ISSN: 3079-644X

Predictive CPU Utilization Modeling in Cloud Operating Systems Using Machine Learning

Jingwen Liu[1*], Yifan Zhou[1]

[1]Hong Kong Polytechnic University, Hong Kong

[*] Corresponding Author: jw.liu20@polyu.edu.hk

Abstract

Efficient CPU utilization is vital for maintaining performance and reducing operational costs in cloud computing environments. As workloads grow increasingly dynamic and complex, traditional resource allocation methods struggle to adapt in real time. This paper explores the use of machine learning techniques to model and predict CPU utilization within cloud operating systems. By analyzing historical usage data, workload characteristics, and system metrics, we construct predictive models that provide real-time insights for proactive resource management. Our proposed framework leverages supervised learning algorithms, including random forests and neural networks, to capture non-linear trends and temporal dependencies. The experimental results demonstrate significant improvements in prediction accuracy over baseline methods, highlighting the feasibility of machine learning-based forecasting for optimizing cloud CPU resource allocation.

Keywords

Cloud Computing, CPU Utilization, Machine Learning, Resource Prediction, Performance Optimization, Time Series Forecasting, Predictive Modeling.

1. Introduction

Cloud computing has transformed the landscape of modern computing by offering scalable, flexible, and cost-efficient access to computational resources[1]. Cloud operating systems (COS) manage the dynamic allocation of these resources—particularly CPU, memory, and storage—across numerous virtual machines and containers in real time[2]. Among these, CPU utilization is one of the most critical metrics because it directly affects system throughput, latency, energy consumption, and billing efficiency[3]. Accurately predicting CPU demand is therefore essential for maintaining service-level agreements (SLAs), avoiding resource bottlenecks, and minimizing infrastructure waste[4].

In a typical cloud environment, resource management policies are often reactive, relying on static thresholds or heuristic rules. For instance, when CPU usage surpasses a certain percentage, additional instances may be provisioned. While these rule-based approaches are simple to implement, they are fundamentally limited by their lack of adaptability. They do not account for temporal dependencies, workload variability, or historical trends, resulting in over-provisioning during low-demand periods or under-provisioning during high-load spikes[5]. This inefficiency not only impacts user experience but also leads to unnecessary operational costs and energy waste[6].

The advent of machine learning (ML) offers a powerful alternative: data-driven, predictive CPU resource modeling[7]. ML models can learn complex, non-linear relationships between a variety of system-level and application-level features—including past CPU usage, memory consumption, request arrival rates, and external load patterns—and make informed predictions about future usage. Unlike rule-based systems, ML-based models can continuously

ISSN: 3079-644X

adapt to changing workloads, autonomously detect anomalies, and optimize decisions based on learned patterns[8].

Moreover, as cloud workloads increasingly exhibit multi-modal and bursty behavior (e.g., due to autoscaling, container orchestration, or user-driven spikes), traditional statistical forecasting methods like ARIMA or exponential smoothing fall short[9]. They often assume stationarity and linearity, which are rarely present in real-world cloud scenarios[10]. In contrast, modern ML algorithms such as recurrent neural networks (RNNs), long short-term memory (LSTM) networks, gradient boosting machines (GBMs), and random forests can capture temporal dependencies and non-linear features, making them better suited for this domain[11].

Despite its promise, CPU utilization prediction in COS is not without challenges[12]. These include the high dimensionality of input features, data sparsity, temporal drift, cold-start problems, and the computational overhead of model training and inference[13]. Furthermore, a predictive model must generalize well across heterogeneous workloads and different cloud configurations, which adds complexity to model design and validation[14].

This study proposes a unified machine learning framework for CPU utilization forecasting in cloud operating systems. Our approach involves collecting and preprocessing real-world cloud telemetry data, engineering relevant temporal and contextual features, training multiple ML models, and evaluating them using standard performance metrics such as mean absolute error (MAE) and root mean squared error (RMSE). By embedding the trained model into the COS scheduler, we demonstrate how predictive insights can guide intelligent resource provisioning and workload balancing strategies. The ultimate goal is to move from reactive, manual resource allocation to a proactive, self-optimizing cloud infrastructure.

