Estimating Cloud Performance Metrics Using Deep Learning

Surbhi Jhariya[1], Prof. Pawan Panchole[2]
Department of CSE, VITM, Indore, India[1,2]

***Abstract*— Cloud computing has become the backbone of modern digital services, supporting applications that demand high availability, scalability, and performance. As cloud infrastructures grow in complexity, accurately estimating cloud performance metrics—such as response time, throughput, latency, resource utilization, and availability—has become a critical challenge. Traditional analytical and rule-based models often struggle to capture the dynamic, non-linear behavior of cloud environments. In this context, deep learning (DL) has emerged as a powerful data-driven approach for modeling and predicting cloud performance with higher accuracy and adaptability. A cloud environment is inherently dynamic due to factors such as fluctuating workloads, heterogeneous virtual machines, multi-tenancy, and varying network conditions. Performance metrics are influenced by complex interactions between compute, storage, and network resources. This paper presents a deep learning model for estimating cloud performance metrics. It can be observed that the proposed work attains improved performance compared to existing work in the domain.**

***Keywords*— Cloud Computing, Performance Estimation, service-level agreements (SLAs). Regression Learning, Deep Learning, Forecasting Accuracy.**

I. INTRODUCTION

Performance metrics for cloud platforms are influenced by complex interactions between compute, storage, and network resources. Traditional methods, including queuing theory and regression-based models, rely on simplifying assumptions that often fail under real-world workloads. Deep learning models, on the other hand, can learn complex, non-linear relationships directly from large volumes of monitoring data collected from cloud platforms, making them well-suited for performance estimation tasks [1].

II. PERFORMANCE ESTIMATION

A cloud environment is inherently dynamic due to factors such as fluctuating workloads, heterogeneous virtual machines, multi-tenancy, and varying network conditions. Performance metrics are influenced by complex interactions between compute, storage, and network resources. Traditional methods, including queuing theory and regression-based models, rely on simplifying assumptions that often fail under real-world workloads. Deep learning models, on the other hand, can learn complex, non-linear relationships directly from large volumes of monitoring data collected from cloud platforms, making them well-suited for performance estimation tasks [2].

One of the key advantages of using deep learning for cloud performance estimation is its ability to adapt to workload variability. Cloud workloads often exhibit bursty, seasonal, or unpredictable behavior. Deep learning models trained on historical and real-time data can generalize across diverse workload patterns and provide accurate predictions even under sudden workload changes. This capability is especially important for proactive resource management, where performance estimation is used to trigger auto-scaling, load balancing, or migration decisions before service-level agreements (SLAs) are violated [3]

Deep learning-based performance estimation also supports intelligent cloud optimization and decision-making. Accurate predictions of response time or resource utilization enable cloud providers to optimize virtual machine placement, reduce energy consumption, and improve overall system efficiency. For cloud users, performance-aware scheduling and cost-performance trade-off analysis become feasible when reliable performance estimates are available. Moreover, integrating deep learning models with reinforcement learning frameworks allows cloud systems to continuously learn optimal control policies based on predicted performance outcomes [4].

