Journal of Cognitive Human-Computer Interaction

♦ ISSN: 2771-1463 Vol: 09, No. 02, 2025 ♦ PP. 32–36

Article DOI: https://doi.org/10.54216/JCHCI.090207

FULL LENGTH ARTICLE

An Intelligent Framework for Flavor Recommendation and Cost Optimization in Hybrid Cloud Autoscaling

Agnes Osagie[1,*] **Sandra Terazic**[2] **Barbara Charchekhandra**[3]

[1] Cape Peninsula University of Technology, Faculty of Applied Science, South Africa
[2] Department of Mathematics, University of Rijeka, City of Rijeka, Croatia
[3] Jadavpur University, Department of Mathematics, Kolkata, India

Emails: Osagieagne2000@cput.ac.za · Sandy1997te@Uniri.hr · Charchekhandrabar32@yahoo.com

Received: January 27, 2025 Revised: February 25, 2025 Accepted: April 25, 2025 ⋆ Corresponding author

ABSTRACT

This research presents a flavor recommendation framework intended for hybrid clouds to address resource provisioning and cost issues. A cloud flavor is an instance type that assigns values for CPU, memory, storage, and networking. At present, flavor selection is often manual and static, which makes the process inefficient when workloads change and some flavors remain underutilized. The proposed framework provides flavor recommendations for automated dynamic capacity provisioning by using predictive analysis of workload and cost across different cloud service providers. It uses an RNN–LSTM-based Proactive Predictive Engine (PPE) to estimate future resource requirements and a Recommendation Engine composed of scoring and flavor engines. The framework receives actual and predicted CPU and memory consumption, cost fluctuations, and provider options, then selects suitable flavors at runtime. Metrics are gathered, stored, and analyzed in real time through Telegraf, InfluxDB, and Apache Libcloud. Experimental results on AWS and OpenStack show that the proposed framework reduces the number of EBS volumes and VMs by 19% and achieves cost savings of up to 17% compared with traditional and reactive approaches. This solution transforms static resource allocation into real-time predictive optimization of resources and expenses in hybrid cloud environments.

Keywords: Autoscaling ▪ Flavor recommendation ▪ Hybrid cloud ▪ RNN–LSTM ▪ Cost optimization ▪ Resource forecasting ▪ Dynamic resource allocation

1. INTRODUCTION

Cloud computing service delivery has drastically changed how companies and organizations adapt and deploy computational assets. As organizations need more flexible infrastructure, the hybrid cloud has emerged as a transition between public and private clouds. This approach helps organizations benefit from both environments while meeting cost, resource, and data-protection requirements [1]. In this context, flavors—configurations of computing elements including CPU, memory, storage, and networking—are central to matching infrastructure capabilities with application demands.

Cloud service providers (CSPs) provide a wide variety of flavors designed for CPU-bound, memory-bound, or network-bound applications [2]. However, different CSPs do not follow a uniform standard for defining flavor configurations. Selecting an appropriate flavor therefore requires navigating multiple criteria, heterogeneous vendor offerings, dynamic workloads, and pricing models. Manual flavor selection is tedious and inefficient, especially in hybrid cloud environments where resource demand changes over time [3].

Proper selection of flavors determines resource utilization and operating cost. Applications in hybrid clouds typically experience variable computing demands that cannot always be predetermined. Traditional selection methods often rely on current workload or predefined configurations; once selected, a flavor is repeatedly used even if demand changes. This static approach increases operational cost and weakens one of the main advantages of cloud systems: elastic scalability.

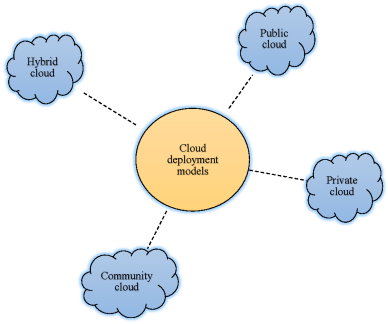

Figure 1. Deployed cloud models.

