Auto-Scaling Anomaly Detection in Cloud Computing Models using MAPE-K Loop

Nikhil Mahesh
Department of Computer Science and Engineering,
Amrita School of Computing,
Amrita Vishwa Vidyapeetham, Bengaluru, India
bl.en.p2dsc22003@bl.students.amrita.edu

Supriya M.
Department of Computer Science and Engineering,
Amrita School of Computing,
Amrita Vishwa Vidyapeetham, Bengaluru, India
m_supriya@blr.amrita.edu

Abstract: Computing paradigms are changing at a faster pace leading to highly complex and dynamic environments, thereby demanding resource management acumen. Pivoting in these paradigms, auto scaling ensures optimal resource allocation, sufficient to accommodate variable demands. However, the dynamic nature of such settings poses challenges in maintaining reliability and efficiency in auto-scaling methodologies. This paper proposal solves this challenge through a dependable anomaly detection system intended for auto-scaling in cloud computing frameworks. This work tries to design, develop, and validate the framework of anomaly detection for resource consumption and workload patterns using a MAPE-K loop. The envisioned framework is meant to use machine learning techniques and methodologies of data analysis to track in real time system parameters for the early identification of potential scalability issues. In terms of scores for all metrics, it has performed better with the MAPE-K loop, and the R2 score for all metrics came out to be 1.

***Keywords**: Auto Scaling, Anomaly Detection, ARIMA, Isolation Forest, MAPE-K*

I. INTRODUCTION

Over the past few years of fast digital transformation, cloud computing frameworks have proven to be critical architecture for enterprises in employing technologies around data for maximum potential. It comes with dynamic, scalable, and widely distributed computing environments supporting deployment of applications and services close to the source of data. In this way, latency is reduced, response is improved, and resource utilization is optimal. As these computer systems get bigger and more complex, it becomes harder to manage resources efficiently with autoscaling.

Auto scaling for application and service expansion will likely be one of the most significant features of cloud computing. It should be such that, in the face of changing workloads, more computing resources are added automatically when the workload is heavy, and resources are reduced when the workload is light. Adaptability ensures the alignment of computational capacities with the demand at a particular time, preventing any possible bottlenecks in resources and keeping the cost of operation low. However, orchestrating this provision of resources to workloads is anything but a trivial task, yet very promising.

The dynamism in cloud environments, the variety of resources, the complications of distributed computing, and workload variations in the cloud make auto scaling algorithms a bit difficult. Effectively, hence, scalability in such contexts requires an understanding of system behavior, responsiveness to changes in conditions, and the capability of detecting anomalies that would disrupt the system's stability. This paper aims to deal with these problems by developing state-of-the-art auto-scaling, resource management, and improving resiliency, responsiveness, and efficiency of a computing infrastructure by implementing a robust anomaly detection system customized to cloud computing environments. Detailed, in the subsequent sections of this proposal are the existing research, project's objectives, methodologies, anticipated outcomes and the significance of its findings in the context of contemporary computing paradigms.

II. LITERATURE SURVEY

This section discusses the papers related to Auto Scaling and Anomaly detection techniques.

A survey on techniques of autoscaling in container-based cloud and edge/fog computing analyzed different autoscaling mechanisms, classified different autoscaling architectures, and came up with a comprehensive taxonomy and survey of autoscaling solutions for container-based virtualization in cloud and edge/fog computing applications. Besides, the authors have also pointed out some of the challenges and limitations of auto-scaling in such environments, providing room for further research. The overall idea was to get a better understanding of auto-scaling techniques, possibly applied on cloud and edge/fog computing [1].

The article, "Online machine learning for auto-scaling in edge computing" focuses on implementing the method of online machine learning to predict the future workloads with the use of MAPE-K to decide on scaling actions. It also introduces an auto-scaling policy agent based on safe reinforcement learning (RL), which quickly responds to traffic changes to ensure the latency requirements of services. The proposed agent learns the reward associated with the multi-access edge computing (MEC) environment by the Q-learning algorithm. The MEC environment and auto-scaling decision-making are modeled using Markov Decision Process (MDP). The authors evaluate this in comparison with auto-scaling mechanisms that already exist through the means of simulations of edge environments. The experimental results reveal that the proposed approach enhances scalability, latency, and resource utilization in comparison to the other mechanisms [2].

The authors of the paper [3] state the fact that designing an auto-scaling algorithm for the cloud proves to be an uphill task because one lacks the prior knowledge of the cloud environment. In their recommendations, the authors confirmed that RL-based approaches have been proved to work in highly dynamic environments when trying to handle this challenge. Performance evaluation of SpotRL and comparisons with other state-of-the-art auto-scaling algorithms were conducted by the authors through simulation experiments. The reported results indicate that the proposed SpotRL can dramatically reduce the deployment cost for SaaS providers while the servicing availability is high. In summary, the authors find that SpotRL can help SaaS providers in enhancing their overall revenue through lowering their services' deployment cost in compliance with certain SLAs.

In the study [4], the authors propose anomaly detection for micro-services architecture that uses temporal and spatial data analysis. Authors developed a two-stage validation framework that targets micro services architecture to detect anomalies using logs and service query traces. The first level is the prediction of the deviation between normal expectations and real outputs, and the corresponding calculation of the degree of anomalies. The second level is responsible for the interpretation of the representative features of behaviors and queries to reveal the hidden systematic status. This enables the fusion of two representations to be followed by another anomaly detector based on one-class classifier, which can learn insights into empirical data. The proposed scheme was evaluated in a real-world dataset of a 5G mobile system and outperformed the methods in accuracy and efficiency, respectively. Authors have also stated that this approach could be utilized in a real-world situation, including security monitoring.