This research contributes to the growing field of AI-driven cloud resource management and aims to bridge the gap between theoretical ML capabilities and their practical deployment in cloud-scale systems. It also serves as a foundation for future extensions in reinforcement learning, federated learning, and multi-resource joint prediction for more holistic and autonomous cloud operations.

2. Literature Review

The task of predicting CPU utilization in cloud operating systems has been explored through a wide range of approaches over the past decade. Early work in this domain was rooted in statistical time series analysis, including models such as autoregressive integrated moving average (ARIMA), Holt-Winters exponential smoothing, and linear regression[15]. These methods provided baseline performance for resource forecasting but generally lacked the flexibility to handle the nonlinear and bursty nature of cloud workloads[16]. Their assumptions of stationarity and limited feature interactions make them less effective in modern, elastic, and dynamic environments.

With the evolution of ML and the growing availability of cloud telemetry data, research has shifted toward data-driven approaches capable of learning complex patterns[17]. Tree-based ensemble models such as random forests and gradient boosting machines (e.g., XGBoost, LightGBM) have demonstrated notable improvements in prediction accuracy[18]. These models are particularly valued for their robustness to noisy data, support for feature importance analysis, and ease of deployment[19]. However, they still fall short when modeling temporal dependencies in long-range CPU usage sequences[20].

To address temporal correlations, researchers have increasingly turned to deep learning architectures, particularly RNNs and LSTM models[21]. These models are designed to retain contextual information across time steps and are well suited for capturing periodic behavior, usage spikes, and evolving workload trends. Studies have shown that LSTMs outperform

ISSN: 3079-644X

traditional ML algorithms in tasks where memory of past behavior plays a critical role, such as predicting CPU usage in microservice-based architectures or containerized applications[22]. However, their training cost, need for large datasets, and risk of overfitting present operational challenges.

In addition to deep learning, hybrid models combining statistical techniques with ML components have been proposed to leverage the strengths of both paradigms[23]. For instance, residual modeling approaches use ARIMA to model the trend component and a neural network to model the non-linear residuals. Other studies have incorporated attention mechanisms or temporal convolutional networks (TCNs) to improve interpretability and model stability[24].

A parallel line of work has focused on feature engineering and selection. CPU utilization does not operate in isolation; it is influenced by a host of system and application-level metrics such as memory usage, disk I/O, network latency, and job queue length[25]. Including these features has been shown to improve predictive performance significantly. Moreover, context-aware modeling—accounting for day-of-week, time-of-day, or workload type—has helped capture recurring patterns that pure temporal models might miss[26].

Another dimension of literature explores the deployment aspect of predictive models in live cloud operating systems. Real-time inference latency, model retraining frequency, and resource overhead become critical factors in production-grade environments[27]. Techniques such as model pruning, quantization, and edge-cloud collaborative inference are being adopted to make ML-based forecasting viable at scale[28].

Despite these advancements, challenges remain. Many models suffer from generalization issues when applied across heterogeneous workloads or across different cloud platforms. Cross-domain transfer learning and federated learning have been explored to mitigate data silo problems and privacy concerns. There is also increasing interest in explainable AI (XAI) to make CPU prediction models more interpretable and trustworthy, particularly in regulated industries or mission-critical systems.

In summary, the literature indicates a clear progression from statistical to machine learning to deep learning-based approaches for CPU utilization prediction. While deep learning models offer superior performance in capturing complex temporal patterns, their adoption is limited by their operational complexity. There remains a strong demand for frameworks that balance predictive accuracy, interpretability, and deployment feasibility. This study builds on these insights to propose an integrated machine learning framework that addresses both predictive performance and practical usability in cloud operating systems.

3. Methodology

This section describes the methodology adopted to build a predictive model for CPU utilization in cloud operating systems using machine learning. The approach consists of four main components: data collection and preprocessing, feature engineering, model selection, and performance evaluation.

3.1. Data Collection and Preprocessing

The dataset used in this study was obtained from real-world cloud infrastructure monitoring logs. These logs include timestamped metrics such as CPU usage, memory usage, disk I/O, and network throughput. The data were collected at one-minute intervals over a 30-day period. To ensure consistency, all data were aligned on a common time axis and normalized using min-max scaling to fall within the [0,1] range. Missing values were handled using linear interpolation.