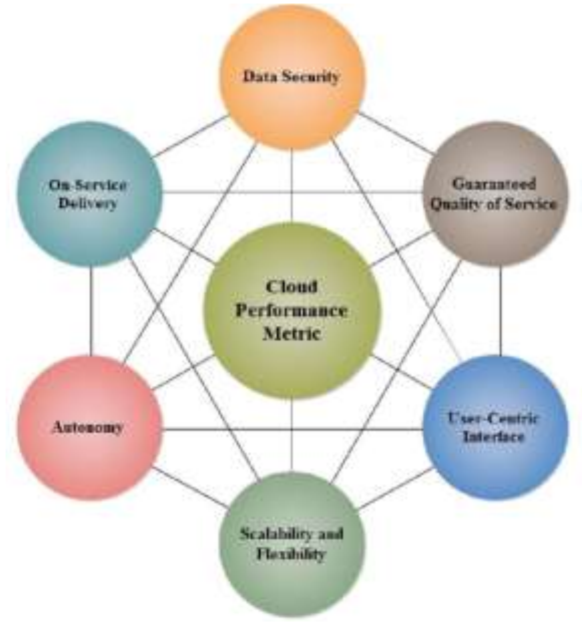


Fig.1 Cloud Performance Metrics

Deep learning-based estimation typically begins with data collection from cloud monitoring tools. Metrics such as CPU usage, memory consumption, disk I/O, network bandwidth, task arrival rates, and historical response times are continuously logged. This multivariate time-series data serves as input to deep learning models [5]. Before training, data preprocessing steps—such as normalization, noise filtering, missing value handling, and feature selection—are applied to improve model robustness and convergence. The quality and granularity of monitoring data play a crucial role in determining the accuracy of performance estimation [6].

III. EMPLOYING DEEP LEARNING MODELS FOR PERFORMANCE ESTIMATION

Different deep learning architectures are employed depending on the nature of the performance metric and data characteristics. Feedforward deep neural networks (DNNs) are commonly used for static or short-term performance estimation, where current resource states are mapped to expected performance outcomes. Convolutional neural networks (CNNs) can extract spatial correlations among multiple virtual machines or containers, especially in large-scale data centers. Recurrent neural networks (RNNs), particularly long short-term memory (LSTM) and gated recurrent unit (GRU) models, are widely used for capturing temporal dependencies in workload patterns and predicting future performance metrics such as latency and throughput [7].

One of the key advantages of using deep learning for cloud performance estimation is its ability to adapt to workload variability. Cloud workloads often exhibit bursty, seasonal, or unpredictable behavior. Deep learning models trained on historical and real-time data can generalize across diverse workload patterns and provide accurate predictions even under sudden workload changes. This capability is especially important for proactive resource management, where performance estimation is used to trigger auto-scaling, load balancing, or migration decisions before service-level agreements (SLAs) are violated [8].

Deep learning-based performance estimation also supports intelligent cloud optimization and decision-making [9]. Accurate predictions of response time or resource utilization enable cloud providers to optimize virtual machine placement, reduce energy consumption, and improve overall system efficiency. For cloud users, performance-aware scheduling and cost-performance trade-off analysis become feasible when reliable performance estimates are available. Moreover, integrating deep learning models with reinforcement learning frameworks allows cloud systems to continuously learn optimal control policies based on predicted performance outcomes [10].

Despite its advantages, estimating cloud performance using deep learning also faces several challenges. Training deep models requires large volumes of high-quality data, which may not always be available due to privacy, security, or monitoring overhead constraints [11]. Model interpretability is another concern, as deep learning models often act as black boxes, making it difficult for cloud operators to understand the reasons behind specific predictions. Additionally, model retraining and deployment in rapidly changing cloud environments introduce computational and operational overheads [12].

IV. METHODOLOGY

In the proposed approach the neural network model is used. In this approach the Bayesian Momentum Based optimization is used to train the neural network model. Predicting these metrics accurately is essential for autoscaling, SLA management, and resource allocation [13]. Traditional regression techniques often struggle with the stochastic and dynamic nature of cloud workloads, especially when data exhibits noise, sudden bursts, or nonlinear dependencies [14].

A momentum-based approach provides an efficient and probabilistically grounded solution, improving both prediction accuracy and uncertainty estimation for cloud performance analysis. At the core of this method is Bayesian regression, which models cloud performance metrics as random variables with prior probability distributions [15]. Instead of producing fixed parameter estimates, Bayesian regression generates posterior distributions that represent uncertainty in the model. This is particularly valuable in cloud environments where workload demand fluctuates unpredictably. The Bayesian framework allows the incorporation of prior knowledge—such as historical performance trends or resource behavior—which refines predictions and improves robustness against noisy or incomplete data. The weights of the network are updated such that the condition for maximization is satisfied of a new sample bearing a conditional probability defined as [22]:

$$\boldsymbol{P}\left(\frac{\boldsymbol{X}}{\boldsymbol{X}_{\boldsymbol{i},}\boldsymbol{k}_1,\boldsymbol{k}_2,\boldsymbol{M}}\right)=\frac{\boldsymbol{P}\left(\frac{\boldsymbol{X}_i}{\boldsymbol{X},\boldsymbol{k}_2,\boldsymbol{M}}\right)\boldsymbol{P}\left(\frac{\boldsymbol{X}_i}{\boldsymbol{k}_1,\boldsymbol{M}}\right)}{\boldsymbol{P}(\frac{\boldsymbol{X}}{\boldsymbol{k}_1,\boldsymbol{k}_2,\boldsymbol{M}})}$$