To overcome these issues, modern cloud systems increasingly use advanced resource-allocation and cost-optimization techniques. Workload prediction and forecasting estimate future requirements so that provisioning decisions can be made before resource shortages or overprovisioning occur [4, 5]. Machine learning and deep learning approaches, including Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, are particularly useful for time-series workload data because they model temporal dependencies and changing resource patterns [6, 7].

Cost reduction is also essential in flavor selection. Cloud pricing varies according to provider, number of VMs, flavor types, and storage or network services. A predictive system can compare expected workload requirements with current provider offerings to select a cost-efficient configuration before demand spikes occur. The need for automated intelligent systems is therefore clear: when workload forecasting and cost modeling are combined, hybrid cloud resource management can become more accurate, scalable, and economical.

2. RELATED WORK

A literature survey is an important foundation for academic research because it helps establish existing knowledge, identify trends, and locate research gaps. In cloud resource management, previous research has examined autoscaling, task scheduling, edge-cloud resource allocation, and intelligent provisioning methods [8, 9, 10]. These studies show the importance of accurate workload analysis when applications are distributed across heterogeneous infrastructure.

Traditional cloud autoscaling methods often use reactive thresholds. They add or remove resources after CPU, RAM, or request-load metrics exceed predefined limits. Although threshold-based autoscaling is simple, it can produce delayed responses, overprovisioning, or underprovisioning when workloads change abruptly. Prediction-based scheduling has been proposed to address this weakness by forecasting future demand and preparing resources earlier [3, 11].

Machine learning has become a common technique for resource forecasting. RNN and LSTM models are suitable for this task because cloud workloads are time-dependent. Prior studies on machine learning for big data analytics, IoT, financial trends, and predictive maintenance show that these methods can capture nonlinear patterns and temporal dependencies [9, 7, 12]. In hybrid clouds, however, forecasting alone is not sufficient; the predicted workload must also be translated into concrete flavor recommendations across multiple CSPs.

Other work has focused on cloud-edge computing, data-intensive application scheduling, and sustainable offloading [13, 14, 15]. These methods improve workload placement, but many do not directly address flavor heterogeneity and price variation across public and private clouds. The present research contributes by integrating workload prediction with a recommendation engine that scores resource requirements and selects cost-efficient flavors dynamically.

3. OBJECTIVES OF THE RESEARCH

Resource orchestration in hybrid-cloud host environments is central to improving performance and reducing cost. This research addresses flawed, passive, and time-consuming flavor selection by developing a dynamic predictive model. The main objectives are:

- Design a low-cost architecture for recommending flavors to improve resource management in hybrid-cloud platforms.
- Embed workload prediction by applying RNN–LSTM to forecast application resource needs accurately.
- Compare and evaluate quantitative and qualitative improvements in cost and performance against conventional flavor-selection approaches.

4. INSPIRATION FOR RESEARCH

Resource provisioning in increasingly complex hybrid cloud environments requires intelligent techniques. Reusable, static, and manual flavor-selection methods are inefficient, waste resources, and increase expenses. This work is inspired by the need for an automated solution that integrates workload forecasting and cost modeling to recommend suitable flavors for scalable, high-performing, and cost-effective cloud systems in dynamic environments.

5. DEVELOPED METHODOLOGY

The proposed flavor recommendation system suggests cost-efficient flavors for anticipated workloads to support autoscaling. The system has two main components: the Proactive Predictive Engine (PPE) and the Recommendation Engine. The Recommendation Engine includes the Scoring Engine and the Flavor Engine. Using RNN–LSTM, the PPE predicts future CPU and RAM needs. The Scoring Engine calculates required resources from actual and predicted consumption, while the Flavor Engine determines the most suitable flavor among registered public and private cloud providers.

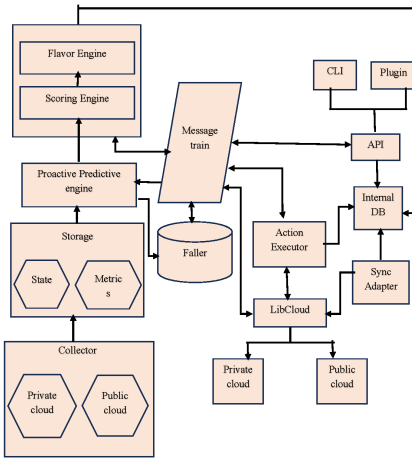

Figure 2. Illustration of the Flavor Recommendation Framework.

