iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 Issue: VI Month of publication: June 2025

DOI: https://doi.org/10.22214/ijraset.2025.71746

A Novel Approach to Dynamic Cloud Load Balancing Based on Reinforcement Learning

Rudresh Shah[1], Sachin Solanki[2], Sourabh Dave[3], Gaurav Sharma[4]
[1, 2, 3, 4] *Department of Computer science and Engineering, Medi-Caps University, Indore, India*

***Abstract:** In cloud computing, effective load balancing has to be managed to attain optimal resource utilization with minimal response times as it decreases the likelihood of server overload. Algorithms used in the traditional load balancing-like round robin or the least connections are the least flexible and thus cannot keep pace with the dynamic nature of various characteristics of workloads in a cloud environment. In this paper, an innovative adaptive load balancing framework that is based on reinforcement learning (RL) is provided for the purpose of solving these challenges. Depending on the distribution of observed real-time system performance, this system learns and gets better over time. It then makes decisions depending on resource availability and traffic patterns. To disperse resource utilisation across servers and minimise latency, our system is built to dynamically reallocate tasks. According to the findings of the experiment, the suggested Modified RL-based load balancer outperforms both reinforcement learning -based load balancer and conventional methods in parameter of resource usage, response time, and workload adaption. It also indicates that AI-based solutions can make the cloud infrastructure not only efficient but also scalable.*

***Index Terms:** AI-driven Load Management, Resource Optimization, Adaptive Algorithms, Cloud Computing, Load Balancing, Reinforcement Learning*

I. INTRODUCTION

Cloud computing as the backbone for many modern services in the current digital transformation era, covering everything from storage of data and on-demand application delivery, the increasing demand for cloud services creates the need for efficiency, scalability, and reliability in cloud infrastructure. Load balancing (LB) is spreading workloads across many servers so that no one server gets overloaded while others have resources to spare [1]. That's a key factor influencing cloud performance.

The common traditional techniques [2], [3] used for load balancing widely in the cloud include least connections, round-robin and weighted balancing. Because these algorithms depend on certain rules to perform operations based on a set of fixed workloads, they tend to be straightforward and quite easy to implement. However, a cloud environment possesses intrinsic dynamism resulting from network traffic, user demand, or the application itself. This fluctuation typically results in performance challenges for conventional load balancing techniques that lack the adaptability to respond to changing situations in real time. Consequently, server overcrowding, prolonged response times, and suboptimal resource utilisation are prevalent challenges linked to cloud systems.

There is also a solution, artificial intelligence [4], which promises adaptive and intelligent load management [5] strategies in response to the above issues. Reinforcement Learning has recently drawn much attention for its capability of learning through or by interacting with the environment and decision-making based upon experience [6]. In contrast to supervised learning, reinforcement learning (RL) agents acquire knowledge by error and trial, continually optimising their actions to enhance long-term rewards. Reinforcement Learning's flexibility makes it a good fit for ever-changing settings like cloud computing, where adjustments are needed in real-time due to constantly changing conditions.[7].

This research proposes a new architecture A Novel Approach to Dynamic Cloud Load Balancing Based on Reinforcement Learning. This framework employs reinforcement learning algorithms to optimise workload distribution by learning and adapting to system performance indicators of traffic patterns. This system operates in real-time, monitoring network throughput, CPU utilisation, and server response times to judiciously distribute workloads across servers, minimise response times, prevent server overload, and optimise resource utilisation.

Our framework adapts to changes in workload and infrastructure, unlike other load balancing techniques. Such algorithms are fine in steady states but unresponsive when there is a rapid change in environment. The framework is continuously learning from the environment; thus, new strategies in load balancing are updated in real time by incorporating the latest data. This will lead to more responsive and resilient systems that will then perform well under multiple divergent workloads.

We show a set of tests to evaluate the feasibility of the Modified Reinforcement Learning -based load balancer instead of the classic ones. In general, we were able to get quite notable improvements in regard to the key performance metrics in question, especially regarding the response time, resources being used, and the efficiency of the system at large. Hence, these results conclude that AI-driven load balancing would considerably improve the performance as well as scalability of the cloud systems, especially within such dynamic workloads.

