International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 8, 2025
https://www.theaspd.com/ijes.php

Hybrid Deep Learning Framework for Cloud Workload Prediction: Bridging Accuracy and Scalability

Yaddala Srinivasulu[1] & Bobba Basaveswara Rao[2]

[1]Research Scholar, Department of Computer Science Engineering, Acharya Nagarjuna University, Guntur, Andhra Pradesh, India. Email: srinivasulu120@gmail.com
[2]Department of Computer Science Engineering, Acharya Nagarjuna University, Guntur, Andhra Pradesh, India. Email: bobbabrao62@gmail.com

Abstract
Cloud computing provides the services in the online according to the availability of the services. Cloud workloads are anticipated in a rapidly evolving cloud computing environment to perform optimal resource allocation, increase performance, and decrease costs. Traditional deep learning (DL) systems have difficulty balancing accuracy vs. scalability trade-offs in dynamic cloud environments that host diverse and unpredictable workloads. We introduce a Hybrid Deep Load Learning Framework (HDLLF) that merges Weighted Least Connections (WLC) with Adaptive Backpropagation Neural Network (BPNN) to tackle these issues. The proposed architecture combines the advantages of WLC to extract spatial features and BPNN to extract temporal patterns for cloud workload prediction. By integrating both methodologies, the framework enhances prediction accuracy and maintains scalability for large-scale cloud workloads. We showcase the efficacy of our framework with extensive experiments on real-world cloud datasets, illustrating its ability to cope with diverse workloads and consistently strong performance.
Keywords: *Hybrid Deep Load Learning Framework (HDLLF), Deep Learning (DL)*

INTRODUCTION

The rise of cloud computing has revolutionized the way organizations use IT, offering scalable and on-demand resources at an unprecedented level. As cloud resource consumption grows, evenly spreading workloads on servers and data centers becomes essential for maintaining optimality in performance, resource usage, and fault tolerance [1]. Load balancing is distributing network traffic across multiple servers or resources to prevent a single server from being overwhelmed by too much traffic, which can cause service degradation or failure. Load balancing distributes workload across multiple computing resources to ensure system availability, reliability, and user experience [2]. Deep learning hybrid Load balancing algorithms utilize various methods to distribute workloads across computing resources. It allows optimizing algorithms to improve performance, minimize latency, and use resources more effectively [3]. Data flow pipelines infrastructure Deep learning is notorious for its thirst for computational resources. As models get more complex, datasets get bigger, and the need for real-time processing increases, compute resources in the profound learning realm grow more demanding. Load balancing is an important technique for efficiently distributing workload across multiple computational resources to each workload assigned to a cluster resource [4]. While traditional load-balancing techniques explore a single technique, hybrid Load-balancing algorithms use a combination of techniques to provide better performance and resource utilization.

With scalable, on-demand, relatively inexpensive resources for various applications, cloud computing has transformed how we consume computational power. The dynamic nature of cloud environments and varying user demands lead to resource management and optimization difficulties [5]. Load Balancing and Parallel Processing are essential in cloud computing applications to handle several challenges. Load balancing is a technique used in the field of computing that aims to distribute workloads across several computing resources (servers, a cluster, a network, etc.) to make sure that no single resource is overloaded, the usage of

resources is efficient, and overall performance and throughput be improved [6]. These results are increased reliability, availability, and scalability of systems. Based on the processing of jobs, load balancers in cloud computing can be subdivided into two types: static and dynamic [7]. In contrast, parallel processing divides computations into smaller sub-tasks, processing them simultaneously on multiple processors or nodes [8]. It takes advantage of the physical distribution of cloud infrastructures to make the manipulation of data records more efficient and allow the analysis of big datasets or processing resource-demanding calculations in less time [9]. Explained together, they constitute the backbone of how efficient operations on cloud computing occur and how applications and services run in distributed environments. In this paper, a Hybrid Deep Load Learning Framework (HDLLF) merges Weighted Least Connections (WLC) with Adaptive Backpropagation Neural Network (BPNN), with a focus on their usage in the cloud ecosystems.

