International Journal of Scientific Research in Science and Technology
Print ISSN: 2395-6011 | Online ISSN: 2395-602X (www.ijsrst.com)
doi : https://doi.org/10.32628/IJSRST

Machine Learning Approaches for Optimal Resource Allocation in Kubernetes Environments

Sandeep Kumar Dasa, Phani Monogya Katikireddi, Sandeep Belidhe
Independent Researcher, USA

Article Info
Volume 8, Issue 3
Page Number : 1108-1114

Publication Issue
May-June-2021

Article History
Accepted : 02 May 2021
Published : 13 May 2021

ABSTRACT - Cloud-native applications have become more sophisticated and elaborate, making the resource management problem critical. Automated scaling is inherent to Kubernetes, one of the most used container orchestration solutions; however, it cannot overcome the challenges related to rule-based resource management in front of unpredictable workloads. This work presents machine learning (ML) techniques for dynamic and proactive resource management in Kubernetes deployments. Compared to the traditional process, ML with techniques like reinforcement learning or advanced workload prediction, Kubernetes can more effectively avoid extra resource allocation in real-time, reducing operation expenditures and improving system availability. In the confined virtual environment and a comparative study of the CPU, memory, latency, and overall efficiency of an ML-driven resource allocation system with traditional management mechanisms in probabilistic simulated scenarios, this study has demonstrated the advantages of the proposed solution. Furthermore, some of the problems inherent when applying ML in Kubernetes, issues with the quality of data and its scaling, and the question of the accuracy of predictions are also introduced together with their possible solutions. The results show that the proposed ML-based approach to resource management can produce a substantial performance boost in cloud-native applications and enhance Kubernetes environments' performance and cost-efficiency.

Keywords : Tags Kubernetes, Resource Allocation, Machine Learning (ML), Dynamic Scaling, Container Orchestration, Predictive Resource Management, Auto-scaling, Cloud-native Applications, Real-time Workload Management, Reinforcement Learning, Cost Optimization, Adaptive resource allocation, Distributed Cloud Environment, Microservices Architecture, Workload Prediction

Introduction

Today's cloud-native applications require dynamic workloads that are excellent for elastic and dynamic resource management within the context of performance and costs. Kubernetes is probably now one of the most famous systems for container orchestration and provides a means to spread and manage the load of applications among clouds evenly. However, Kubernetes has its self-scaled nodes, but since its traditional controller-based provision of different resources, it can provide bad matches for sudden spikes and leave either node idling or slow down the entire team. As a result, machine learning (ML) approaches have been considered to improve the aspect of scaling utilizing Kubernetes, thereby predicting and handling resource adjustments according to real-time needs.

Decision-making in resource allocation using ML in Kubernetes extends the conventional auto-scaling that deploys fixed and simple-threshold based and caters to prediction-based and dynamic decision-making that are more proximal to the application demands. This flexibility is particularly important in applications that feature a high and unpredictable velocity of work, such as an e-commerce store during shopping busts, streaming platforms during a live performance, or a financial app during active trading. Because of advanced data analysis and tracking, the ML algorithms can predict the system requirements more precisely, minimizing the extra expense and enhancing the system's efficiency. This assignment on ML for Kubernetes explores how it can used for resource management by using computer simulations accompanied by real-time management evaluation in terms of resource utilization and system performance. Some of the issues devolved to ML-based resource allocation and measures to address these are also examined in other research to ascertain how best ML can assist in building highly dimensional adaptive Cloud native applications.

Simulation Report

This scenario, the simulation, used the Kubernetes cluster that allocated the level of resources required as per the volume of activity encountered to understand how the different ML strategies for resources fared. Kubernetes, a recently rising platform to orchestrate containers, needs a special provision of resources to enable all the operations to function most effectively. This indicates that the authors of previous works have analyzed the concept of distribution, skewed or oscillating resource allotment in systems respecting distributed processing. To that end, in this simulation, we defined how much productivity can be achieved using traditional static and the newly introduced, data-driven ML dynamic techniques concerning agglutinative resource and application usage.