The remainder of this work is structured as follows: In Section 2, we examine pertinent literature on LB approaches and their constraints in dynamic situations. Section 3 delineates the revised reinforcement learning mechanism employed in our proposed framework. Section 4 delineates our experimental configuration and findings on the performance comparison between our Modified RL-based technique and conventional algorithms. Finally, Section 5 ends this work and explores prospective avenues for further research on this topic.

II. LITERATURE REVIEW

One of the cores for distributed systems, especially in cloud computing, is knowing how to distribute tasks and computational resources across multiple servers efficiently.

Certain LB strategies have been thoroughly researched and applied; nevertheless, their inflexible, static characteristics render them ill-suited for the dynamic and unpredictable nature of contemporary Cloud systems. This paper starts with providing an overview of traditional load balancing techniques, followed by the AI-based current approaches and focusing on Reinforcement Learning in load balancing specifically.

A. Load Balancing Techniques

Traditional techniques for LB distribute predetermined numbers of jobs to servers using fixed, predefined algorithms. Some of the common techniques that are typically used are:

- Round-Robin [8]: A simple algorithm that distributes incoming requests sequentially across servers. In stable environments, it is capable of even distribution but fails in the event of changing workload and server capacity differences.
- Least Connections [9]: The server with the fewest active connections at the moment receives new requests from this algorithm. Although it is more dynamic than round-robin, when workload factors or server capacities diverge, this may lead to an inefficient use of resources [10].
- Weighted Load Balancing [11]: Here, a weight is assigned to each server according to its performance or capacity. They are given the tasks based on these weights. Although it is versatile, it may not be able to adjust well to shifts in workload or server performance because it is dependent on static weights.

Since workloads constantly change very fast in dynamic cloud scenarios and server performance can, for instance, change per hour, it may not be effective with these traditional systems at this level. There will always be a demand for intelligent-based load balancing solutions in such cloud infrastructures, especially as their complexity increases, to reflect real-time responses to changing situations.

B. AI-Based Approaches to Load Balancing

The potential for AI to enhance decision-making in dynamic contexts has recently attracted a lot of attention in load balancing. In order to optimise the distribution of tasks and anticipate patterns in workload, AI-based techniques usually employ machine learning models.

- Machine learning-based predictive load balancing [12]: Most of the articles evaluated supervised learning approaches by working with the predictability of traffic or server loads while being trained using historical records. Techniques like neural network, regression-based models estimate peak usage so that relevant resources are put in anticipation of such occurrences; highly data-driven approaches and most of the times may be useless in situations where this information is unavailable or about something new.
- Self Adaptive Load Balancing [13]: It was applied by AI methods that include genetic algorithms, as well as fuzzy logic in the development of adaptive self-adjusting load balancing with dynamic adaptation of its parameters according to real conditions, but these methods, typically, do not involve any learning and improvement performance in time. These AI

Table 1. Comparative study

Study	Methodology	Key Findings	Strengths	Limitations
[17]	Proposed a hybrid load balancing model combining RL and fuzzy logic.	The model adapts resource allocation based on real-time metrics, improving overall performance.	Effective in dynamic environments; combines the strengths of RL and fuzzy logic for better adaptability.	Limited scalability in very large cloud environments; relies on fuzzy logic parameters that require tuning.
[18]	Developed an RL-based framework that continuously monitors and adjusts task distribution.	Showed significant improvements in server utilization and reduced response times compared to static algorithms.	Real-time monitoring leads to adaptive adjustments; effective in environments with variable workloads.	Lack of large-scale implementation data; the need for extensive training time for the RL model.
[19]	Implemented a deep reinforcement learning (DRL) model for load balancing in a cloud environment.	Confirmed the effectiveness of DRL in optimizing resource allocation and minimizing latency in a large-scale setting.	Large-scale evaluation provides strong evidence of RL effectiveness; robust performance under varying workloads.	Complexity of DRL models may lead to longer training times; potential overfitting with insufficient training data.
[20]	Combined RL with container orchestration to enhance load balancing across microservices.	Demonstrated improved resource management and adaptability in cloud environments utilizing microservices architecture.	Integration with container orchestration systems leads to better efficiency and scalability.	Focused primarily on microservices, which may not generalize to all cloud environments; reliance on specific orchestration tools.
[21]	Explored deep reinforcement learning techniques for optimizing load balancing in large-scale cloud settings.	Highlighted DRL’s ability to address complexity and dimensionality challenges in cloud environments.	Offers solutions to real-time performance challenges; potential for continuous learning and adaptation.	Still faces challenges in computational overhead and training time; practical implementation needs further validation.