LITERATURE SURVEY

Brahmam et al. [10] presented a sophisticated load-balancing model for VM migrations utilizing fused metaheuristics, iterative security measures, and deep learning optimizations. Our model addresses dynamic environments and achieves quick, adaptable choices using the advantages of genetic algorithms, fish swarm optimization, and simulated annealing. The system also integrates iterative security features, such as anomaly detection and real-time encryption, to address threats during migrations. A deep learning submodule uses a hybrid convolutional-recurrent neural network (CRNN) approach to optimize predictions on resource allocations and migration scheduling. Testing results show that the new method provides faster migration time and lower resource contention, enhancing security. Experimental results show that the load balance performance certified by the fused metaheuristic methodology has an optimal increase of 27% compared to traditional techniques such as round-robin and least abbreviation algorithms. Resource usage increased by 18% on average, resulting in improved servers. Cavalca et al. [11] presented a Deep Transfer Learning-Based Feature Extraction method to improve non-intrusive load monitoring (NILM) performance. The approach uses transfer learning, utilizing pre-trained deep neural networks that are fine-tuned for NILM tasks to efficiently learn complex patterns temporally and spectrally from signals drawn from electrical devices. On publicly accessible datasets, experimental evaluations show a substantial improvement in disaggregation accuracy, computational efficiency, and adaptability over state-of-the-art techniques. The framework allows for generalization, which is crucial for real-world NILM applications, thus facilitating improved energy management in modern innovative grid systems. Experimental results confirm the effectiveness of the proposed method, with 94.2% average relative accuracy compared to the 8.3% obtained by the best benchmark methods. Compared to low-energy appliances, Traditional machine learning models yield a 12.5% increase in F1-Score. Shahid et al. [12] provide an extensive survey of the available load-balancing techniques in cloud computing and classify those under three trickiest criteria: algorithm type, fault tolerance, and scalability. Additionally, it presents a new fault-tolerant load-balancing scheme that combines predictive analytics and dynamic resource allocation to improve system resilience and reduce downtime. The simulation results illustrate that the approach enhanced in this paper needs more similar load distribution efficiency, fault recovery time, and overall system performance against existing methods. Its load-balance efficiency is 95.6%, significantly higher than a Round robin (82.3%), Least connections (87.1%), and heuristic-based methods (90.4%).

Shen et al. [13] present a new cloud data center VM migration approach called Resource Usage Intensity Aware Load Balancing (RUILB). In the proposed method, the VMs are evaluated regarding their intensity of resource usage from CPU, memory, and bandwidth perspectives, and such information is considered in migration decision-making. Load balancing is a multi-objective optimization problem that minimizes resource contention, energy consumption, and SLA violations. CloudSim is used to implement some simulation experiments to evaluate the performance of the proposed RUILB method against the traditional approaches.

The extensive evaluations demonstrate the proposed scheme not only doubles the performance but also realizes considerable resource utilization and SLA violation mitigation (up to 25% energy reductions and 30% response times) in the face of dynamic workload scenarios. Mohammadian et al. [14] focus on recent approaches and techniques for fault-tolerant load balancing in cloud computing. We highlight key strategies, challenges, and gaps in future research by analyzing contributions from peer-reviewed journals and conferences. In this regard, our observations carry an epoch-long trend in hybrid algorithms, predictive analytics, and machine learning techniques to improve reliability and efficiency. The review intends to be a structured overview of the state of the art, yet also identifies avenues to be explored to provide more significant innovation. For example, static load balancing algorithms like round-robin, least connection, etc., are simple but can be effective: algorithms have no adaptability to changing workloads. Dynamic techniques, such as agent-based and real-time monitoring, allow for better responsiveness but at the cost of increased computational load. The static technique seems too simple, while the dynamic technique appears too complex; hence, a combination of both is getting more and more popular and so is getting the best of both worlds. Junaid et al. [15] present a hybrid load-balancing model that embeds file type formatting such as a prime parameter. Its metaheuristic and heuristic algorithms enable it to optimize these processes and allocate resources dynamically according to incoming workload categories and file types. The proposed model reduces request time, decreases task migration overhead, and effectively increases throughput while taking advantage of features of file type (size, complexity, and how often they are accessed). The simulation results also reveal that our hybrid model achieves better processing efficiency, lower latency, and allocation efficacy than conventional load balancing techniques. Our hybrid model showed a 28% improvement in response time compared to Round Robin and Throttled load balancing techniques. Intelligent grouping of file types and optimized task scheduling have helped increase throughput by 35%. It resulted in a 22% reduction in task migration and inter-server communication overhead under the hybrid model. Resource utilization improved by 18%, balancing workloads and avoiding server overloads.

