensive and latency-sensitive applications, requiring cloud providers to deliver high availability, elasticity, and efficient resource utilization. As cloud infrastructures continue to expand, resource management has evolved into a complex optimization problem due to dynamic workload patterns, heterogeneous computing resources, and stringent Quality of Service (QoS) requirements.

Load balancing is one of the most critical functions in cloud resource management because it directly influences system throughput, response time, energy consumption, and Service Level Agreement (SLA) compliance. Conventional scheduling algorithms, including Round Robin, First-Come First-Served (FCFS), Min-Min, and Max-Min, distribute workloads based primarily on the current state of computing resources. Although computationally efficient, these approaches are reactive and often fail to adapt to rapidly changing workload conditions. Consequently, cloud providers may experience server overload, underutilized resources, unnecessary VM migrations, and increased operational costs.

Recent advances in Artificial Intelligence (AI) have introduced new possibilities for intelligent cloud scheduling.

Deep learning models such as Long Short-Term Memory (LSTM), Gated Recurrent Units (GRU), and Transformer architectures have demonstrated strong performance in workload prediction[1-7], while Reinforcement Learning (RL) has enabled adaptive resource allocation through continuous interaction with cloud environments[14-16]. Similarly, Graph Neural Networks (GNNs) have improved the representation of complex cloud infrastructures by modeling relationships among physical servers, virtual machines, and computational tasks[15-18]. Despite these advances, existing approaches typically address forecasting, scheduling, and resource optimization as independent problems, limiting their ability to achieve global optimization.

This research proposes an Intelligent Closed-Loop Multi-Objective Load Balancing Framework that combines three complementary modules within a unified architecture. The Spatiotemporal Adaptive Ensemble Forecasting (SAEF) module predicts future workload behavior using temporal and spatial learning techniques. The Hybrid Adaptive Scheduling Engine (HASE) performs forecast-aware multi-objective scheduling by considering workload priority, resource availability, and QoS constraints. The Graph-Augmented Multi-Agent Deep Reinforcement Learning (GA-MADRL) module continuously optimizes VM placement and migration through cooperative learning agents operating on graph-based infrastructure representations. Unlike conventional cloud management systems, the proposed framework establishes a continuous feedback mechanism in which execution outcomes are utilized to refine forecasting models, scheduling strategies, and reinforcement learning policies.

The primary contributions of this work are summarized as follows:

- A novel closed-loop architecture integrating workload forecasting, adaptive scheduling, and graph-based reinforcement learning.
- A Spatiotemporal Adaptive Ensemble Forecasting (SAEF) model for predictive workload estimation with uncertainty awareness.
- A Hybrid Adaptive Scheduling Engine (HASE) for proactive multi-objective resource allocation.
- A Graph-Augmented Multi-Agent Deep Reinforcement Learning (GA-MADRL) framework for intelligent VM placement and migration.
- A unified optimization strategy designed to improve resource utilization, reduce energy consumption, and enhance QoS in heterogeneous cloud environments.

The remainder of this paper is organized as follows. Section 2 reviews related work in intelligent cloud scheduling. Section 3 presents the proposed closed-loop framework. Sections 4–6 describe the SAEF, HASE, and GA-MADRL modules, respectively. Section 7 outlines the experimental methodology and evaluation strategy. Finally, Section 8 concludes the paper and discusses future research directions.

## 2. Related Work

Cloud load balancing has been extensively studied using heuristic, metaheuristic, machine learning, and reinforcement learning techniques. Traditional scheduling algorithms, including Round Robin (RR), First-Come First-Served (FCFS), Min-Min, and Max-Min, provide low computational complexity and are easy to implement. However, these approaches allocate resources based on the current system state and therefore struggle to adapt to dynamic workload variations in heterogeneous cloud environments.[1-3]

To overcome these limitations, researchers have proposed metaheuristic optimization techniques such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Whale Optimization Algorithm (WOA), Grey Wolf Optimization (GWO), and NSGA-II[11-13]. These algorithms improve scheduling quality by exploring large search spaces and optimizing multiple objectives simultaneously. Nevertheless, their convergence speed and solution quality often depend on parameter tuning, making them less suitable for real-time cloud scheduling.