Here,
P denotes the probability of occurrence of an event.
X_i denotes the vector corresponding to the bias and weight values of the network.
X denotes the training data set

The training rule for the approach is based on the Bayes theorem of conditional probability which is effective training, based on a penalty $\boldsymbol{\rho}=\frac{\boldsymbol{\mu}}{\boldsymbol{v}}$. The weights are updated based on the modified regularized cost function:

$$\boldsymbol{F}(\boldsymbol{w})=\boldsymbol{\mu}\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{w}+\boldsymbol{v}[\tfrac{\mathbf{1}}{\boldsymbol{n}}\textstyle\sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{n}}(\boldsymbol{p}_{\boldsymbol{i}}-\boldsymbol{a}_{\boldsymbol{i}})^{\mathbf{2}}]$$

If $(\pi \ll v)$: Network error are generally low.

else if $(\pi \geq v)$: Network errors tend to increase, in which case the weight magnitude should be reduced so as to limit errors (Penalty).

To optimize the regression parameters efficiently, the approach incorporates a momentum-based gradient update mechanism. In traditional gradient descent, parameter updates can oscillate when the loss surface is irregular, as is common in high-dimensional cloud performance data. Momentum reduces these oscillations by accumulating a moving average of past gradients, enabling smoother and faster convergence. When combined with Bayesian inference, momentum helps explore the posterior distribution more effectively, reducing computational overhead while preserving probabilistic accuracy. The Bayesian momentum-based algorithm typically uses stochastic gradient updates to approximate the posterior, similar to methods like Stochastic Gradient Langevin Dynamics (SGLD). However, adding momentum enhances sampling efficiency, especially when dealing with complex likelihood surfaces common in multi-metric cloud datasets [16]. The momentum term boosts the parameter updates in consistent gradient directions and dampens randomness caused by noisy performance measurements. This ability to stabilize posterior sampling makes the approach well-suited for real-time cloud performance prediction [17]. A key advantage of this technique is the ability to quantify prediction uncertainty, which is crucial for cloud resource management. Performance predictions are not just point estimates but distributions, allowing system controllers to assess the probability of SLA violations or resource saturation. For example, if the posterior variance for predicted CPU utilization is high, the autoscaler may proactively allocate additional VMs to prevent overload. This uncertainty-aware decision-making leads to more reliable and risk-sensitive cloud operations [18].

Algorithm:

```
Start
{
Step.1 Identify benchmark datasets with cloud attributes.

Step.2 Demarcate independent and dependent variables.

Step.3 Split data into training and testing samples, and apply data filtration.

Step.4 Initialize weights randomly and start training.

Step.5 Update weights as:

        $\mathbf{w_{k+1}=w_k-\left[J_kJ_k^T+\mu I\right]^{-1}J_k^Te_k}$
Step.6 Check for condition:
        $(\mathrm{Iterations}\ ==$
        $\mathrm{Max.iterations\ or\ cost\ function\ stabilizes})$
        {
        Truncate Training
        }
        else
        {
        Continue retraining as per step 5


Step.7 Compute performance  metrics.
}
Stop.
```

The next section presents the experimental results obtained.

V. EXPERIMENTAL RESULTS

The simulations are carried out on MATLAB.