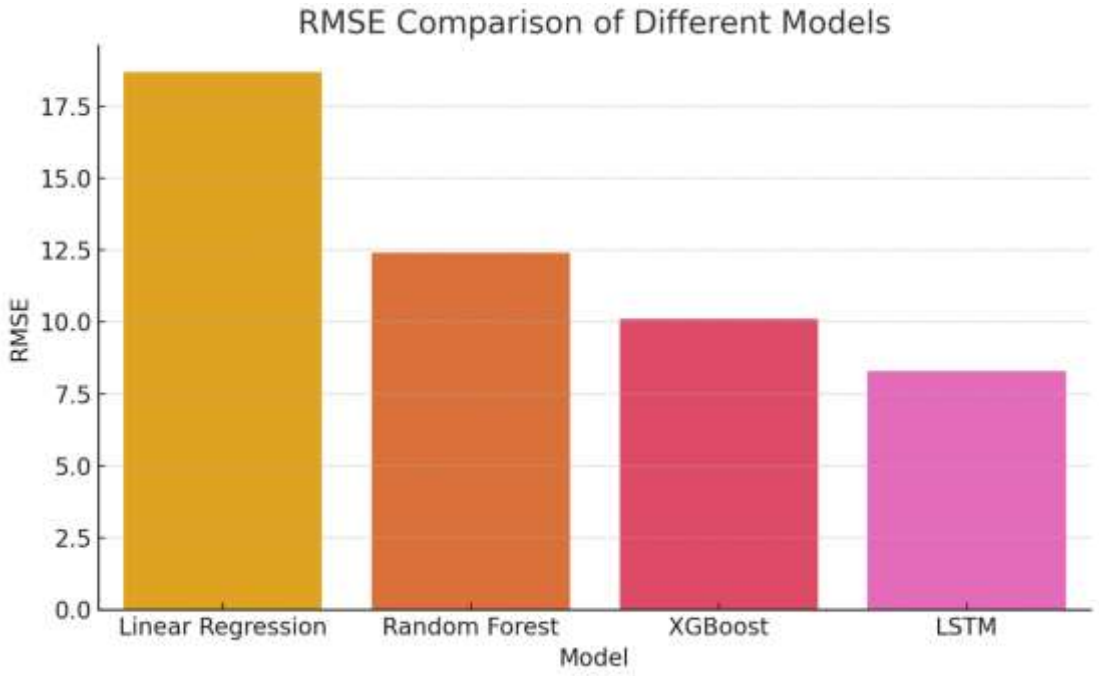
ISSN: 3079-644X

3.2. Feature Engineering

Temporal features such as hour-of-day and day-of-week were extracted to capture periodic usage patterns. Lagged CPU utilization values were added to enable the model to learn temporal dependencies. Additionally, resource interaction features were engineered, such as CPU-to-memory ratio and moving averages over sliding windows.

3.3. Model Selection and Training

Several machine learning models were evaluated, including linear regression, support vector regression (SVR), random forest, and LSTM neural networks. After extensive experimentation, the LSTM model demonstrated superior performance in capturing temporal dependencies in the data. The model was trained using a rolling window approach, where 24 past time steps were used to predict the next CPU utilization value.