Recent advances in Artificial Intelligence have significantly improved cloud resource management. Deep learning models, including Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Transformer architectures, have demonstrated promising performance in workload prediction by learning temporal dependencies from historical resource utilization data[4-7]. Similarly, Graph Neural Networks (GNNs) have been introduced to model complex relationships among physical servers, virtual machines, and computational tasks, providing richer infrastructure representations for scheduling decisions.[15-18]

Reinforcement Learning (RL) has further enhanced adaptive resource allocation by enabling agents to learn scheduling policies through continuous interaction with cloud environments. Algorithms such as Deep Q-Network (DQN), Deep Deterministic Policy Gradient (DDPG), and Proximal Policy Optimization (PPO) have shown superior adaptability compared with conventional schedulers. More recently, Multi-Agent Reinforcement Learning (MARL) has been investigated for distributed cloud resource management, allowing multiple agents to cooperatively optimize large-scale infrastructures.[16-20]

Although these approaches have advanced cloud scheduling, several limitations remain. Most forecasting models operate independently of scheduling algorithms, while reinforcement learning methods typically rely only on current infrastructure states. Existing studies also optimize a limited set of objectives, such as response time or energy consumption, without jointly considering workload prediction, SLA compliance, migration

overhead, and resource utilization[9-13]. Furthermore, very few frameworks establish continuous feedback among forecasting, scheduling, and reinforcement learning modules, preventing adaptive optimization over successive scheduling cycles.

The proposed framework addresses these limitations by integrating Spatiotemporal Adaptive Ensemble Forecasting (SAEF), the Hybrid Adaptive Scheduling Engine (HASE), and Graph-Augmented Multi-Agent Deep Reinforcement Learning (GA-MADRL) into a unified closed-loop architecture that continuously learns from cloud execution outcomes.

**Table 1. Summary of Existing Cloud Load Balancing Approaches**

| Method | Strength | Limitation |
|---|---|---|
| Round Robin | Simple implementation | Static scheduling |
| Min-Min / Max-Min | Low makespan | Resource imbalance |
| GA / PSO | Global optimization | High execution time |
| ACO | Efficient search | Slow convergence |
| LSTM / GRU | Accurate temporal prediction | Ignores spatial dependency |
| Transformer | Captures long-term patterns | High computational cost |
| RL (DQN, PPO) | Adaptive scheduling | Reactive decision-making |
| MARL | Distributed optimization | No predictive workload awareness |
| Proposed Framework | Forecast-aware, adaptive, closed-loop optimization | Higher training complexity |

## 3. Proposed Closed-Loop Framework

### 3.1 Framework Overview

The proposed framework integrates workload forecasting, adaptive scheduling, and reinforcement learning into a unified cloud resource management architecture. As illustrated in Figure 1, the framework operates through a continuous optimization cycle consisting of three intelligent modules: Spatiotemporal Adaptive Ensemble Forecasting (SAEF), Hybrid Adaptive Scheduling Engine (HASE), and Graph-Augmented Multi-Agent Deep Reinforcement Learning (GA-MADRL).

Initially, historical workload traces and real-time infrastructure metrics are collected from the cloud monitoring system. These data include CPU utilization, memory usage, network traffic, storage utilization, task arrival rate, and VM status. The collected information is processed by the SAEF module to predict future workload characteristics and estimate prediction uncertainty[3-6].

The predicted workload is subsequently provided to HASE, which performs forecast-aware multi-objective scheduling. Unlike conventional schedulers that rely only on current infrastructure conditions, HASE proactively allocates tasks by considering expected future resource demand. The scheduler simultaneously optimizes makespan, response time, resource utilization, SLA compliance, and energy consumption through hybrid optimization strategies[9-13].

The optimized scheduling decisions are then forwarded to the GA-MADRL module. Cloud resources are represented as a heterogeneous graph where nodes correspond to physical servers, virtual machines, and computational tasks, while edges describe communication and dependency relationships. Multiple reinforcement learning agents cooperate to determine VM placement, migration, and consolidation strategies that maximize long-term system performance[15-18].

Finally, execution outcomes are continuously monitored and fed back into the forecasting and scheduling modules. This feedback mechanism enables adaptive learning and continuous optimization under evolving cloud workloads, transforming the proposed architecture into a self-improving intelligent cloud management system.

### 3.2 System Architecture

The proposed framework consists of five logical layers:

- **Monitoring Layer:** Collects workload traces and infrastructure metrics.
- **Forecasting Layer (SAEF):** Predicts future resource demand using spatiotemporal learning.
- **Scheduling Layer (HASE):** Performs forecast-aware multi-objective task allocation.