Based approaches do better than the old guardians; however, all too often, they are strangulated by invariant properties of their frameworks which might not cope with unforeseen events. This is where RL provides a particular advantage in enabling continuous adaptation in rapidly changing environments while learning.

C. Reinforcement Learning Based Approaches to Load Balancing

In summary, RL is the most promising AI approach developed for dynamic adaptive load balancing. It learns by trial and error according to real-time feedback it receives as it interacts with its environment. In such an RL framework, it has some agent that actually takes various factions within an environment and returns feedback in form of reward, which helps increase the effectiveness of decision over time.

Reinforcement Learning in Dynamic Environments [14]: Robotics, autonomous systems, and game AI are just a few of the dynamic decision-making contexts where reinforcement learning has proven effective. Through real-time work distribution modifications between servers, RL can be utilised in cloud computing to continuously analyse system performance. In dynamic cloud systems where workloads can change quickly and without warning, RL is especially well-suited.

Applications of RL in Networking [15]: In the context of software-defined networking, reinforcement learning has been recently proven to significantly contribute to adaptive routing, network traffic management, and load balancing. Additional applications of reinforcement learning also exist to enhance real-time routing of network traffic in order to boost flow, alleviate congestion, and optimize utilization.

Issues in Applying RL in Load Balancing [16]: Though promising, there are challenges in the implementation of RL for load balancing. The enormous dimensionality of cloud systems, typified by the simultaneous management of several servers, jobs, and performance measures, complicates the design of an efficient reinforcement learning agent. Furthermore, reinforcement learning methods may necessitate extensive training durations, and attaining real-time performance in large-scale systems presents significant challenges. Recent advancements in reinforcement learning for deep learning have successfully tackled various challenges by combining the strengths of deep learning with the flexibility of reinforcement learning.

Therefore, the above studies represent the rising interest in using RL to apply to load balancing in dynamic cloud environments. More importantly, key results show improvement in adaptability, efficiency, and overall performance compared with traditional methods. In addition, inherent drawbacks such as scalability issues, demand of large training data, and computational overhead still persist. Further research in this area will be necessary to overcome the mentioned challenges and to improve practical applications of RL-based load balancing solutions.

D. Research Gaps and Motivation

Several gaps still need to be filled in research literature, starting with what has already been reported: related work on the promise of using AI and RL for load balancing. Most of the proposed AI-driven systems focus on job allocation but do not present integrated solutions capable of capturing the full complexity of a dynamic cloud environment. It also stands for very limited scope for deep investigation to be shown in order to prove that the RL-based load balancing functions properly within real cloud infrastructures. Lastly, there is a developing literature in integrating reinforcement learning with other cloud management technologies such as container orchestration and microservices. To achieve this, an adapted reinforcement learning-based load balancing framework is developed and investigated by this research, capable of making a dynamic real-time assignment of tasks while reacting to changing workloads. The core benefit behind the system is the continuous learning and improvement over time, as in reinforcement learning, that can ensure an effective and scalable solution for today's cloud computing environments.

III. PROPOSED METHODOLOGY

In this section, the Modified Reinforcement Learning-Based Adaptive Load Balancing framework is introduced. This can be realized into three parts, namely: system architecture, reinforcement learning model, and experimental setup. All these are discussed in greater detail below.

A. Architecture

The proposed structure of the load balancing framework facilitates the dynamic assignment of jobs to a cluster of cloud servers according to workload conditions and real-time system performance. This architecture comprises three components.

The Task Scheduler: It is the component responsible for incoming requests and spreading jobs among the available servers. To allocate tasks intelligently, the task scheduler is in constant communication with the RL agent.