5.1 Framework Components

The Metric Collector gathers real-time CPU, RAM, and related usage metrics from virtual machines. It uses tools such as Telegraf and plugin-based collectors to obtain performance measurements. These metrics are passed to the Metric Storage component, where time-series platforms such as InfluxDB store historical workload data for querying and prediction.

The PPE is the forecasting module. It uses RNN–LSTM models to predict future CPU and RAM requirements by analyzing metrics stored in the repository. The predicted values are sent to the Scoring Engine for resource calculation. A Message Queue, such as RabbitMQ, enables immediate communication among framework modules, schedules tasks, and improves scalability and fault tolerance.

The API provides REST-based services for external applications and users. Through a command-line tool or graphical interface, users can manage, monitor, and control the flavor recommendation system. The Action Executor implements Recommendation Engine decisions by provisioning resources or changing configurations through the internal database and other modules. Apache Libcloud provides a unified interface for interacting with multiple cloud providers, fetching flavor details, VM specifications, and provider configurations.

The InternalDB stores available cloud-provider resources, user templates, and flavor history. A synchronization adapter updates flavor information, costs, and specifications from public and private clouds so that recommendations are based on current provider data.

5.2 Proactive Predictive Engine

The PPE predicts workload behavior from historical time-series metrics. Let X_t denote the input metric vector at time t, including CPU and RAM usage. The LSTM updates can be expressed as:

$$f_t = \sigma(W_f[h_{t-1}, X_t] + b_f), \tag{1}$$

$$i_t = \sigma(W_i[h_{t-1}, X_t] + b_i), \tag{2}$$

$$\tilde{C}_t = \tanh(W_c[h_{t-1}, X_t] + b_c), \tag{3}$$

$$C_t = f_t C_{t-1} + i_t \tilde{C}_t, \tag{4}$$

$$o_t = \sigma(W_o[h_{t-1}, X_t] + b_o), \tag{5}$$

$$h_t = o_t \tanh(C_t). \tag{6}$$

The predicted resource vector is then produced as:

$$\hat{Y}_{t+1} = W_y h_t + b_y, \tag{7}$$

where $\hat{Y}_{t+1}$ includes expected CPU and RAM utilization for the next time interval.

5.3 Scoring Engine

The Scoring Engine converts predicted utilization into required infrastructure resources. If CPU_p and RAM_p are predicted usage values and CPU_f and RAM_f represent the CPU and RAM capacity of a candidate flavor, the required VM count may be estimated as:

$$VM_{req} = \max\left(\left\lceil \frac{CPU_p}{CPU_f} \right\rceil, \left\lceil \frac{RAM_p}{RAM_f} \right\rceil\right). \tag{8}$$

A utilization score combines CPU and RAM needs:

$$Score = w_c \frac{CPU_p}{CPU_f} + w_r \frac{RAM_p}{RAM_f}, \tag{9}$$

where w_c and w_r are weighting factors.

5.4 Flavor Engine

The Flavor Engine identifies the least-cost flavor that satisfies predicted requirements. The cost of provisioning a candidate flavor is:

$$Cost_f = VM_{req} \times Price_f. \tag{10}$$

The optimal flavor is selected as:

$$f^* = \arg\min_{f \in F} \{Cost_f : CPU_f \geq CPU_{req},\ RAM_f \geq RAM_{req}\}. \tag{11}$$

The engine stores the selected option in the recommendation table and returns its name, vCPU capacity, RAM capacity, and cost.

5.5 Optimized Recommendation Algorithm

The optimized recommendation algorithm operates in the following stages:

1. Obtain expected CPU and RAM usage from the PPE.
2. Calculate required vCPU and RAM resources.
3. Retrieve available flavors and costs from the InternalDB.
4. For each cloud provider, calculate the required number of VMs and total provisioning cost.
5. Store the best option in the Recommend_Flavor table.
6. Return the optimal flavor for resource provisioning.