• **Optimization Layer (GA-MADRL):** Optimizes VM placement and resource consolidation.
• **Feedback Layer:** Continuously updates prediction models and scheduling policies using execution outcomes.
This layered architecture improves modularity, scalability, and interoperability, allowing each component to be independently enhanced while maintaining seamless integration within the overall optimization pipeline.

### 3.3 Closed-Loop Optimization Process

The proposed optimization cycle follows six sequential stages:
• Collect workload and infrastructure data.
• Forecast future resource demand using SAEF.
• Perform proactive task scheduling through HASE.
• Execute VM placement and migration using GA-MADRL.
• Monitor QoS metrics, energy consumption, and resource utilization.
• Feed execution outcomes back into the forecasting and learning modules for continuous adaptation.
Unlike conventional cloud management systems, the proposed framework continuously refines its forecasting models, scheduling strategies, and reinforcement learning policies, enabling proactive resource management and improved long-term operational efficiency.

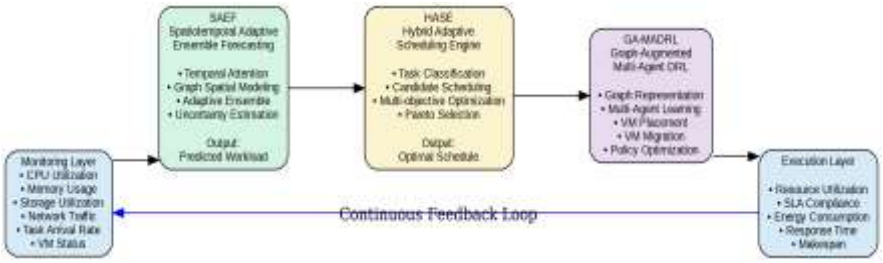


**Figure 1. Overall architecture of the proposed Intelligent Closed-Loop Cloud Resource Management Framework. (To be included in the final manuscript.)**

## 4. Spatiotemporal Adaptive Ensemble Forecasting (SAEF):

### 4.1 Overview

Accurate workload forecasting is essential for proactive cloud resource management. Traditional forecasting techniques primarily model temporal dependencies and often ignore structural relationships among cloud resources, reducing prediction accuracy under highly dynamic workloads. To address this limitation, the proposed Spatiotemporal Adaptive Ensemble Forecasting (SAEF) module combines temporal attention mechanisms, graph-based spatial learning, and adaptive ensemble prediction within a unified framework.
Historical workload traces collected from cloud monitoring systems are first preprocessed to remove noise, normalize resource utilization values, and construct time-series sequences. These sequences include CPU utilization, memory consumption, storage utilization, network traffic, and task arrival rates[3-6]. The processed data are then used to learn both temporal evolution and spatial interactions among cloud resources. The forecasting module generates multi-step workload predictions together with uncertainty estimates[3-5]. These predictions are forwarded to the scheduling module to enable proactive task allocation before workload fluctuations occur.

### 4.2 Forecasting Workflow

The SAEF framework performs workload prediction through four sequential stages:
• Historical workload acquisition and preprocessing.
• Temporal feature extraction using attention-based sequence learning.
• Spatial dependency modeling through graph representations of cloud resources.
• Adaptive ensemble prediction with uncertainty estimation.
The predicted workload vector is represented as:
$\hat{W}(t+1)=f\_SAEF(W(t-k:t), A, \Theta)$
Where $W(t-k:t)$=historical workload sequence, $A$=graph adjacency matrix, $\Theta$=learnable parameters.

### 4.3 Adaptive Ensemble Prediction

Instead of relying on a single forecasting model, SAEF combines multiple predictors using adaptive weights. The ensemble prediction is computed as

$$\hat{Y} = \Sigma(i=1\rightarrow n)\ \alpha\_i\ F\_i(X)$$

Subject to: $\Sigma(i=1\rightarrow n)\ \alpha\_i = 1$
$F\_i(X)$: i-th forecasting model, $\alpha\_i$: adaptive weight

$$\alpha_i = \exp(\beta_i) / \Sigma \exp(\beta_j)$$

The adaptive weighting strategy improves robustness against workload variability and increases prediction reliability[3-5].