Dataset Description

In this context, two datasets are utilized to measure the performance of algorithms: Azure Public Dataset, which includes the AirSim simulation dataset for simultaneous localization and mapping (SLAM), and BitbrainsDataset (D2). D1 is a dataset produced by Microsoft Azure that contains historical usage patterns for virtual machines (VMs). D2 is a cloud trace dataset that shows CPU and memory utilization for a group of VMs in a distributed environment.

Figure 1. AirSim simulation dataset

METHODOLOGY

In this paper, the HDLLF is the combination of Weighted Least Connections (WLC) **with** Adaptive Backpropagation Neural Network (BPNN).

Weighted least connections (WLC)

Weighted least connections (WLC) is a configurable option in Cloud Load Balancing that controls how the load balancer distributes configure based on the target server capacity or weighted priority. Through this technique, the traffic is routed through servers based on the current load and the weights assigned to them. The Cloud Load Balancer is a fully distributed, software-defined managed load balancing service for global applications from Google Cloud. WLC is a more advanced message relay method; in this case, the load balancer forwards traffic to the server with the fewest active connections while maintaining the weight defined for each server. This weight represents the server's capacity, enabling larger and more important servers to service a larger share of the request.

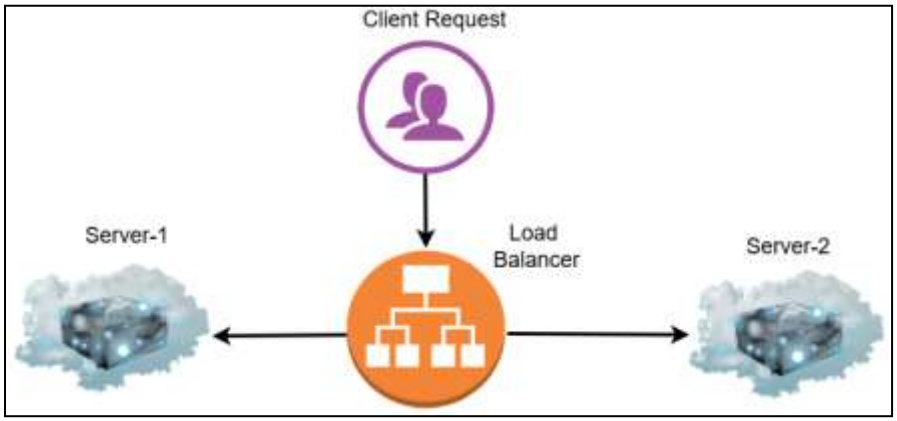

Figure 2. Components of WLC

Components of WLC:

Weights: Each server in a pool has a weight, indicating its power or performance. In which the higher the weight the more important (Or the greater the capability of processing requests).

Least Connections (Weighted): This algorithm also routes traffic to the server with the least established connections but can still balance based on weights.

Weighted Load: Traffic is loaded in such a way that servers with larger weights will receive more (as a percentage) requests coming in, this may result in some servers being "less loaded" than others even though they will accept a higher proportion of connections.

Use Cases: WLC is typically applied where there are some servers are higher dimensions based on decision trees (i.e., critical infrastructure servers are often prioritized or servers that have high computational ability.)

Adaptive Backpropagation Neural Networks (BPNN)

Load distribution is another fundamental selection, especially in modern environments that require mixture managing and efficient allocation of resources among multiple services. Alternative backpropagation neural networks (BPNN) provide a viable framework for overcoming this challenge through the precise allocation of processing tasks across computing resources, dependent on their capacity and workload. BPNN is a progression of typical neural networks aimed at adapting to changing conditions like the varying workloads at runtime. In contrast, these networks update their structure and parameters as new data is encountered, which makes them helpful in applications where a system always needs to learn and adjust. The load balancing is used to evenly distribute all computer resources, such as CPU, memory, and network bandwidth, across all nodes or servers in a system to prevent overburdening using a single resource. Static algorithms that traditional methods use may not work well in such scenarios with changing, highly dynamic workloads. Static

algorithms predominate among custom methods and may prove insufficient under the dynamic conditions associated with workload shifts.