Server Pool: This is a set of servers (VM) in which tasks are executed. Each server had different performance characteristics that influenced its ability to handle incoming tasks such as CPU utilization, memory, network bandwidth, etc.

It tracks real-time metrics like response time and utilization of the server. Therefore, it dynamically decides the allocation of a particular server for a particular task. An agent learns through experiences, thereby continually improving its strategy about load balancing regarding any type of feedback received from its environment.

The system is of feedback loop form: the RL agent observes the server pool's state, including resource utilization and completion times for tasks, and takes actions by assigning a task to a particular server and, upon reception, receives rewards concerning the performance of the system in absolute terms, such as higher rewards in case the response time is lower. Through this, the optimal policy for the distribution of tasks is learned by the RL agent.

B. *Modified Reinforcement Learning Model*

The environment simulates a cloud system where multiple servers handle varying workloads. The states, actions, and rewards are defined to reflect the dynamics of cloud computing, such as resource utilization, task arrival rates, and server capacities. To structure this Modified RL-based load balancing framework model based on the Q-learning algorithm, we can break it down into its core elements and functions, designing it with the following components:

Model Components

State (S): The state is defined to encapsulate critical information about the system, including:

Resource Utilization: Network, memory, and percentage of CPU usage for each server.

Task Metrics: The number of active jobs, their types, and resource requirements.

System Load: Overall load metrics indicating the total number of tasks across all servers and response times.

Action (A): Actions involve deciding how to allocate incoming tasks to the available servers. Possible actions include:

Assigning a task to a specific server.

Migrating tasks from one server to another to balance the load.

Scaling resources up or down based on current demand.

Reward (R): Rewards are designed to incentivize optimal load balancing, incorporating factors such as:

Response Time: A lower response time yields a higher reward.

Resource Utilization Efficiency: Balancing resource use across servers can maximize efficiency and minimize waste.

Task Completion: Completing more tasks within a given time frame increases the reward.

Overload Penalties: Servers experiencing overload receive a negative reward, encouraging avoidance of such states.

Q-Value Update: The Q-learning algorithm is modified to include additional factors that address load balancing specifically. The Q-value update rule remains similar but can be enhanced with mechanisms like eligibility traces or experience replay to improve learning efficiency. The basic formula can be adapted to include weighted factors reflecting the importance of response time and resource utilization:

The Q-learning algorithm modifies the Q-value according to the subsequent formula:

$$Q(x,y) \leftarrow \alpha[R + \gamma \max_a Q(x',y') - Q(x,y)] + Q(x,y) \ldots\ldots (1)$$

Where:

- For the next state x′, $\text{Max}_{a'}$Q(x′,y’) is the maximum expected future reward.
- γ is the discount factor, which balances immediate and future rewards.
- R is the reward received after performing action y in state x.
- α is the learning rate, indicating how much new information supersedes old data.
- Q(x,y) is the Q-value for state x and action y.

The Q-learning agent continuously enhances its policy by updating the Q-values until it identifies the optimal method for distributing tasks across servers.

Model Workflow

1. Initialize Q-values: Start with arbitrary Q-values for each state-action pair.
2. Observe State (S): Gather real-time metrics of the environment to define the current state.
3. Select Action (A): Choose an action based on the Q-values (e.g., using an ϵ\epsilonϵ-greedy policy to explore/exploit).

4. Execute Action: Assign the task to the selected server.
5. Observe Reward (R): Evaluate the system's response and compute the reward.
6. Update Q-Value: Update the Q-value based on the observed reward and the future state.
7. Repeat: Continue the process to iteratively improve Q-values and the load balancing policy.

Training Process

The model is trained by continuously interacting with the environment:

By exploring actions and updating Q-values, the agent learns to select actions that optimize load balancing.

The agent gradually converges to an optimal strategy that reduces server overload, minimizes response times, and enhances resource utilization.

C. Experimental Setup

We emulate a cloud environment using the CloudSim simulator to evaluate the performance of the suggested RL framework for LB.