**4.4 Output**

The forecasting module produces:

- Predicted CPU utilization
- Predicted memory utilization
- Predicted task arrival rate
- Prediction confidence score
- These outputs guide the scheduling engine during resource allocation.

**5. Hybrid Adaptive Scheduling Engine (HASE):**

**5.1 Overview**

Following workload prediction, the Hybrid Adaptive Scheduling Engine (HASE) performs forecast-aware resource allocation. Unlike traditional schedulers that optimize a single performance metric, HASE simultaneously considers multiple objectives, including resource utilization, energy consumption, response time, makespan, and SLA compliance.[11-13]

The scheduling engine integrates heuristic decision-making with multi-objective optimization to produce balanced task assignments under heterogeneous cloud environments.

**5.2 Multi-Objective Optimization**

The scheduling problem is formulated as

$$\min J = w1M + w2E + w3S + w4C - w5U$$

M: Makespan
E: Energy Consumption
S: SLA Violation
C: Migration Cost
U :Resource Utilization

The weights w1–w5 were empirically selected based on preliminary experiments to balance the conflicting objectives of makespan, energy consumption, SLA violations, migration cost, and resource utilization.

$CPU_i \leq CPU_i(max)$
$RAM_i \leq RAM_i(max)$
$\Sigma x_{ij} = 1$
$x_{ij} \in \{0,1\}$

Where, xij=1 if task i is assigned to VM j, otherwise 000.

These constraints ensure that allocated tasks satisfy available computational resources and Quality of Service requirements.

**5.3 Scheduling Procedure**

The scheduling process consists of the following stages:

- Receive forecasted workload from SAEF.
- Classify incoming tasks according to priority.
- Generate candidate schedules.
- Perform multi-objective optimization.
- Select the Pareto-optimal scheduling solution.
- Allocate tasks to virtual machines.

This proactive strategy enables workload balancing before infrastructure overload occurs.

**5.4 Scheduling Objectives**

The scheduling engine aims to

- Minimize response time
- Reduce makespan
- Improve CPU utilization
- Reduce VM migration
- Lower energy consumption
- Maintain SLA compliance
- The optimized schedule is then forwarded to the reinforcement learning module for dynamic infrastructure optimization.

## 6. Graph-Augmented Multi-Agent Deep Reinforcement Learning (GA-MADRL):

### 6.1 Overview

The final optimization stage is performed using Graph-Augmented Multi-Agent Deep Reinforcement Learning (GA-MADRL). Cloud infrastructure is represented as a heterogeneous graph in which physical servers, virtual machines, and computational tasks are modeled as interconnected nodes. This representation enables the learning algorithm to capture structural dependencies that are not available in conventional state vectors.

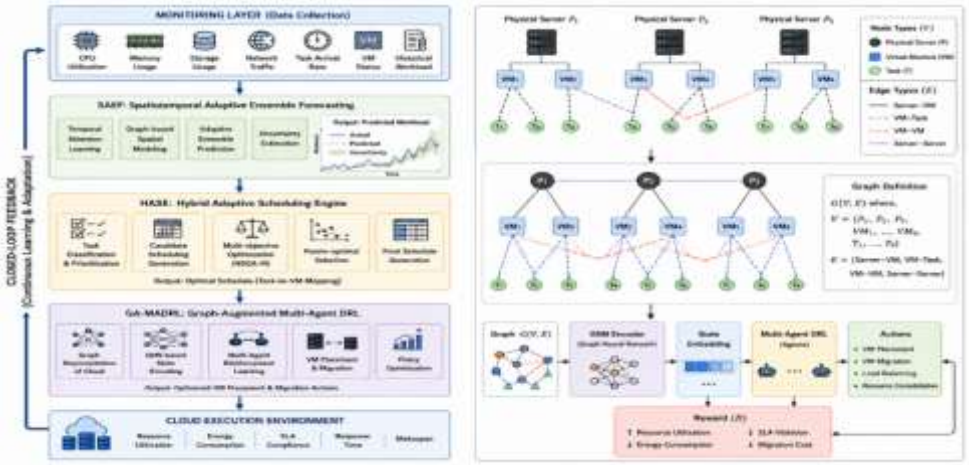

**Figure 2. Graph representation of heterogeneous cloud infrastructure and the working process of the GA-MADRL module.**

Multiple reinforcement learning agents cooperatively optimize VM placement, migration, and resource consolidation while continuously interacting with the cloud environment.