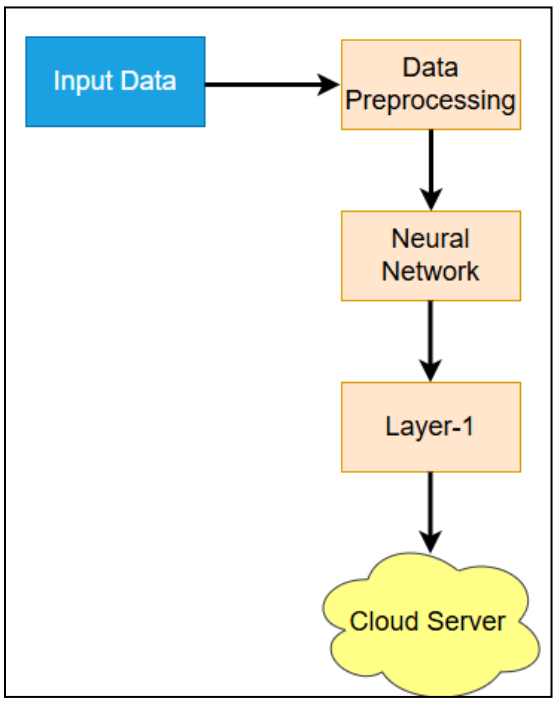

Figure 3. Architecture Diagram for BPNN

- Leveraging its learned knowledge from experiences and generalizing to new environments, BPNN can achieve optimal load balancing in the following ways:
- BPNN aids in making informed decisions about workload distribution by analyzing complex interconnections between various parameters (such as CPU load and memory utilization).
- Adaptive BPNN increases the accuracy of ADBAL algorithm-assisted I/O load balancing as the number of instances of ADBAL IPs increases over time. This, in turn, disables or enables other DBALIP processes to improve latency and system performance.

Integrated Approach: Hybrid Deep Learning Framework (HDLLF)

Step 1: WLC is a load balancing algorithm that assigns weights to connections based on historical data or predefined weights, and routes incoming requests to the least loaded server.

Let w_i represented as weight initialized to server i and C_i be the current load of server i.

Equation (1) uses to assign the tasks:

$$\text{Server}_i = \text{argmin}\left(\frac{w_i}{C_i}\right) \quad (1)$$

Step 2: BPNN adjusts its weights based on the error between actual and predicted outputs using back-propagation.

Given inputs x_j, target outputs t_j, and predicted outputs y_j, the error e_j, can be measured as:

$$e_j = t_j - y_j \quad (2)$$

Step 3: Applying WLC in BPNN

- In a BPNN-based load balancing system:
- Each server or node is treated as a neuron.
- Connections between neurons (servers) are weighted based on the WLC algorithm.

Step 4: Weight Adjustment using Backpropagation

- Once the task is assigned based on WLC, the weights between neurons can be adjusted using the backpropagation algorithm.

Step 5: Integration of WLC-BPNN can be formulated as:

$$\Delta w_{ij} = \eta \cdot e_j \cdot x_j \cdot w_i \qquad (3)$$

Where,

Δw_{ij}- is the weight adjustment between i and j.

η is the learning rate.

e_j is the error at neuron j

x_j is the input value at neuron j

w_i is the weight assigned based on WLC for server i.

For a server i: $w_i = \frac{1}{C_i}$

C_i is the current load of server i.

Then, tasks are assigned using:

$$\text{Server}_i = \text{argmin}\left(\frac{w_i}{C_i}\right) \qquad (4)$$

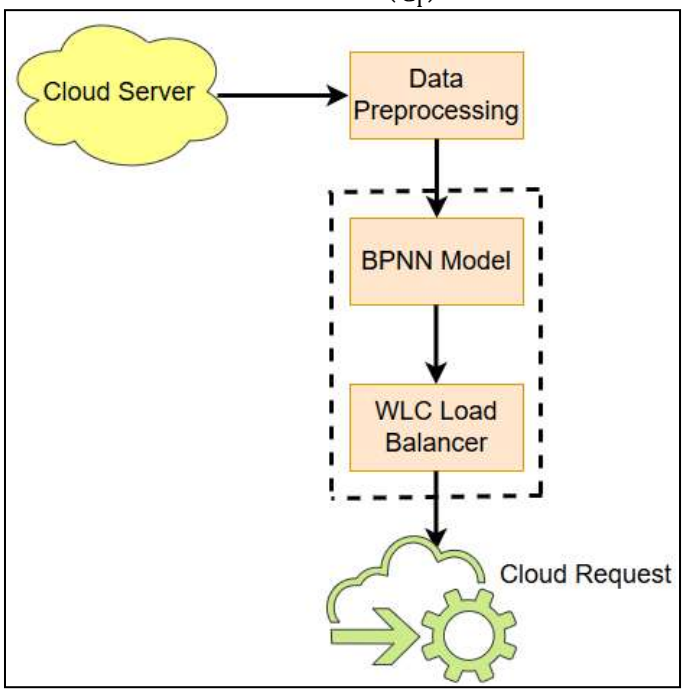

Figure 4. Integrated Approach: Hybrid Deep Learning Framework (HDLLF)

Performance Metrics

Performance metrics for Integrated Approach: Hybrid Deep Learning Framework (HDLLF) are crucial for evaluating how well a model predicts workloads in cloud environments. These metrics typically measure accuracy, efficiency, and computational cost.