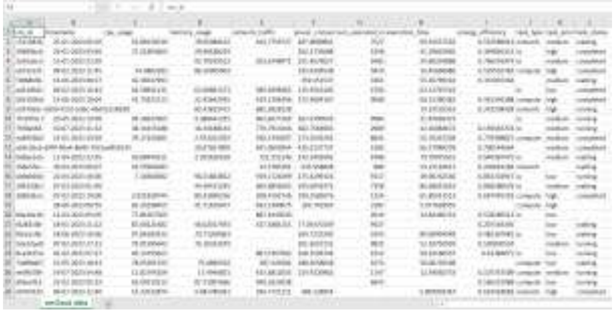
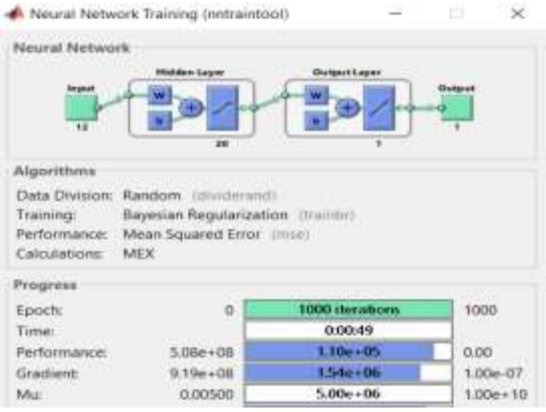


Fig. 2. Raw Data

Figure above shows the raw data which is used for the simulation.



Neural Network Training (nntraintool)

Neural Network

Algorithms

Data Division: Random (dividerand)
Training: Bayesian Regularization (trainbr)
Performance: Mean Squared Error (mse)
Calculations: MEX

Progress

Epoch: 0 1000 iterations 1000
Time: 0:00:49
Performance: 5.08e+08 1.10e+05 0.00
Gradient: 9.19e+08 1.54e+06 1.00e-07
Mu: 0.00500 5.00e+06 1.00e+10

Fig.3. Deep Neural Network Model

Figure above shows the design of the deep neural network which is used for pattern recognition. It can be observed that the proposed model has 12 neurons in the input layer and 1 neuron in the output layer.

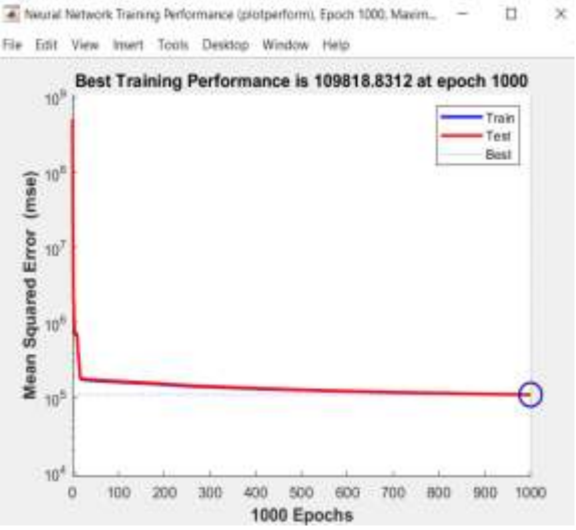


Fig.4. Training Convergence

Figure above shows the training convergence for the model designed. It can be observed that the training converges at 1000 iterations without any spikes in the cost function exhibiting the fact that the model is stable.

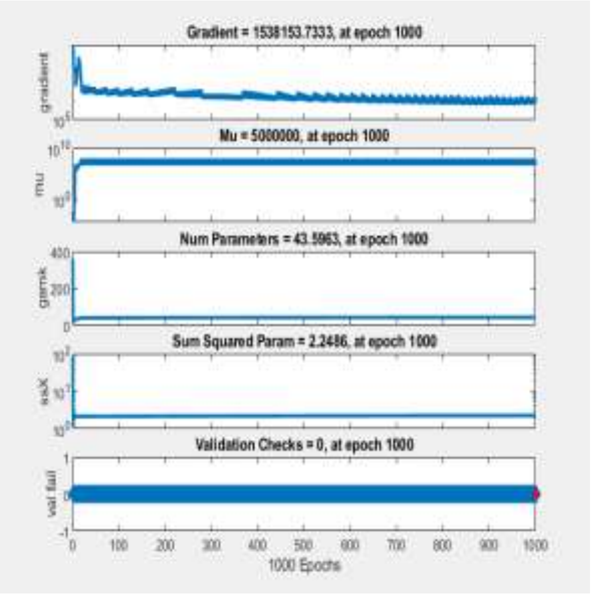


Fig.5. Training States

Figure above shows the training states for the models and variation of important parameters such as gradient, learning rate, sum squared parameters and validation checks.