### 6.2 Graph Representation

The cloud graph is defined as [15-18]

G = (V, E)

V: Vertices (Hosts, VMs, Tasks)

E: Edges

Adjacency Matrix: A ∈ R^(|V|×|V|)

Node Feature Matrix: X=[x1,x2,...,xn]

Graph embeddings generated from GGG are used as state representations for reinforcement learning agents.

### 6.3 Reinforcement Learning Formulation

S_t={CPU,RAM,Bandwidth,Queue Length,Energy,VM Status}

The cloud optimization problem is modeled as a Markov Decision Process (MDP)[14-16]

M = (S, A, P, R, γ)

S: State

A: Action

P: Transition Probability

R: Reward

γ: Discount Factor

A_t={Allocate, Migrate, Consolidate, Idle}

The reward function is defined as

$R=w1U'-w2E'-w3S'-w4M'$

U: Resource Utilization

E: Energy

S: SLA Violation

M: Migration Overhead

The prime symbol denotes normalized values scaled to the interval [0,1][0,1][0,1] using min-max normalization.

The coefficients w1,w2,w3,w4 control the relative importance of each optimization objective.

### 6.4 Learning Process

Each reinforcement learning agent performs the following operations:

- Observe the current cloud state.
- Receive graph embeddings.
- Select an optimization action.
- Execute VM migration or placement.
- Receive reward.
- Update policy parameters[16-18].
- The learning process continues until policy convergence.

### 6.5 Closed-Loop Integration

Unlike existing cloud scheduling approaches, the proposed framework continuously exchanges information among the forecasting, scheduling, and reinforcement learning modules. Updated infrastructure observations are fed back into the forecasting model, allowing the prediction model and scheduling policies to evolve as workload characteristics change. This closed-loop interaction enables adaptive optimization and long-term resource efficiency.

### 6.6 Computational Complexity

The computational complexity of the proposed framework is determined by the forecasting, scheduling, and reinforcement learning modules. The overall complexity can be expressed as the sum of the individual computational costs.

**SAEF Module**

The forecasting model processes historical workload sequences over a prediction window of length T. Its computational complexity is

$O(nT)$

where:

n is the number of workload samples,

T is the forecasting window length.

**HASE Module**

The scheduling engine performs multi-objective optimization over the candidate scheduling solutions. Assuming efficient sorting and scheduling operations, its computational complexity is

$O(n \log n)$

where nnn denotes the number of tasks.

GA-MADRL Module

Graph embedding requires processing all graph edges, resulting in

$O(|E|d)$

where:

$|E|$ is the number of graph edges,

d is the embedding dimension.

The multi-agent reinforcement learning process performs policy updates across all agents during multiple training episodes. Therefore, its computational complexity is

$$O(Nagents \times Episodes)$$

where:

Nagents is the number of learning agents,

Episodes represents the total number of training episodes.

Overall, the proposed framework exhibits polynomial computational complexity and remains scalable for heterogeneous cloud environments. The modular architecture also allows independent optimization of each component without affecting the remaining framework.

## 7. Experimental Methodology

### 7.1 Experimental Environment

This chapter presents the experimental methodology adopted to evaluate the proposed eco-aware cloud optimization framework. The evaluation is structured across three operational layers: forecasting, scheduling, and consolidation, enabling a comprehensive assessment of both predictive accuracy and system-level efficiency. The study reports repeated-seed results with 95% confidence intervals to ensure statistical reliability and reproducibility.

The experimental design follows a layered validation approach. First, workload forecasting is assessed using Mean Absolute Percentage Error (MAPE). Second, task scheduling efficiency is evaluated through power footprint in watts. Third, VM consolidation effectiveness is measured using Power Usage Effectiveness (PUE), which captures infrastructure overhead reduction.

### 7.2 Experimental Pipeline

The implementation begins with ingestion of the cloud trace data, where the raw dataset is standardized into four internal fields: time, virtual machine identifier, CPU utilization, and memory utilization. The data are then sorted by VM and time to preserve temporal consistency before constructing spatiotemporal tensors using a sliding window of 12 time steps and a prediction horizon of 6 steps

After preprocessing, the dataset is split into training, validation, and test segments using a 70:15:15 partition. Although the final evaluation table is reported on test behavior, the repeated-seed execution framework ensures stable performance estimation across 10 independent runs. This design reduces the impact of random initialization and supports statistically grounded comparison.