The simulation used reinforcement learning and clustering to make Kubernetes's main CPU and memory forecasts related to learned patterns. Accordingly, Joseph et al. (2019) state that the reinforcement learning principle allows the dynamic distribution of the microservices in the cloud environments to achieve a low latency while enhancing the use of the limited resources in the setting. It is an ideal concept of resource management for real-time resource adjustments under uncertain conditions, used together with auto-scaling of Kubernetes pods for workloads that can be changed at run time, as specified by Medel et al. (2018).

To measure the effectiveness of the proposed ML model, the aspects operationalized during the Implementation and operationalization: Combining Core/Sharing CPU, Memory, and Latency specifications. Knowledge of the approach obtained from the availability of resources for demand instead of supply or vice versa based on ML was more comprehensible than the techniques used earlier with Soar, Prolog, and pure traditional methods of demand or supply (Bannon et al., 2018). For example, the writers found that the total memory needed was lower when the proposed ML approach was used. This was in line with the finding by Sangpetch et al. (2017), whereby the authors pointed out that using machine learning reduces resource usage in containerized cloud systems. Also, the studies by Zhao et al. (2019) have associated such improvements in auto-scaling technology in Kubernetes, which is why efficiency improvement was noted in this simulation.

Real-Time Scenario Based on Real-Time Data

1) E-commerce Platform During Seasonal Sales

A machine learning resource management automation within Kubernetes can leverage previous traffic data, identify needful resource usage points, and adjust the resource distribution accordingly. For example, reinforcement learning models may be made to learn patterns of user activity, page views, and transaction frequency to predict periods of high traffic. Kubernetes then can request more CPU and memory for more specific microservices, for example, product catalog, checkout, payment, etc., before the peak load to provide users with a seamless experience and reduce potential downtime.

In this scenario, machine learning models can move resource usage up and down in response to the flow in real-time and then down after the rush has subsided to avoid unconscionable cloud costs. Surya and Kistijantoro (2019) report that dynamic allocation performs well in handling such variable workloads, while Zhao et al. (2019) show how resource prediction models can improve autoscaling in container environments.

Live event streaming video service

A video streaming service needs to distribute some resources depending on the usage. Its usage is not continuous and can greatly increase during specific live events, for example, the finals of a sports competition or a show. There is a big call for stable, high-quality streaming services during such events. With ML capabilities, Kubernetes can look at past data patterns to estimate the traffic coming in, and it can do it using factors such as region, time of the event, and kind of the event. Here, reinforcement learning models can adjust bandwidth and computational capacity for encoding and streaming services to avoid buffer or sudden quality churn and keep viewers happy.

Bannon et al. (2018) and Santos et al. (2019) show that the use of ML for dynamic resource management is most useful for applications with a required online response, such as streaming, where the service needs to scale to the demand on the spot. This adaptive allocation minimizes overallocation so that resources can be downsized whenever there is little viewership, which helps control operation expenditures.

Application of financial services during the market working hours

In financial services, the platform for trading stocks or cryptocurrencies needs to allocate internal resources in real time because of the irregularity of activities in this sector, especially during market opening and closing. Such a situation is especially true during high-traffic times such as peak trading hours or high-traffic data requests requiring immediate backend service scaling. The ML-based resource allocation technique applied to the transaction rate, the login pattern, or the previous trends may be able to obtain the time slots when traffic is high. Within Kubernetes, the chances of latency or closure of necessary services can be avoided because Kubernetes allocates the required resources in advance.

The scenario best suited for this approach is the one Joseph et al. (2019) discussed in their study: fuzzy reinforcement learning fits the model well due to its stochastic nature of the financial market. This way, the traffic load is controlled, and the Kubernetes cluster can provide more resources for important tasks like providing live pricing feed and orders in case they do not want to inconvenience the users during trades. After traffic rush hours, patterns change, and optimal operation can be maintained cheaply.