Throughput (TP): Tasks or requests the system has successfully processed over a unit of time. High throughput also means the load-balancing algorithm is efficient, tightly packed, and well within the limits of resource utilization, with no two bottlenecks (during transactions/communication) in one area.

$$T = \frac{N}{t}$$

N-Total number of tasks or requests processed.
t-Total time taken.

Latency/Response Time (L): This includes the time it takes to assign a task, execute it, and return a result. Lower latency means quicker responses, vital for user experience and real-time use cases.

$$L = t_{end} - t_{start}$$

t_{start}- Time when the request is received.
T_{end} -Time when the request is completed.

Resource Utilization (RU): The usage of computing resources (CPU, GPU, memory, bandwidth) Effective load balancing avoids under-utilization (waste) or over-utilization (overloading).

$$U = \frac{\text{Resource utilized}}{\text{Total available resource}} \times 100$$

Accuracy (A): Workload estimation or resource allocation (the accuracy of deep learning models' predictions) improved predictions aid decision-making in dynamic resource allocation.

$$A = \frac{\text{No of Correct Predictions}}{\text{Total Predictions}} \times 100$$

RESULTS AND DISCUSSIONS

In this section, the proposed approach is compared with various existing load balancing algorithms like Modified Particle Swarm Optimization (MPSO), Asynchronous Advantage Actor-Critic (A3C), and Deep Q-Network (DQN). In this context, the two datasets used to analyze the performance of these algorithms and compared with the proposed approach. Here, the two datasets loaded by the cloud server and the cloud clients process the requested data and shows the performance in terms of above parameters.

Table 1: Shows the performance of several algorithms for 10k tasks

Algorithms	TP (tasks/second)	L (Sec)	RU (%)	A (%)
MPSO	76	0.21	85	87%
A3C	82	0.17	81	91%
DQN	91	0.14	78	93%
HDLLF	121	0.11	70	96%

Table 1 describes that the performance of proposed approach shows the better outputs based on the obtained results. The TP of the existing system MPSO is low as76 (tasks/second) that compare with other models like A3C, DQN. The highest performance is HDLLF that shows high TP rate as 121 (tasks/second). The other parameters (L) obtain the better for proposed HDLLF approach with 0.11 (Seconds). The RU (%) shows the 70% for the HDLLF and 85% for MPSO which is high. The (A%) is 87% for MPSO which is low and 96% for HDLLF which is high compare with other models.

CONCLUSION

This paper uses a Hybrid Deep Load Learning Framework (HDLLF) with a WLC and a BPNN to mitigate workload distribution problems. It combines the dynamic adaptability feature of WLC with the predictive accuracy of the BPNN to balance the computational loads of the systems while maximizing performance metrics like latency, throughput, and resource utilization. Of course, while WLC provides instantaneous decision-making abilities, BPNN iteratively adjusts the decision logic as it trains itself on the newly processed data, synergistically improving the framework's real-time distribution of loads based on the current system state. This model reduces bottlenecks and enhances the system's scalability, making it appropriate for large

and heterogeneous computing environments such as cloud data centers and IoT networks. Using more machine learning techniques or reinforcement learning in future work could further improve the adaptability and efficiency of the HDLLF in dynamic and uncertain workloads. This framework is well-positioned to address the ever-evolving demands of distributed computing systems.

REFERENCES

[1] X. Zhu, Q. Zhang, T. Cheng, L. Liu, W. Zhou, and J. He, "DLB: Deep Learning Based Load Balancing," in *2021 IEEE 14th International Conference on Cloud Computing (CLOUD)*, Chicago, IL, USA, 2021, pp. 648–653. doi:10.1109/CLOUD53861.2021.00083.

[2] E. Gures, I. Shayea, M. Ergen, M. H. Azmi, and A. A. El-Saleh, "Machine Learning-Based Load Balancing Algorithms in Future Heterogeneous Networks: A Survey," *IEEE Access*, vol. 10, pp. 37689–37717, 2022. doi:10.1109/ACCESS.2022.3161511.