### 7.3 Methodological Components

#### 7.3.1 Forecasting Layer

The forecasting stage uses a sequential predictor based on an LSTM architecture. The proposed eco framework is compared against a vanilla LSTM baseline, and performance is measured using MAPE. The reported results show that the proposed method achieves a markedly lower forecast error than the baseline across all seeds, indicating stronger temporal prediction capability.

#### 7.3.2 Scheduling Layer

The scheduling stage applies a minimum-minimum style allocation strategy to map tasks to virtual machines. The proposed scheduling component, described as an energy-aware scheduler, is evaluated against a baseline scheduling profile. The metric of interest is power footprint in watts, where lower values indicate better efficiency.

#### 7.3.3 Consolidation Layer

The consolidation stage uses a First-Fit Decreasing strategy for baseline allocation and compares it against the proposed energy-aware consolidation approach. The evaluation metric is PUE, which reflects the ratio of total facility energy to useful IT energy. Lower PUE indicates improved operational efficiency and lower overhead.

### 7.4 Statistical Validation

To ensure that the observed improvements are not due to random chance, the framework is evaluated across 10 random seeds, and 95% confidence intervals are computed using the Student's t-distribution. Additionally, paired t-tests are used to compare the proposed method with the baseline for each objective layer. This makes the evaluation suitable for publication-quality empirical reporting.

### 7.5 Results and Interpretation

The experimental output shows that the proposed eco framework outperforms the baseline consistently across all three objectives. In forecasting, the proposed method reduces MAPE from 13.86% to 2.28%, demonstrating substantially improved prediction quality. In scheduling, power consumption decreases from 1204.0 W to 1099.3 W, showing reduced operational energy demand. In consolidation, the framework improves PUE from 1.285 to 1.151, indicating lower infrastructure overhead

**Table 7.1 Experimental Results**

| Objective Layer | Metric | Proposed Eco Framework μ±σ\mu \pm \sigmaμ±σ [95% CI] | Baseline Profile μ±σ\mu \pm \sigmaμ±σ | Statistical Validation |
|---|---|---|---|---|
| O1: Forecasting | Forecast Error (MAPE) | 2.28% ± 0.03% [±0.026%] | 13.86% ± 0.10% | p<0.001p < 0.001p<0.001 |
| O2: Scheduling | Power Footprint (Watts) | 1099.3 W ± 1.4 W [±1.08 W] | 1204.0 W ± 0.8 W | p<0.001p < 0.001p<0.001 |
| O3: Consolidation | Power Usage Effectiveness (PUE) | 1.151 ± 0.000 [±0.0003] | 1.285 ± 0.001 | Overhead reduction: 47.1% |

The results indicate that the proposed framework is not only more accurate but also more energy-efficient

across the cloud workflow. The reduction in PUE suggests that the consolidation strategy contributes meaningfully to operational sustainability, while the forecasting improvement likely supports better downstream scheduling decisions.

### 7.6 Discussion

The evaluation confirms that integrating forecasting, scheduling, and consolidation into a single eco-aware workflow provides measurable benefits. The forecasting layer improves the quality of workload estimation, the scheduling layer reduces immediate power consumption, and the consolidation layer lowers infrastructure overhead. Together, these results support the claim that the proposed framework is suitable for energy-conscious cloud optimization

### 7.2 Evaluation Metrics

The proposed framework is evaluated using standard cloud computing performance metrics.

**Resource Utilization**

$$RU = (\Sigma(i=1 \rightarrow N)\ U\_i) / N$$

U_i: Utilization of i-th resource

**Energy Consumption**

The total energy consumed by cloud infrastructure is computed throughout the scheduling interval to evaluate energy-efficient resource management.

**Makespan**

Makespan is defined as the completion time of the final scheduled task and represents the overall execution efficiency of the scheduling algorithm.

**SLA Violation Rate**

The percentage of tasks violating predefined Quality of Service constraints is measured to evaluate scheduling reliability.

**Forecasting Accuracy**

Prediction quality is assessed using Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE).

### 7.3 Comparative Evaluation

The effectiveness of the proposed framework is assessed by comparing it with representative heuristic, metaheuristic, and learning-based scheduling techniques. The comparison focuses on forecasting capability, scheduling quality, VM migration efficiency, energy consumption, and resource utilization under heterogeneous cloud workloads.

The evaluation methodology is designed to demonstrate the contribution of each proposed module within the overall closed-loop optimization framework.