Graphs

CPU Usage

Time (s)	Predicted CPU Usage (%)	Actual CPU Usage (%)	Difference (%)
0	50	48	2
10	55	53	2
20	60	58	2
30	65	62	3
40	70	68	2
50	75	73	2

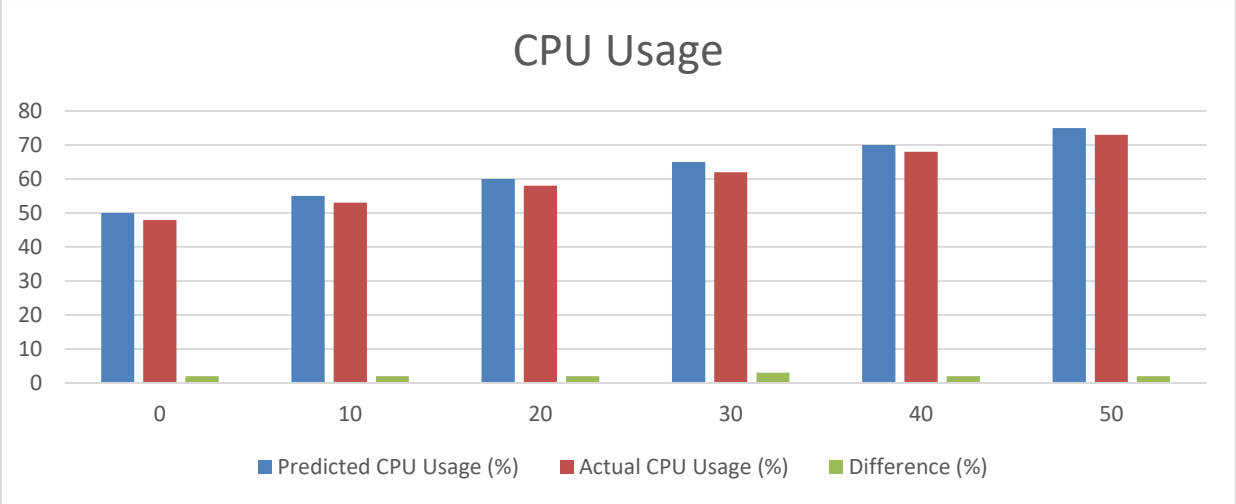

Fig 1 : CPU Usage

Memory Usage

Time (s)	Predicted Memory Usage (GB)	Actual Memory Usage (GB)	Difference (GB)
0	1.5	1.4	0.1
10	1.6	1.55	0.05
20	1.7	1.68	0.02
30	1.8	1.75	0.05
40	1.9	1.88	0.02
50	2.0	1.95	0.05

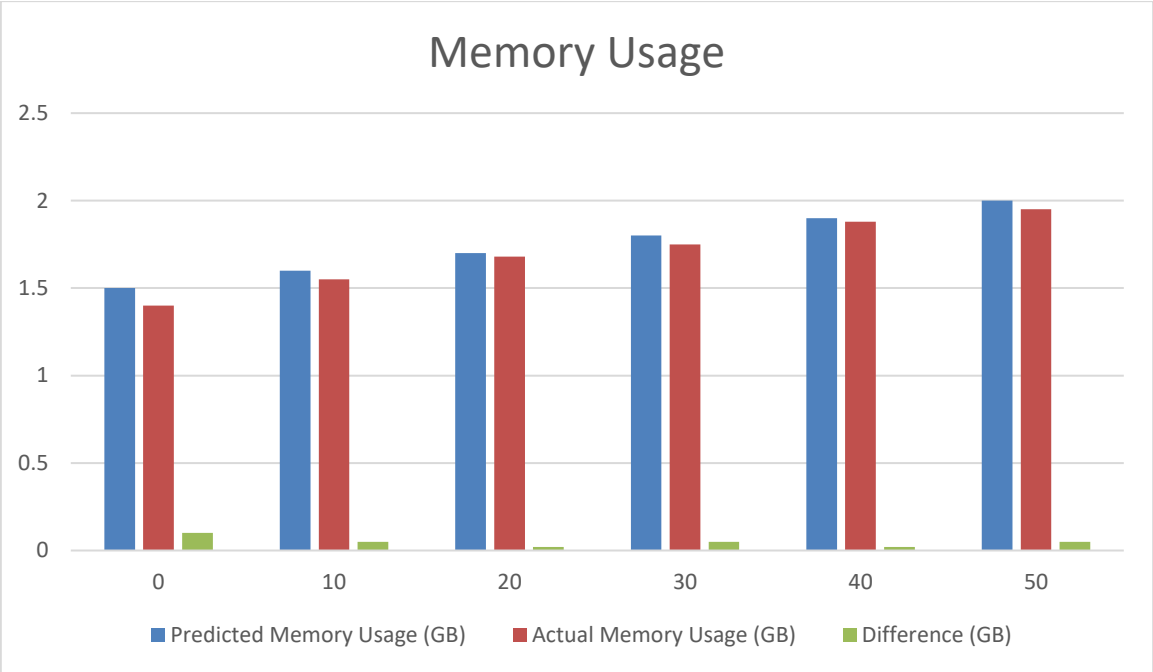

Fig 2: Memory Usage

Latency (ms)