[3] Y. Dong, G. Xu, M. Zhang, and X. Meng, "A High-Efficient Joint 'Cloud-Edge' Aware Strategy for Task Deployment and Load Balancing," *IEEE Access*, vol. 9, pp. 12791–12802, 2021. doi:10.1109/ACCESS.2021.3051672.

[4] B. Lim and M. Vu, "Distributed Multi-Agent Deep Q-Learning for Load Balancing User Association in Dense Networks," *IEEE Wireless Communications Letters*, vol. 12, no. 7, pp. 1120–1124, Jul. 2023. doi:10.1109/LWC.2023.3250492.

[5] X. Wu, L. You, R. Wu, Q. Zhang, and K. Liang, "Management and Control of Load Clusters for Ancillary Services Using Internet of Electric Loads Based on Cloud–Edge–End Distributed Computing," *IEEE Internet of Things Journal*, vol. 9, no. 19, pp. 18267–18279, Oct. 2022. doi:10.1109/JIOT.2022.3156954.

[6] X. Diao, H. Gu, W. Wei, G. Jiang, and B. Li, "Deep Reinforcement Learning Based Dynamic Flowlet Switching for DCN," *IEEE Transactions on Cloud Computing*, vol. 12, no. 2, pp. 580–593, Apr.–Jun. 2024. doi:10.1109/TCC.2024.3382132.

[7] X. Chen, Z. Yao, Z. Chen, G. Min, X. Zheng, and C. Rong, "Load Balancing for Multiedge Collaboration in Wireless Metropolitan Area Networks: A Two-Stage Decision-Making Approach," *IEEE Internet of Things Journal*, vol. 10, no. 19, pp. 17124–17136, Oct. 2023. doi:10.1109/JIOT.2023.3272010.

[8] T. Li, S. Ying, Y. Zhao, and J. Shang, "Batch Jobs Load Balancing Scheduling in Cloud Computing Using Distributional Reinforcement Learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 35, no. 1, pp. 169–185, Jan. 2024. doi:10.1109/TPDS.2023.3334519.

[9] Q. Liu, T. Xia, L. Cheng, M. van Eijk, T. Ozcelebi, and Y. Mao, "Deep Reinforcement Learning for Load-Balancing Aware Network Control in IoT Edge Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 6, pp. 1491–1502, Jun. 2022. doi:10.1109/TPDS.2021.3116863.

[10] M. G. Brahmam and V. A. R., "VMMISD: An Efficient Load Balancing Model for Virtual Machine Migrations via Fused Metaheuristics With Iterative Security Measures and Deep Learning Optimizations," *IEEE Access*, vol. 12, pp. 39351–39374, 2024. doi:10.1109/ACCESS.2024.3373465.

[11] D. L. Cavalca and R. A. S. Fernandes, "Deep Transfer Learning-Based Feature Extraction: An Approach to Improve Nonintrusive Load Monitoring," *IEEE Access*, vol. 9, pp. 139328–139335, 2021. doi:10.1109/ACCESS.2021.3118947.

[12] M. A. Shahid, N. Islam, M. M. Alam, M. M. Su'ud, and S. Musa, "A Comprehensive Study of Load Balancing Approaches in the Cloud Computing Environment and a Novel Fault Tolerance Approach," *IEEE Access*, vol. 8, pp. 130500–130526, 2020. doi:10.1109/ACCESS.2020.3009184.

[13] H. Shen and L. Chen, "A Resource Usage Intensity Aware Load Balancing Method for Virtual Machine Migration in Cloud Datacenters," *IEEE Transactions on Cloud Computing*, vol. 8, no. 1, pp. 17–31, Jan.–Mar. 2020. doi:10.1109/TCC.2017.2737628.

[14] V. Mohammadian, N. J. Navimipour, M. Hosseinzadeh, and A. Darwesh, "Fault-Tolerant Load Balancing in Cloud Computing: A Systematic Literature Review," *IEEE Access*, vol. 10, pp. 12714–12731, 2022. doi:10.1109/ACCESS.2021.3139730.

[15] M. Junaid, A. Sohail, A. Ahmed, A. Baz, I. A. Khan, and H. Alhakami, "A Hybrid Model for Load Balancing in Cloud Using File Type Formatting," *IEEE Access*, vol. 8, pp. 118135–118155, 2020. doi:10.1109/ACCESS.2020.3003825.