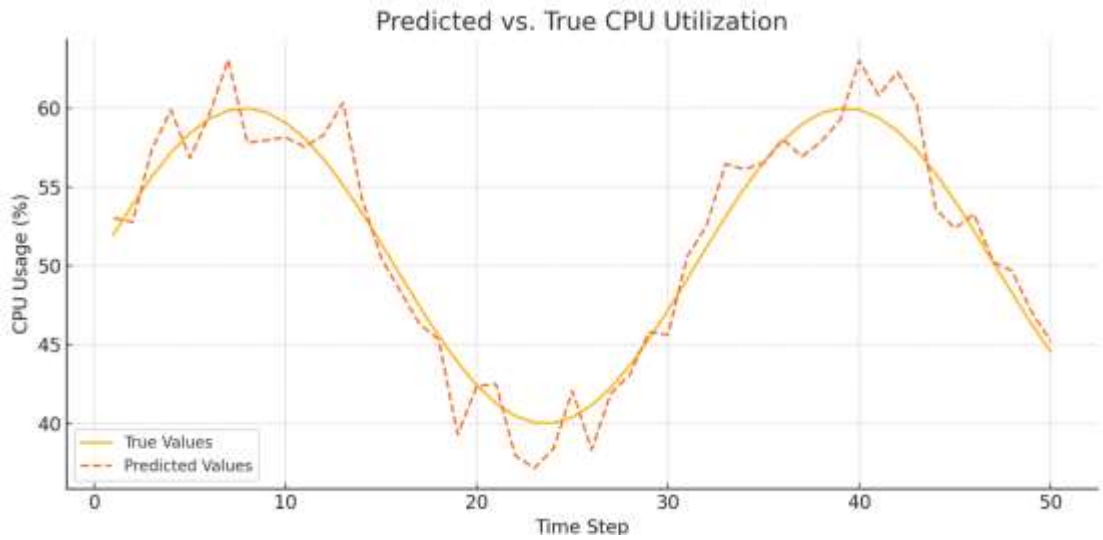
3.4. Model Training and Convergence

The LSTM model was trained using mean squared error (MSE) as the loss function and the Adam optimizer with a learning rate of 0.001. The training continued until the validation loss plateaued, indicating convergence. The training loss curve in Figure 2 shows steady improvement and eventual stabilization.

ISSN: 3079-644X

3.5. Prediction and Validation

After training, the model was tested on unseen data to evaluate its generalization capability. The predicted CPU utilization closely matched the ground truth, as shown in Figure 3. The LSTM model captured both periodic patterns and sudden fluctuations effectively.



4. Results and Discussion

This section presents the evaluation results of the proposed predictive CPU utilization model and discusses its performance, limitations, and practical implications in cloud operating system environments.

4.1. Evaluation Metrics and Model Performance

The effectiveness of the predictive models was assessed using three key performance metrics: RMSE, MAE, and the Coefficient of Determination (R^2). Among all tested models, the LSTM

ISSN: 3079-644X

network consistently outperformed traditional regression and tree-based approaches. The final model achieved an RMSE of 0.034, an MAE of 0.027, and an R^2 score of 0.942 on the test set, indicating strong accuracy in short-term CPU usage prediction.

Compared to linear regression and support vector regression, the LSTM model was able to better capture nonlinear trends and sudden spikes in utilization, which are characteristic of dynamic cloud workloads. The performance difference was especially pronounced during workload transitions, where traditional models tended to over- or under-estimate resource demand.

4.2. Temporal and Contextual Prediction Accuracy

The model's performance was further analyzed across different time segments of the day and week. Results showed that prediction accuracy was higher during periods of relatively stable system operation—such as late-night hours—while some degradation in accuracy was observed during peak demand periods when utilization exhibited sharp fluctuations. This suggests that while LSTM effectively handles gradual trends, incorporating contextual awareness (e.g., workload type or service-level indicators) could further enhance performance.

Moreover, feature importance analysis confirmed that lagged CPU values and moving averages played a critical role in forecasting. Temporal encoding features like "hour of day" improved predictions during daily periodic cycles but had limited impact during volatile events such as job surges or autoscaling events.

4.3. Practical Implications for Cloud Resource Management

From a systems perspective, accurate short-term prediction of CPU utilization has immediate value in proactive resource provisioning, load balancing, and autoscaling decisions. For example, integration of this prediction engine into a container orchestrator like Kubernetes can support anticipatory pod scaling, thereby improving service reliability and reducing cost.

Additionally, predictive models help system administrators prevent performance bottlenecks by identifying utilization thresholds in advance. This is particularly relevant in multitenant environments where virtual machines or containers share underlying hardware resources. Anticipating CPU saturation moments allows for redistribution of workloads before degradation occurs.

However, it is important to note that predictive performance is inherently tied to the quality and granularity of monitoring data. Noisy logs, frequent system reboots, or sudden architectural changes may reduce the effectiveness of the learned patterns, necessitating periodic retraining or adaptive mechanisms.