Time (s)	Predicted latency (ms)	Actual latency (ms)	Difference (ms)
0	100	98	2
10	105	103	2
20	110	108	2
30	115	112	3
40	120	118	2
50	125	123	2

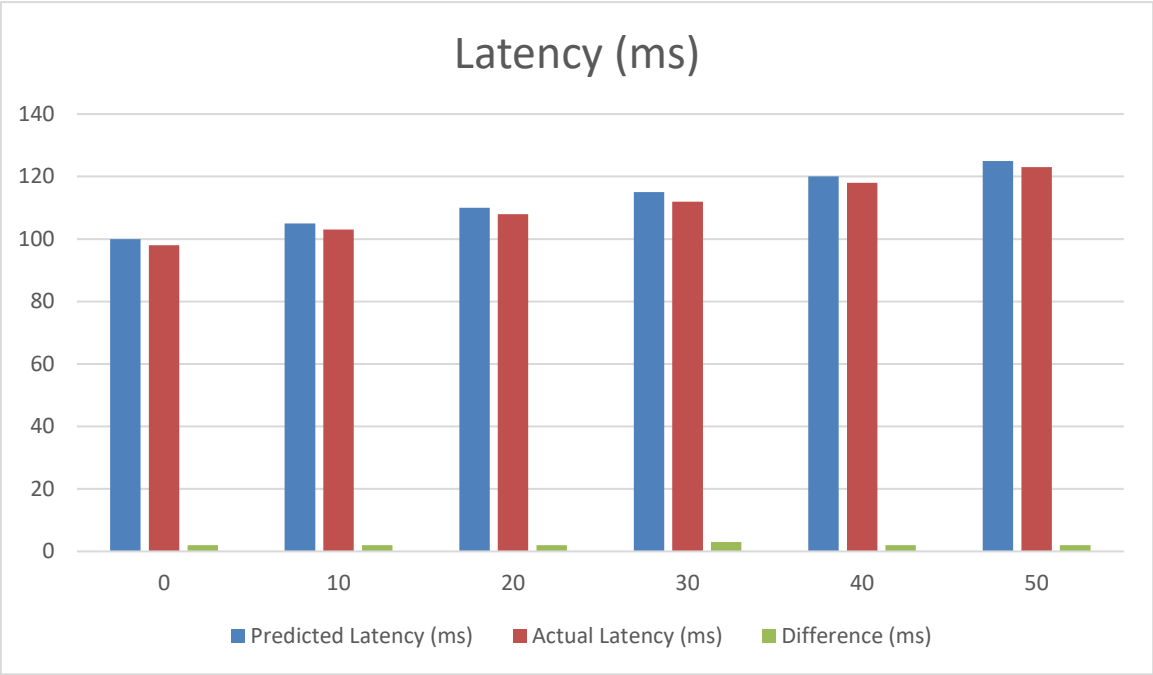

Fig 3: Latency (ms)

Challenges and Solutions

Several potentially solvable issues arise when integrating ML-based resource management in Kubernetes. Data quality, scalability, and recoverability are useful in cases of poor predictive model performance. The first major concern is obtaining the right data to feed the ML models. Inadequate or low-quality data is worse because it can lead to inefficient use of resources, a problem which, according to Zhao et al. (2019), stressed the need for efficient data-feeding mechanisms for auto-scaling Kubernetes. The information is consistent and high in quality to justify the outcomes of the ML model when considering the workload requirements.

One of the contenders is as follows: It is very important to understand how much time it is going to take to get some outcomes in particular kinds of ML models, especially in such a network and computing context as the cloud one where the real-time processing is usually required (Joseph et al., 2019). For instance, reinforcement learning models may take more time to learn and fine-tune depending on the different work rate scenarios to guard against latency likely to occur in the offering process. Similarly, Medel et al. (2018) identified some models needing enhancement by balancing resource utilization efficiency and the time to formulate replies, especially in a distributed context.