Experimental Configuration:

Cloud Environment: Simulation of 20 servers with different CPU and RAM capacity, with heterogeneous cloud environment.

Workload: A dynamic workload generator that emulates incoming workloads, fluctuating over time with both consistent and sporadic traffic patterns to evaluate the framework's adaptability.

The Reinforcement Learning -based framework is evaluated against conventional LB approaches, including weighted load balancing, least connections, and round-robin, which act as benchmarks for assessing the efficacy of the RL approach.

Performance measures: The performance measures employed include the following:

Response Time: The average duration required for service delivery subsequent to submission to the cloud system. Resource Utilisation: This denotes the percentage of network, memory, and CPU resources employed by servers.

Task Completion Rate: The quantity of tasks finalised within a designated timeframe.

The Reinforcement Learning agent undergoes training through numerous iterations in contrast to the baseline methods. The evaluation assesses if the RL agent has reduced reaction times, optimised resource utilisation, and managed variable workloads well.

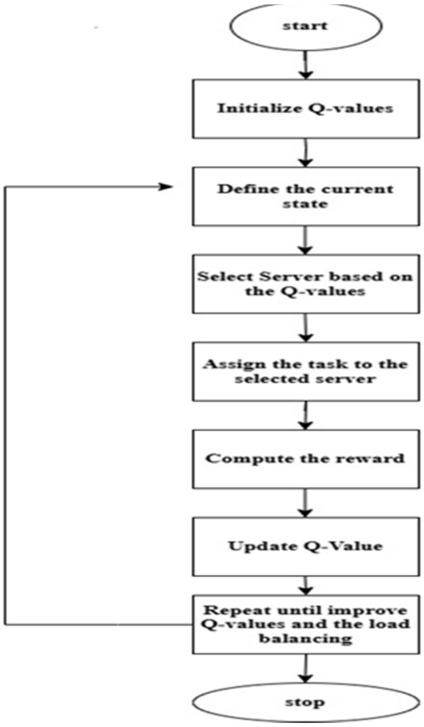
start
Initialize Q-values
Define the current state
Select Server based on the Q-values
Assign the task to the selected server
Compute the reward
Update Q-Value
Repeat until improve Q-values and the load balancing
stop

Fig. 1 Model Workflow

IV. EXPERIMENTAL RESULTS AND DISCUSSION

This Q-learning algorithm has been implemented in Python by taking the libraries like OpenAI Gym [23] and NumPy [22] for numerical computations for handling the Reinforcement Learning environment. Then, CloudSim [24] is used for the simulation of the cloud infrastructure to allow a detailed model of cloud resource management and workload distribution.

This section introduces an exploration of experimental results. The proposed RL-based load balancing is compared against traditional methods of LB, like weighted load balancing, least connections, and round-robin. Three notable performance parameters are highlighted: response time, resource utilization, and the rate of task completion.

A. Response Time

The proposed Reinforcement Learning -based load balancer should prioritise minimising the response time for job completion upon submission to the system. Figure 2 illustrates the comparative average response times of the RL-based framework against various conventional load balancing methods across varied workload intensities. Figure 2 illustrates that the Modified Reinforcement Learning -based framework outperforms other conventional LB algorithms under high traffic scenarios. In contrast, least connections and round-robin exhibit significant increases in response times as demand escalates. The system utilising reinforcement learning dynamically grows and adapts to perform tasks more effectively. This generally reduces and stabilises responses, even under conditions of high-intensity workload fluctuations.

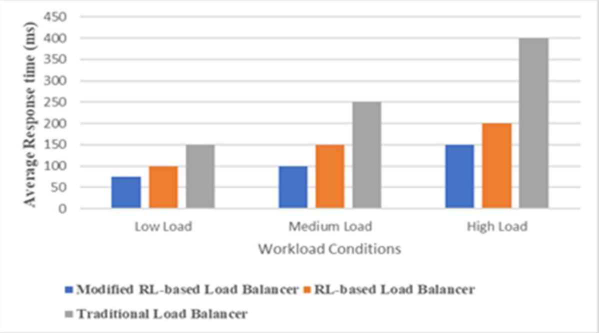

Fig 2. Average response times for various load balancing techniques under varied workloads are compared.