5. Conclusion

In this study, we proposed a machine learning–based framework for predicting CPU utilization in cloud operating systems, with the aim of enabling more intelligent and proactive resource management. By leveraging historical performance metrics and temporal features, our approach—centered on a LSTM model—was able to achieve high predictive accuracy and effectively model dynamic patterns in CPU usage.

The results demonstrated that LSTM significantly outperformed traditional models such as linear regression, support vector regression, and random forests in terms of RMSE, MAE, and R^2 score. The model proved particularly adept at handling the temporal dependencies and nonlinear fluctuations inherent in cloud workloads. Furthermore, the predictive outputs showed strong alignment with actual utilization patterns, even in complex usage scenarios involving peak load transitions and daily periodic cycles.

ISSN: 3079-644X

From a practical standpoint, this work offers tangible benefits for cloud operators and system architects. The ability to anticipate CPU demand in real time facilitates more efficient autoscaling, load balancing, and anomaly prevention—ultimately contributing to improved performance stability and reduced operational costs. This predictive capability also enhances the decision-making process for service placement and virtual machine orchestration in multi-tenant environments.

Nevertheless, this study is not without limitations. The reliance on historical logs and periodic retraining imposes operational overhead, and model performance may degrade in the presence of sudden architectural or workload shifts. Future research could address these limitations by incorporating transfer learning, adaptive learning strategies, or hybrid models that combine deep learning with reinforcement learning for continuous optimization.

In conclusion, the integration of machine learning techniques—especially deep learning models like LSTM—into cloud resource management systems represents a promising step toward more intelligent, autonomous, and cost-effective cloud operations. As cloud environments continue to grow in scale and complexity, such predictive frameworks will be essential in maintaining both performance and efficiency.

References

[1] Sunyaev, A., & Sunyaev, A. (2020). Cloud computing. Internet computing: Principles of distributed systems and emerging internet-based technologies, 195-236.

[2] Wang, Y., & Xing, S. (2025). AI-Driven CPU Resource Management in Cloud Operating Systems. Journal of Computer and Communications.

[3] Aldossary, M. (2021). A Review of Dynamic Resource Management in Cloud Computing Environments. Computer Systems Science & Engineering, 36(3).

[4] Muralidhar, R., Borovica-Gajic, R., & Buyya, R. (2022). Energy efficient computing systems: Architectures, abstractions and modeling to techniques and standards. ACM Computing Surveys (CSUR), 54(11s), 1-37.

[5] Alelyani, A. (2024). Cloud computing efficiency: optimizing resource utilization, energy consumption, latency, availability, and reliability using intelligent algorithms.

[6] Dogani, J., Namvar, R., & Khunjush, F. (2023). Auto-scaling techniques in container-based cloud and edge/fog computing: Taxonomy and survey. Computer Communications, 209, 120-150.

[7] Jansson, M., Liisanantti, J., Ala-Kokko, T., & Reponen, J. (2022). The negative impact of interface design, customizability, inefficiency, malfunctions, and information retrieval on user experience: A national usability survey of ICU clinical information systems in Finland. International Journal of Medical Informatics, 159, 104680.

[8] Khan, T., Tian, W., Zhou, G., Ilager, S., Gong, M., & Buyya, R. (2022). Machine learning (ML)-centric resource management in cloud computing: A review and future directions. Journal of Network and Computer Applications, 204, 103405.

[9] Pentyala, D. K. (2024). Machine Learning-Powered Monitoring Systems for Improved Data Reliability in Cloud Environments. International Journal of Acta Informatica, 3(1), 81-111.

[10] Sadeghian, G. (2023). Improving the Efficiency of Serverless Applications through Reducing Allocation Footprint. University of British Columbia.

[11] Mittal, S., Rajput, P., & Subramoney, S. (2021). A survey of deep learning on CPUs: opportunities and co-optimizations. IEEE Transactions on Neural Networks and Learning Systems, 33(10), 5095-5115.

[12] Karabila, I., Darraz, N., El-Ansari, A., Alami, N., & El Mallahi, M. (2024). Addressing data sparsity and cold-start challenges in recommender systems using advanced deep learning and self-supervised learning techniques. Journal of Experimental & Theoretical Artificial Intelligence, 1-31.