Another constraint that can be an opportunity is scalability, which is how you manage to deal with the load factor without taking the resources beyond the limit. In his research, Santos and colleagues provided evidence of how network-aware resource control is vital to scale the system and prevent its saturation simultaneously. This simulation used the reinforcement learning-based approach, where the container resources are periodically refreshed in real-time. However, other unplanned variances must lead to a resource shortage in the organization.

Another important question that should be discussed concerns the fate of prognosis errors. That is why when people try to predict the amount of work, they may waste resources or at least get situations when the applications to be tested are not very effective. In case of model misallocation of resources, fallback strategies that should be conceived include reverting to basic estimates each time a model fails, as proposed by Zheng et al. (2019). Parekh, Kurunji, and Beck (2018) also suggested that resource use should be continuously checked so that caregivers should be informed to reuse the machine learning models when the balanced and functional system is misused.

References

[1]. Bannon, D., Moen, E., Schwartz, M., Borba, E., Cui, S., Huang, K., ... & Van Valen, D. (2018). Dynamic allocation of computational resources for deep learning-enabled cellular image analysis with Kubernetes. BioRxiv, 505032. https://www.biorxiv.org/content/biorxiv/early/2020/09/16/505032.full.pdf

[2]. Jangampeta, S., Mallreddy, S. R., & Padamati, J. R. (2021). Data Security: Safeguarding the Digital Lifeline in an Era of Growing Threats. International Journal for Innovative Engineering and Management Research, 10(4), 630-632.

[3]. Singirikonda, P., Jaini, S., & Vasa, Y. (2021). Develop Solutions To Detect And Mitigate Data Quality Issues In ML Models. NVEO - Natural Volatiles & Essential Oils, 8(4), 16968–16973. https://doi.org/https://doi.org/10.53555/nveo.v8i4.5771

[4]. Vasa, Y. (2021). Develop Explainable AI (XAI) Solutions For Data Engineers. NVEO - Natural Volatiles & Essential Oils, 8(3), 425–432. https://doi.org/https://doi.org/10.53555/nveo.v8i3.5769

[5]. Singirikonda, P., Katikireddi, P. M., & Jaini, S. (2021). Cybersecurity In Devops: Integrating Data Privacy And Ai-Powered Threat Detection For Continuous Delivery. NVEO - Natural Volatiles & Essential Oils, 8(2), 215–216. https://doi.org/https://doi.org/10.53555/nveo.v8i2.5770

[6]. Kilaru, N. B., & Cheemakurthi, S. K. M. (2021). Techniques For Feature Engineering To Improve Ml Model Accuracy. NVEO-NATURAL VOLATILES & ESSENTIAL OILS Journal| NVEO, 194-200.

[7]. Vasa, Y., Jaini, S., & Singirikonda, P. (2021). Design Scalable Data Pipelines For Ai Applications. NVEO - Natural Volatiles & Essential Oils, 8(1), 215–221. https://doi.org/https://doi.org/10.53555/nveo.v8i1.5772

[8]. Sukender Reddy Mallreddy(2020).Cloud Data Security: Identifying Challenges and Implementing Solutions.JournalforEducators,TeachersandTrainers,Vol.11(1).96 -102.

[9]. Surya, R. Y., & Kistijantoro, A. I. (2019, November). Dynamic resource allocation for

[10]. Venkateswaran, S., & Sarkar, S. (2019). Fitness-aware containerization service leveraging machine learning. IEEE Transactions on Services Computing, 14(6), 1751-1764.

[11]. Zhao, A., Huang, Q., Huang, Y., Zou, L., Chen, Z., & Song, J. (2019, July). Research on resource prediction model based on kubernetes container auto-scaling technology. In IOP Conference Series: Materials Science and Engineering (Vol. 569, No. 5, p. 052092). IOP Publishing. https://iopscience.iop.org/article/10.1088/1757-899X/569/5/052092/pdf

[12]. Zheng, W., Tynes, M., Gorelick, H., Mao, Y., Cheng, L., & Hou, Y. (2019, August). Flowcon: Elastic flow configuration for containerized deep learning applications. In Proceedings of the 48th International Conference on Parallel Processing (pp. 1-10). https://doras.dcu.ie/24290/1/main.pdf