B. Resource Utilization

One of the major performance metrics of load-balancing algorithms is resource utilization. Resource utilization here means how much the servers in cloud infrastructure utilize the CPU, memory, and network resources. Figure 3 shows average resource utilization between Reinforcement Learning -based approach and traditional LB methods. It is proven that the proposed Modified RL-based system has more balance and usage in terms of resource utilization compared to other systems. The RL agent prevents overload on one server by continuing to learn from system performance data, ensuring that it spreads tasks out as evenly as possible and minimizes the idle times across the pool of servers. Traditional algorithms utilize some servers to the point that they are overutilized while underutilizing the others, leading to rather less-than-optimal results.

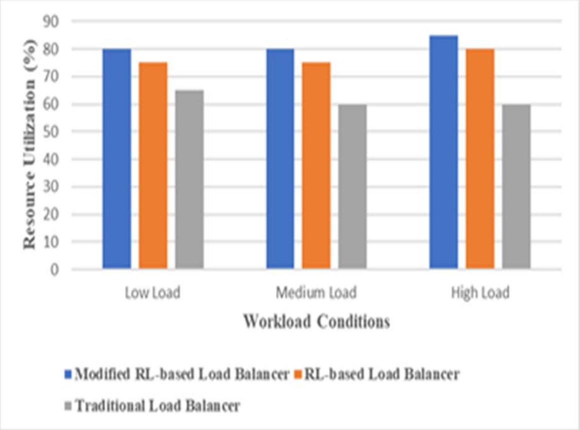

Fig 3. Resource usage for various load balancing techniques is compared.

C. Task Completion Rate

Percentage completed task is the number of jobs completed within a fixed time frame, which defines overall system throughput. Figure 4 shows the comparison task completion rates for Reinforcement Learning based scheme with traditional approach: Overall, the result exhibits more task completion under higher task loads when applying RL techniques in the framework. Using real-time feedback, the dynamic shifting of task allocation prevents bottlenecks, leading to even distribution of tasks between the servers. This thus leads to a total increase in the number of tasks accomplished throughout this time frame.

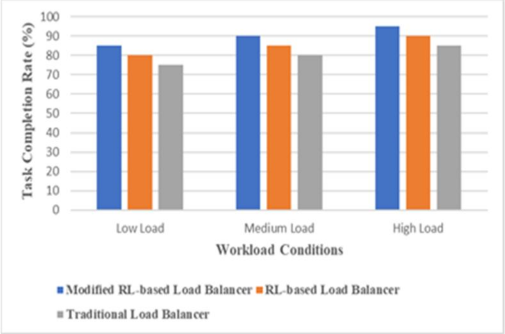

Fig 4. Task completion rates for various load balancing techniques are compared.

D. Discussion

The experimental results demonstrate that the suggested framework surpasses standard frameworks in reaction time, resource utilisation, and jobs accomplished per unit time based on metrics. These enhancements are especially evident under dynamic and heavily loaded conditions, where conventional algorithms may struggle with workload fluctuations. The rationale is rooted in the agent's capacity to continuously learn from the environment and adapt its decision-making in real time. The RL-based system may allocate the load while equilibrating short-term and long-term incentives, so circumventing the performance decrease often associated with static load balancing approaches.

This indicates that the RL-based framework has greater scalability than conventional methods. Owing to the intricacy and expanded scale of cloud infrastructures, the RL agent manages hundreds of servers and jobs without augmentations in response time or resource constraints, rendering it exceptionally suited for the contemporary cloud environment. Challenges, however, confront RL-based load balancing. A significant disadvantage is that the training duration for the RL agent to acquire an optimal policy might be extensive in large-scale environments and may necessitate substantial computer resources. Once trained, the RL agent can adjust to evolving conditions with minimal supplementary effort.

Another restriction is that the implementation of reinforcement learning may prove difficult in highly heterogeneous cloud settings characterised by significant variability in server capacities and network circumstances. Future research may focus on improving the flexibility of the RL model to increasingly complex circumstances. Overall, our results unequivocally demonstrate the potential of AI-driven methodologies, particularly reinforcement learning, for load balancing in cloud environments. This decreases response time, optimises resource utilisation, and enhances the scalability and adaptability of a system, rendering this solution highly attractive for dynamic cloud infrastructures.