## 8. Discussion

The proposed architecture integrates workload forecasting, adaptive scheduling, and reinforcement learning within a single optimization framework. Unlike conventional cloud management systems, forecasting outputs directly influence scheduling decisions, while execution outcomes continuously improve prediction accuracy and resource allocation policies through feedback-driven learning.

The modular design provides several practical advantages. First, workload prediction enables proactive resource provisioning, reducing the likelihood of infrastructure overload during workload spikes. Second, forecast-aware scheduling improves resource allocation by considering anticipated demand rather than relying solely on current infrastructure status. Finally, graph-based reinforcement learning enables adaptive VM placement and consolidation through continuous interaction with heterogeneous cloud environments.

The proposed framework also offers improved scalability because each module can be independently upgraded without affecting the remaining optimization pipeline. For example, future forecasting models or reinforcement learning algorithms can replace existing implementations while preserving the overall closed-loop architecture.

Although the proposed framework introduces additional computational complexity compared with traditional scheduling algorithms, the expected improvement in resource utilization, energy efficiency, and SLA compliance justifies the additional processing overhead in large-scale cloud infrastructures.

## 9. Conclusion and Future Work

This paper presented an Intelligent Closed-Loop Multi-Objective Load Balancing Framework for heterogeneous cloud computing environments. The proposed architecture integrates Spatiotemporal Adaptive Ensemble Forecasting (SAEF), the Hybrid Adaptive Scheduling Engine (HASE), and Graph-Augmented Multi-Agent Deep Reinforcement Learning (GA-MADRL) into a unified optimization framework for predictive and adaptive cloud

resource management.

Unlike conventional cloud schedulers that perform forecasting, scheduling, and resource optimization independently, the proposed framework establishes a continuous feedback mechanism that enables all optimization modules to learn from infrastructure execution outcomes. The architecture supports proactive workload management, adaptive task scheduling, intelligent VM placement, and continuous policy refinement, providing a scalable solution for modern cloud data centers.

Future work will focus on extending the framework to federated cloud environments, edge-cloud collaboration, carbon-aware scheduling, explainable reinforcement learning, and real-time deployment using Kubernetes and OpenStack platforms. Further investigation of large language models and foundation models for autonomous cloud orchestration also represents a promising research direction.