6. RESULT

The effectiveness of the PPE was assessed on a real-time web-based application using CPU and RAM utilization at 10-minute intervals. Accuracy measures the extent to which the prediction model provides the correct number of resources. Precision measures how many predicted provisioning events are valid, while recall measures the model's ability to avoid underprovisioning.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}, \quad (12)$$

$$Precision = \frac{TP}{TP+FP}, \quad (13)$$

$$Recall = \frac{TP}{TP+FN}. \quad (14)$$

Table 1. PPE forecasted results of CPU and RAM utilization.

Time (Minutes)	CPU Utilization (%)	RAM Utilization (%)
13:00	30	40
13:10	35	45
13:20	40	50
13:30	60	40
13:40	90	35
13:50	80	50
14:00	85	60
14:10	95	45
14:20	85	50
14:30	70	55

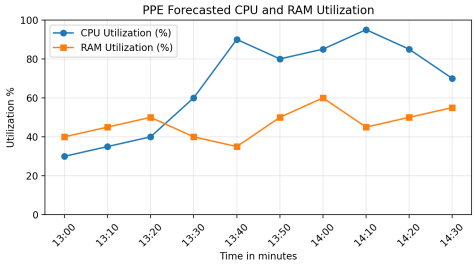

Figure 3. Visual representation of PPE forecasted CPU and RAM utilization.

The PPE measures average CPU and RAM usage for web, application, and database servers at 10-minute intervals. This interval helps account for VM provisioning time, avoids unrealistic reactions to short spikes, and provides a buffer for unexpected surges. With one week of workload history, the RNN–LSTM model predicts CPU and RAM consumption and feeds the predicted outcomes into the Scoring Engine.

Table 2. Scoring Engine forecasted results of CPU and RAM utilization.

Time (Minutes)	CPU Utilization (%)	RAM Utilization (%)
13:00	30	40
13:10	40	35
13:20	50	30
13:30	45	50
13:40	35	45
13:50	50	30
14:00	40	35
14:10	30	40
14:20	50	30
14:30	45	35

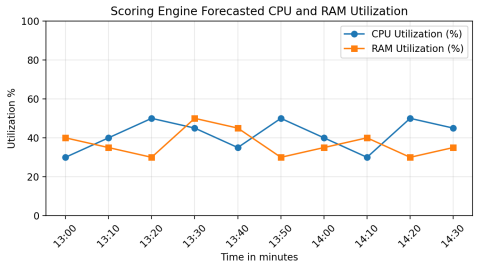

Figure 4. Visual representation of Scoring Engine forecasted CPU and RAM utilization.

The Scoring Engine uses CPU and RAM utilization data to determine minimum and maximum vCPU and RAM thresholds for autoscaling actions. Fluctuations such as a CPU peak of 50% at 13:20 indicate changing resource demand. By integrating real-time data and prediction, the engine allocates resources dynamically and reduces underloading or overloading.

Table 3. Review of total virtual machines allotted by the reactive method and the proposed method.

Time (Minutes)	Reactive Approach (VMs)	Proposed Prediction Approach (VMs)
13:00	10	10
13:10	10	10
13:20	12	10
13:30	12	10
13:40	14	12
13:50	14	12
14:00	12	12
14:10	12	12
14:20	14	12
14:30	12	12

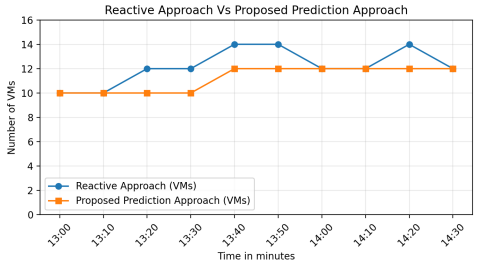

Figure 5. Visual representation of the Reactive Approach versus the Proposed Prediction Approach.

The reactive approach adjusts VMs according to real-time fluctuations and often overprovisions resources, such as 14 VMs at 13:40 and 14:20. The proposed prediction approach provisions a steadier range of 10–12 VMs by anticipating workload requirements. This avoids unnecessary scaling while maintaining enough capacity for the workload.

Table 4. Comparison of performance with existing and proposed models.