V. CONCLUSION AND FUTURE WORK

The work introduces a new approach to dynamic load balancing in cloud computing through the use of Reinforcement Learning. Classic load balancing strategies, often based on round-robin and least-connection algorithms, perform well under stable or predictable conditions, but generally cannot adapt to changing, heterogeneous workloads-characteristic of most today's infrastructures. To alleviate this limitation, we developed an adaptive load balancing framework incorporating reinforcement learning, continuing to assimilate real-time system performance information into optimizing the distribution of tasks across different servers. Experimental results have shown the effectiveness of a reinforcement learning-based load balancer based on reduced response times, efficient resource usage, and higher completion rates of tasks compared with traditional methods.

The RL agent improved the workload management more than any static technique owing to its adaptive load balancing strategy that could change based on environmental variables. The study thus underlines the enormous potential of AI-driven load balancing technologies for significantly enhancing the performance and scalability of cloud computing systems. Promising as the idea may be, the proposed framework with reinforcement learning bases still opens many avenues in future research.

The shortcoming of this approach is the long-time taken before the agent, in reinforcement learning, successfully identifies an optimal policy, especially when it has to operate in large-scale cloud environments. Advanced reinforcement learning techniques like DRL can increase the learning speed for the agent and its capabilities to handle increasingly complex scenarios. More importantly, this could have the development of workload forecasting models within the paradigm of reinforcement learning to improve its flexibility, allow dynamic reallocations of tasks according to expected variations in demand.

REFERENCES

[1] S. G. FATIMA, S. K. FATIMA, S. A. SATTAR, N. A. KHAN, and S. ADIL, “CLOUD COMPUTING AND LOAD BALANCING,” INTERNATIONAL JOURNAL OF ADVANCED RESEARCH IN ENGINEERING &amp TECHNOLOGY, vol. 10, no. 2, Mar. 2019, doi: 10.34218/ijaret.10.2.2019.019.

[2] Divya and B. M. Goel, “A Fuzzy Logic Approach to Performance Analysis and Grading Through Benchmarking of Load Balancers,” in Proceedings - 2023 International Conference on Advanced Computing and Communication Technologies, ICACCTech 2023, 2023. doi: 10.1109/ICACCTech61146.2023.00109.

[3] S. G. Domanal, R. M. R. Guddeti, and R. Buyya, “A Hybrid Bio-Inspired Algorithm for Scheduling and Resource Management in Cloud Environment,” IEEE Trans Serv Comput, vol. 13, no. 1, 2020, doi: 10.1109/TSC.2017.2679738.

[4] D. Sheth, C. Sheth, and B. Patel, “Artificial Intelligence for Automatic Load Management,” Journal of Artificial Intelligence Research & Advances, 2021, doi: 10.37591/joaira.v8i3.92.

[5] A. Muhammad, A. A. Ishaq, I. O. B, and M. B. Idris, “Artificial Intelligence and Machine Learning for Real-time Energy Demand Response and Load Management,” Journal of Technology Innovations and Energy, vol. 2, no. 2, 2023, doi: 10.56556/jtie.v2i2.537.

[6] A. R. Khan, “Dynamic Load Balancing in Cloud Computing: Optimized RL-Based Clustering with Multi-Objective Optimized Task Scheduling,” Processes, vol. 12, no. 3, 2024, doi: 10.3390/pr12030519.

[7] Z. Wang, M. Goudarzi, M. Gong, and R. Buyya, “Deep Reinforcement Learning-based scheduling for optimizing system load and response time in edge and fog computing environments,” Future Generation Computer Systems, vol. 152, 2024, doi: 10.1016/j.future.2023.10.012.

[8] S. S. Priya and T. Rajendran, “Improved round-robin rule learning for optimal load balancing in distributed cloud systems,” International Journal of System of Systems Engineering, vol. 13, no. 1, 2023, doi: 10.1504/IJSSE.2023.10053120.