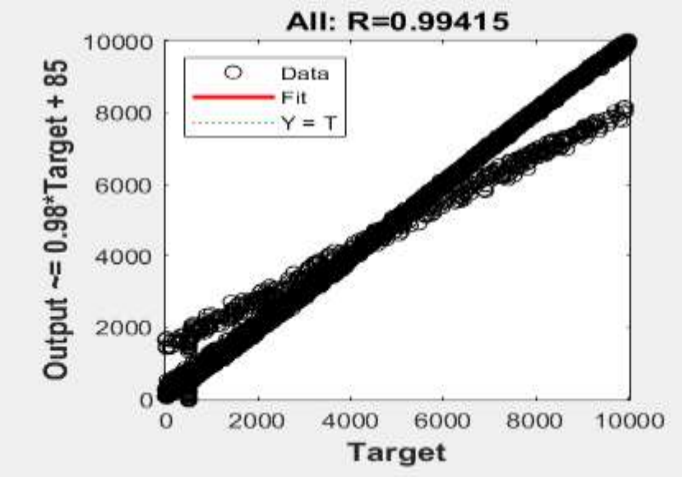


Fig.6. Model Regression

Figure above shows the overall model regression which is 0.99415 showing close similarity between the predicted and actual values of the variable.

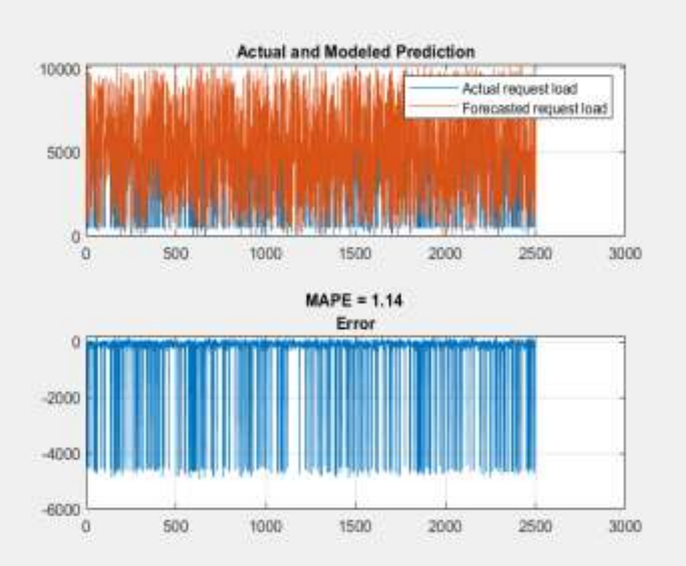


Fig. 7. Model Regression

Figure above shows the MAPE of the designed model which is 1.14% rendering an accuracy of 98.86%.

Table 1 Summary of Results

S.No.	Parameter	Value
1	Model	Neural Networks
2	Algorithm	Bayesian Optimization
3	Iterations	1000
4	Regression	0.99
5	MAPE (Previous Work)	6.8%
	Ref [6]:	
8.	**Obtained MAPE**	**1.14%**

The summary of results obtained is presented in table 1. It can be observed that the proposed approach attains improved MAPE % compared to existing work in the domain.

CONCLUSION

Deep learning provides a powerful and flexible approach for estimating cloud performance metrics in complex and dynamic cloud environments. By learning from large-scale monitoring data, deep learning models can capture intricate resource-performance relationships that traditional models fail to represent. Accurate performance estimation using deep learning not only improves monitoring and prediction accuracy but also enables proactive resource management, SLA assurance, and intelligent cloud optimization. As cloud systems continue to evolve, integrating deep learning with explainable AI and online learning techniques will further enhance the reliability and practicality of cloud performance estimation solutions. In this work, a Bayesian Momentum Based Optimization or Regularization neural network model has been developed for cloud performance prediction in which the dependent variable has been chosen as the number of instructions executed successfully. It is shown that the proposed work clearly outperforms existing work in terms of prediction accuracy.