Model	Acc %	Pre %	Rec %
Traditional model	85.7	80.5	75.8
Proposed model	96.1	93.2	95.8

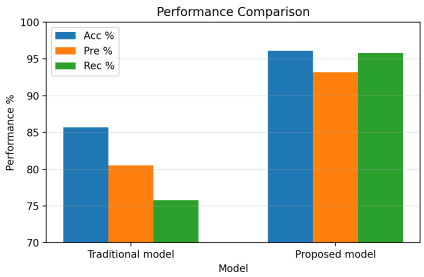

Figure 6. Visual representation of performance comparison.

The proposed model achieves higher accuracy, precision, and recall than the traditional model. It improves resource utilization, reduces cost, and provides dependable system availability through reliable prediction and optimized provisioning plans.

7. CONCLUSION

This research proposes a flavor recommendation framework for improving resource use in hybrid clouds. Static and manual flavor selection is ineffective because resources can be underutilized or overprovisioned, increasing operational cost. The proposed approach addresses these challenges by combining workload prediction through RNN–LSTM models with automated recommendation of the best flavor.

The framework predicts future resource usage from historical usage data and dynamically aligns cloud resources with application needs. Experimental results show a 17% lower cost compared with traditional techniques and 19% fewer virtual machines. These results demonstrate that predictive analytics and automation are promising tools for improving hybrid cloud resource management.

Future work can extend the framework by integrating reinforcement learning to enhance real-time resource-allocation decisions. The work can also be expanded to support multi-cloud environments, additional workload predictions such as bandwidth and storage, and real-time cloud-provider pricing to improve cost efficiency and flexibility.

REFERENCES

[1] S. K. Sarma, "Metaheuristic based auto-scaling for microservices in cloud environment: a new container-aware application scheduling," *International Journal of Pervasive Computing and Communications*, vol. 19, no. 1, pp. 74–76, 2023.

[2] U. Gupta *et al.*, "Deeprecsys: A system for optimizing end-to-end at-scale neural recommendation inference," in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, 2020, pp. 982–995.

[3] G. Kaur and A. Bala, "Prediction-based task scheduling approach for floodplain application in cloud environment," *Computing*, vol. 103, no. 5, pp. 895–916, 2021.

[4] A. Smith and B. Johnson, "A survey of cloud computing architectures for big data applications," *Journal of Cloud Computing: Advances, Systems and Applications*, vol. 11, no. 1, pp. 1–15, 2022.

[5] L. Chen *et al.*, "Enhancing security in cloud computing through advanced encryption techniques," *International Journal of Information Security*, vol. 20, no. 3, pp. 225–239, 2021.

[6] M. R. Patel *et al.*, "Iot-based smart healthcare system: A review," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 4, pp. 4287–4299, 2021.

[7] A. Y. Krishna *et al.*, "Machine learning techniques for predicting financial market trends," *Financial Engineering and Risk Management*, vol. 10, no. 2, pp. 112–130, 2023.

[8] J. Doe and R. Smith, "Blockchain technology in supply chain management: A review," *Journal of Supply Chain Management*, vol. 15, no. 2, pp. 45–60, 2022.

[9] H. Lee *et al.*, "A survey on machine learning for big data analytics," *Big Data Research*, vol. 25, pp. 100–110, 2023.

[10] T. Nguyen *et al.*, "Edge computing for iot applications: A comprehensive survey," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3456–3475, 2021.

[11] A. Kumar and S. Mehta, "Data mining techniques for healthcare: A survey," *Journal of Healthcare Engineering*, vol. 2022, p. 123456, 2022.

[12] T. Zhang and Y. Wang, "A review of machine learning techniques for predictive maintenance," *Journal of Manufacturing Systems*, vol. 60, pp. 101–115, 2023.

[13] M. Alkhalaileh *et al.*, "Data-intensive application scheduling on mobile edge cloud computing," *Journal of Network and Computer Applications*, vol. 167, p. 102735, 2020.

[14] Y. Huang *et al.*, "Task scheduling with optimized transmission time in collaborative cloud-edge learning," in *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, 2021, pp. 1–9.

[15] A. Boukerche *et al.*, "Sustainable offloading in mobile cloud computing: Algorithmic design and implementation," *ACM Computing Surveys*, vol. 52, no. 1, pp. 1–37, 2019.