[9] M. Sholeh, W. Yahya, and P. Hari, “Implementasi Load Balancing menggunakan Algoritme Least Connection dengan Agen Psutils pada Web Server,” Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya, vol. 3, no. 1, 2019.

[10] M. ElGili Mustafa, “Load Balancing Algorithms Round-Robin (RR), Least-Connection and Least Loaded Efficiency,” International Journal of Computer and Information Technology, vol. 1, no. 1, 2017.

[11] K. Hagiwara, Y. Li, and M. Sugaya, “Weighted Load Balancing Method for Heterogeneous Clusters on Hybrid Clouds,” in Proceedings - IEEE International Conference on Edge Computing, 2023. doi: 10.1109/EDGE60047.2023.00035.

[12] M. Ashawa, O. Douglas, J. Osamor, and R. Jackie, “Improving cloud efficiency through optimized resource allocation technique for load balancing using LSTM machine learning algorithm,” 2022. doi: 10.1186/s13677-022-00362-x

[13] V. Lavanya, M. Saravanan, and E. P. Sudhakar, “Self - Adaptive Load Balancing Using Live Migration of Virtual Machines in Cloud Environment,” Webology, vol. 17, no. 2, 2020, doi: 10.14704/WEB/V17I2/WEB17064.

[14] H. Sheng, W. Zhou, J. Zheng, Y. Zhao, and W. Ma, “Transfer Reinforcement Learning for Dynamic Spectrum Environment,” IEEE Trans Wirel Commun, vol. 23, no. 2, 2024, doi: 10.1109/TWC.2023.3289502.

[15] M. Naeem, S. T. H. Rizvi, and A. Coronato, “A Gentle Introduction to Reinforcement Learning and its Application in Different Fields,” IEEE Access, vol. 8, 2020, doi: 10.1109/ACCESS.2020.3038605

[16] D. Wu et al., “Reinforcement learning for communication load balancing: approaches and challenges,” 2023. doi: 10.3389/fcomp.2023.1156064.

[17] A. Ishaq Khan et al., “Intelligent Cloud Based Load Balancing System Empowered with Fuzzy Logic,” Computers, Materials & Continua, vol. 67, no. 1, pp. 519–528, 2021, doi: 10.32604/cmc.2021.013865.

[18] M. Arvindhan and D. R. Kumar, “Adaptive Resource Allocation in Cloud Data Centers using Actor-Critical Deep Reinforcement Learning for Optimized Load Balancing,” International Journal on Recent and Innovation Trends in Computing and Communication, vol. 11, no. 5s, pp. 310–318, May 2023, doi: 10.17762/ijritcc.v11i5s.6671.

[19] M. Xiang, M. Chen, D. Wang, and Z. Luo, “Deep Reinforcement Learning- based load balancing strategy for multiple controllers in SDN,” e-Prime - Advances in Electrical Engineering, Electronics and Energy, vol. 2, p. 100038, 2022, doi: 10.1016/j.prime.2022.100038.

[20] P. V. Lahande, P. R. Kaveri, J. R. Saini, K. Kotecha, and S. Alfarhood, “Reinforcement Learning Approach for Optimizing Cloud Resource Utilization With Load Balancing,” IEEE Access, vol. 11, pp. 127567–127577, 2023, doi: 10.1109/ACCESS.2023.3329557.

[21] K. Siddesha, G. V. Jayaramaiah, and C. Singh, “A novel deep reinforcement learning scheme for task scheduling in cloud computing,” Cluster Comput, vol. 25, no. 6, pp. 4171–4188, Dec. 2022, doi: 10.1007/s10586-022-03630-2

[22] C. R. Harris et al., “Array programming with NumPy,” Nature, vol. 585, no. 7825, pp. 357–362, Sep. 2020, doi: 10.1038/s41586-020-2649-2

[23] T. Beysolow II, Applied Reinforcement Learning with Python. Berkeley, CA: Apress, 2019. doi: 10.1007/978-1-4842-5127-0.

[24] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” Softw Pract Exp, vol. 41, no. 1, pp. 23–50, Aug. 2010, doi: 10.1002/spe.995.