## References


1. Sanjalawe Y. Smart load balancing in cloud computing: Integrating feature selection with advanced deep learning models (SLADRO). Sci Rep. 2025;15:12419607. doi:10.1038/s41598-025-12419-6
2. Narsimhulu B, et al. A hybrid RL–GA–LSTM–AE framework for energy-aware task scheduling in cloud computing. Sci Rep. 2026;16:43108. doi:10.1038/s41598-026-43108-4
3. Guo R, et al. Boosting integration for cloud resource prediction: MHST-GB ensemble network. Preprints. 2025;202509.2313. doi:10.20944/preprints202509.2313.v1.
4. Zhang L, et al. Regime-adaptive weighted ensemble learning for computing-driven dynamic load forecasting in AI data centers. arXiv. 2026;2604.27207
5. Kaim A, et al. Ensemble CNN attention-based BiLSTM deep learning architecture for multivariate cloud workload prediction. Proc ICTAI. 2023;3571306:3571433. doi:10.1145/3571306.3571433
6. Wang X, et al. CFL-LSTM: Collaborative cloud server load prediction using dynamic correlation mining and gated attention. Big Data Min Anal. 2026;9:20250097. doi:10.26599/BDMA.2025.9020097
7. Ibrahim M, Askar A. Attention-based workload prediction and dynamic resource allocation for heterogeneous computing environments. Sci Rep. 2026; 16:38622. doi:10.1038/s41598-026-38622-4
8. Bodra D, et al. Machine learning-based cloud resource allocation algorithms: a comprehensive comparative review. Front Comput Sci. 2025; 17:1678976. doi:10.3389/fcomp.2025.1678976
9. Khan AR. Dynamic multi-objective task scheduling in cloud computing using reinforcement learning for energy and cost optimization (RL-MOTS). Sci Rep. 2025; 15:29280. doi:10.1038/s41598-025-29280-z
10. Sudhakar P, et al. Reinforcement learning based multi-objective task scheduling for energy efficient and cost effective cloud edge computing. Sci Rep. 2025; 15:25666. doi:10.1038/s41598-025-25666-1
11. Liu Y, et al. PHWSOA: A Pareto-based hybrid whale-seagull scheduling for multi-objective tasks in cloud computing. arXiv. 2025;2512.09568.
12. Chen H, et al. D3QN-guided sand cat swarm optimization with hybrid exploration for multi-objective cloud task scheduling. Algorithms. 2026;19(4):321. doi:10.3390/a19040321
13. Alghamdi S, et al. A multi-objective optimization-based container cloud resource scheduling method (HHO-GWO). Future Internet. 2026;18(1):58. doi:10.3390/fi18010058
14. Ibrahim A, Askar A. Multi-objective reinforcement learning: a comprehensive survey. Syst Sci Control Eng. 2026;14(1):2672169. doi:10.1080/21642583.2026.2672169
15. Niyasudeen F, et al. Energy-aware load balancing in cloud environments using graph neural networks and deep reinforcement learning. Sci Rep. 2026;16:51660. doi:10.1038/s41598-026-51660-2
16. Zhang Q, et al. AGMARL-DKS: An adaptive graph-enhanced multi-agent reinforcement learning dynamic Kubernetes scheduler. arXiv. 2026;2603.12031
17. Li W, et al. Neuro-symbolic constrained optimization for cloud application deployment via graph neural networks and satisfiability modulo theory. arXiv. 2025;2511.23109
18. Wang J, et al. Diffusion-based solver for CNF placement on the cloud-continuum. arXiv. 2025;2511.01343
19. Bodra D, et al. Machine learning-based cloud resource allocation algorithms: a comprehensive comparative review. Front Comput Sci. 2025;17:1678976. doi:10.3389/fcomp.2025.1678976
20. Khan AR, et al. A fusion deep Q-learning and particle swarm optimization algorithm for adaptive resource allocation in cloud computing. Sci Rep. 2026;16:33498. doi:10.1038/s41598-025-33498-2
21. Patel, S. K., Prajapati, J. B., & Patel, R. S. (2012). SEO and content management system. International Journal of Electronics and Computer Science Engineering, 1(3), 953-959.
22. Patel, R.S., Chauhan, J.A., Patel, K.C., Bhavsar, K. (2024). Exploring the Effectiveness of On-Page SEO for Webpage Ranking: A Critical Study. In: Rajagopal, S., Popat, K., Meva, D., Bajeja, S. (eds) Advancements in Smart Computing and Information Security. ASCIS 2023. Communications in Computer and Information Science, vol 2040. Springer, Cham. https://doi.org/10.1007/978-3-031-

59107-5_11
23. Patel, R. S., Chauhan, J. A., Patel, K. C., Patel, D. J., Patel, S. J., & Lakhatariya, P. T. (2026). DESIGN AND IMPLEMENTATION OF HETEROGENEOUS FRAMEWORK FOR ORGANIC WEB PAGE RANKING. Genetics and Molecular Research.
24. Parikh, S., Patel, R. S., & Chauhan, J. A. A survey on effective on-page and off-page search engine optimization factors to improve the website visibility on google search engine platform. AEGAEUM J, 9(04), 296-303.
25. Bhavsar, K., Patel, A., Patel, K., Patel, R. (2024). Exploring Data Ownership in Web 2.0 and Web 3.0 with the Integration of Blockchain Technology. In: Rajagopal, S., Popat, K., Meva, D., Bajeja, S. (eds) Advancements in Smart Computing and Information Security. ASCIS 2023. Communications in Computer and Information Science, vol 2039. Springer, Cham. https://doi.org/10.1007/978-3-031-59100-6_27
26. Patel, K., Patel, A., Patel, B., Patel, R. (2024). A Comparative Analytics for Dynamic Load Balancing Mechanisms Intended to Improve Task Scheduling in Cloud Computing. In: Rajagopal, S., Popat, K., Meva, D., Bajeja, S. (eds) Advancements in Smart Computing and Information Security. ASCIS 2023. Communications in Computer and Information Science, vol 2039. Springer, Cham. https://doi.org/10.1007/978-3-031-59100-6_26
27. Patel, K., Patel, A., & Patel, B. (2024). A Comparative Analytics for Dynamic Load Balancing Mechanisms Intended to Improve Task Scheduling in Cloud Computing With Security Challenges. Convergence Strategies for Green Computing and Sustainable Development, 182-201.