REFERENCES

[1] J. Gao, H. Wang and H. Shen, "Task Failure Prediction in Cloud Data Centers Using Deep Learning," in IEEE Transactions on Services Computing, vol. 15, no. 3, pp. 1411-1422, 1 May-June 2022.

[2] D. Saxena and A. K. Singh, "A High Availability Management Model Based on VM Significance Ranking and Resource Estimation for Cloud Applications," in IEEE Transactions on Services Computing, vol. 16, no. 3, pp. 1604-1615, 1 May-June 2023.

[3] R Keller, L Häfner, T Sachs, G Fridgen," Scheduling flexible demand in cloud computing spot markets: A real options

approach", Business and Information Systems Engineering, Springer 2020, vol.62., pp. 25–39.

[4] D. Kong, S. Liu and L. Pan, "Amazon Spot Instance Price Prediction with GRU Network," 2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Dalian, China, 2021, pp. 31-36.

[5] D. Katayama, K. Kasai and T. Koita, "Migration Destination Selection Algorithm for Spot Instances using SPS," 2022 IEEE International Conference on Big Data (Big Data), Osaka, Japan, 2022, pp. 6690-6692

[6] D. Huang, L. Costero, A. Pahlevan, M. Zapater and D. Atienza, "CloudProphet: A Machine Learning-Based Performance Prediction for Public Clouds," in IEEE Transactions on Sustainable Computing, 2024, vol. 9, no. 4, pp. 661-676..

[7] Z. Amekraz and M. Y. Hadi, "CANFIS: A Chaos Adaptive Neural Fuzzy Inference System for Workload Prediction in the Cloud," in IEEE Access, 2022, vol. 10, pp. 49808-49828

[8] A. C. Zhou, J. Lao, Z. Ke, Y. Wang and R. Mao, "FarSpot: Optimizing Monetary Cost for HPC Applications in the Cloud Spot Market," in IEEE Transactions on Parallel and Distributed Systems, 2021, vol. 33, no. 11, pp. 2955-2967

[9] G. J. Portella, E. Nakano, G. N. Rodrigues, A. Boukerche and A. C. M. A. Melo, "A Novel Statistical and Neural Network Combined Approach for the Cloud Spot Market," in IEEE Transactions on Cloud Computing, 2023 vol. 11, no. 1, pp. 278-290.

[10] 6. Liu D, Cai Z, Lu Y (2019) Spot price prediction based dynamic resource scheduling for web applications. In: 2019 Seventh International Conference on Advanced Cloud and Big Data (CBD). IEEE, pp 78–83 7.

[11] Varshney P, Simmhan Y (2019) AutoBot: Resilient and cost-effective scheduling of a bag of tasks on spot VMs. IEEE Trans Parallel Distrib Syst 30(7):1512-1527

[12] Sharma P, Lee S, Guo T, Irwin D, Shenoy P (2017) Managing risk in a derivative IaaS cloud. IEEE Trans Parallel Distrib Syst 29(8):1750-1765

[13] Mishra AK, Yadav DK (2017) Analysis and prediction of Amazon EC2 spot instance prices. Int J Appl Eng Res 12(21):11205– 11212

[14] Teylo L, Arantes L, Sens P, Drummond LM (2019) A bag-of-tasks scheduler tolerant to temporal failures in clouds. In: 2019 31st International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD). IEEE, pp. 144–151

[15] Khandelwal V, Chaturvedi AK, Gupta CP (2020) Amazon EC2 spot price prediction using regression random forests. IEEE Trans Cloud Comput 8(1):59–72

[16] Y. Huang, J. Sun and Y. Tian, "A Bayesian Optimization Method for Finding the Worst-Case Scenarios of Autonomous Vehicles," in IEEE Transactions on Intelligent Transportation Systems, 2024, vol. 26, no. 1, pp. 529-543

[17] S. Amini, I. Vannieuwenhuyse and A. Morales-Hernández, "Constrained Bayesian Optimization: A Review," in IEEE Access, 2025, vol. 13, pp. 1581-1593.

[18] J. Gao, H. Wang and H. Shen, "Machine Learning Based Workload Prediction in Cloud Computing," 2020 29th International Conference on Computer Communications and Networks (ICCCN), Honolulu, HI, USA, 2020, pp. 1-9.