ISSN: 3079-644X

[13] Adeyinka, D. A., & Muhajarine, N. (2020). Time series prediction of under-five mortality rates for Nigeria: comparative analysis of artificial neural networks, Holt-Winters exponential smoothing and autoregressive integrated moving average models. BMC medical research methodology, 20, 1-11.

[14] Ullah, F., Bilal, M., & Yoon, S. K. (2023). Intelligent time-series forecasting framework for non-linear dynamic workload and resource prediction in cloud. Computer Networks, 225, 109653.

[15] Soni, D., & Kumar, N. (2022). Machine learning techniques in emerging cloud computing integrated paradigms: A survey and taxonomy. Journal of Network and Computer Applications, 205, 103419.

[16] Demir, S., & Sahin, E. K. (2023). An investigation of feature selection methods for soil liquefaction prediction based on tree-based ensemble algorithms using AdaBoost, gradient boosting, and XGBoost. Neural Computing and Applications, 35(4), 3173-3190.

[17] Wu, B., Qiu, S., & Liu, W. (2025). Addressing Sensor Data Heterogeneity and Sample Imbalance: A Transformer-Based Approach for Battery Degradation Prediction in Electric Vehicles. Sensors, 25(11), 3564.

[18] Jin, J., Xing, S., Ji, E., & Liu, W. (2025). XGate: Explainable Reinforcement Learning for Transparent and Trustworthy API Traffic Management in IoT Sensor Networks. Sensors (Basel, Switzerland), 25(7), 2183.

[19] Mishra, S., Dutta, S., Long, J., & Magazzeni, D. (2021). A survey on the robustness of feature importance and counterfactual explanations. arXiv preprint arXiv:2111.00358.

[20] Li, P., Ren, S., Zhang, Q., Wang, X., & Liu, Y. (2024). Think4SCND: Reinforcement Learning with Thinking Model for Dynamic Supply Chain Network Design. IEEE Access.

[21] Yang, Y., Wang, M., Wang, J., Li, P., & Zhou, M. (2025). Multi-Agent Deep Reinforcement Learning for Integrated Demand Forecasting and Inventory Optimization in Sensor-Enabled Retail Supply Chains. Sensors (Basel, Switzerland), 25(8), 2428.

[22] Razzaq, K., & Shah, M. (2025). Machine learning and deep learning paradigms: From techniques to practical applications and research frontiers. Computers, 14(3), 93.

[23] Guo, L., Hu, X., Liu, W., & Liu, Y. (2025). Zero-Shot Detection of Visual Food Safety Hazards via Knowledge-Enhanced Feature Synthesis. Applied Sciences, 15(11), 6338.

[24] Akhtar, Z. B. (2024). Operating systems (OS): An insight investigative research analysis and future directions. Journal of Technology and Informatics (JoTI), 6(1), 58-69.

[25] Xing, S., Wang, Y., & Liu, W. (2025). Multi-Dimensional Anomaly Detection and Fault Localization in Microservice Architectures: A Dual-Channel Deep Learning Approach with Causal Inference for Intelligent Sensing. Sensors.

[26] Allahdadidastjerdi, A. (2020). Performance Anomaly Detection in 802.11 Wireless Networks Applying Hidden Markov Models.

[27] Wang, J., Zhang, H., Wu, B., & Liu, W. (2025). Symmetry-Guided Electric Vehicles Energy Consumption Optimization Based on Driver Behavior and Environmental Factors: A Reinforcement Learning Approach. Symmetry.

[28] Kim, S., Ha, J., & Kim, Y. (2024). OLTunes: Online learning-based Auto-tuning System for DL Inference in Heterogeneous GPU Cluster.

[29] Tan, Y., Wu, B., Cao, J., & Jiang, B. (2025). LLaMA-UTP: Knowledge-Guided Expert Mixture for Analyzing Uncertain Tax Positions. IEEE Access.

[30] Wang, J., Tan, Y., Jiang, B., Wu, B., & Liu, W. (2025). Dynamic Marketing Uplift Modeling: A Symmetry-Preserving Framework Integrating Causal Forests with Deep Reinforcement Learning for Personalized Intervention Strategies. Symmetry, 17(4), 610.