The work presented in [5] implemented auto-scaling and self-healing in Kubernetes. Besides, is also implemented the various methods of anomaly detection, such as moving average, exponential smoothing, seasonal decomposition of time series, isolation forest, one-class support vector machine, autoencoder, and long short-term memory, beside the web application service with the help of K8s. Here, nine types of abnormality were injected that degraded the web application's performance and executed the concepts related to auto-scaling and self-healing. The delays and performance of these anomaly detection methods were also verified by the collected metrics. The authors presented that the performance of anomaly detection methods vary; LSTM is the best performing method in the dataset with its experiments. Anomaly scores of LSTM can represent the degree of abnormality. The authors concluded that their analysis offers useful insights for operators to manage Kubernetes with auto-scaling and self-healing.

A novel AIOps solution to predict node failures in a large-scale cloud computing platform proposed in [6], conducted an exploratory study on the characteristics of alert data. They provided an outline for their AIOps solution, explained its implementation details, and presented their evaluation results. Besides, they shared their lessons learned for developing and deploying the AIOps pipeline, in the hope that it would be of immense help to researchers and practitioners in ensuring their AIOps solutions are adopted.

Our proposed approach is to detect the anomaly in the cloud model using MAPE-K loop explained in the next section. The implementation uses ARIMA and Isolation Forest models in the analysis phase of MAPE-K loop to detect the anomalies.

III. Dataset Description

With a large dataset that would include CPU utilization, Network packet in/out, Network in/out, CPU credit utilization, and AWS CloudWatch balances [7], one would be able to find the anomaly. Further to this is predefining the thresholds in the upper and lower limits for these parameters in a manner that reflects the normal behavior spectrum. In this manner, the process of detecting anomalies would focus on those caused by deviations from the benchmark. For instance, whenever there are metrics that surpass or inconsistencies in the performance metrics, further investigation or corrective measures are triggered.

This will enable setting upper and lower bounds for each metric, which will help in identifying sudden fluctuations and therefore assist in detecting problems in a timely manner. Analysis of the behavior of the system in totality and the detection of an anomaly in its various facets are done simultaneously. It is a proactive methodology that in turn results in the rapid detection of problems, hence minimizing downtime and maintaining high levels of system performance.

It also offers a structured framework for anomaly detection with automated alerts or triggers if there are deviations from the normal which are beyond the acceptable limits. This methodology is proactive, and the detection of problems is rapid, which in turn leads to minimized downtime and maintained high system performance. A sample dataset for CloudWatch is presented in Fig. 1. The metrics used for evaluation are described below.

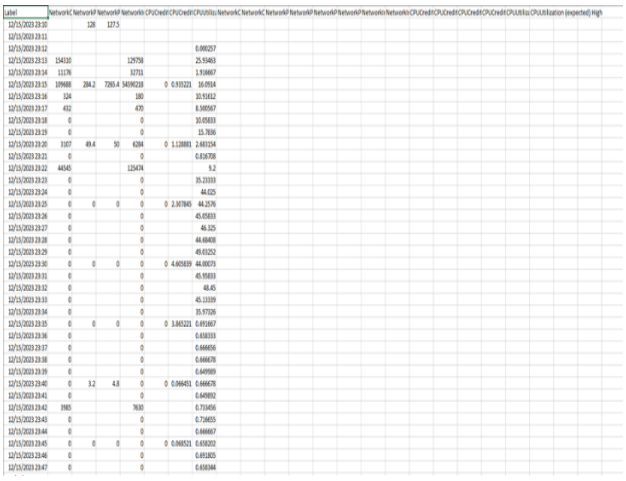

Fig 1: Dataset for Cloudwatch

A. CPU Utilization

CPU utilization is one of the important statistics for the measurement of the effectiveness and efficiency of any computer system; it is described as the time the CPU utilizes in executing instructions regarding the available time [8]. This study explores the critical role of CPU utilization in measuring system health and resource management. The overall objectives in studying various techniques of observation and optimization of the usage of CPUs are to find ways that would improve the performance of the general system while being in control of the computational demands. Properly understood CPU usage helps in identifying the source of the bottlenecks in performance and devising strategies for work and resource distribution that enhances dependability and efficiency.

B. Network PacketsIn

AWS CloudWatch "NetworkPacketsIn" is the statistic that keeps track of incoming network packets received by an Amazon EC2 instance. The statistic is the number of packets this instance has received from the network. "NetworkPacketsIn" is one of the most important metrics among all those metrics from CloudWatch, monitoring the performance and wellness of EC2 instances. This will be useful in capacity planning, performance management, and optimization of network operation, as it will be able to demonstrate the nature of incoming network traffic the instance is serving.

C. Network PacketsOut

The 'NetworkPacketsOut' metric in AWS CloudWatch tracks packets on outgoing network traffic from an Amazon EC2 instance. It simply keeps track of the number of packets that flow out of an instance and into the network. 'NetworkPacketsOut' is one of the important signals provided by CloudWatch to monitor the health and performance of the EC2 instances. This reveals much about the outgoing traffic of the network that this instance is controlling, hence helping in planning for capacity, evaluation of performance, and management efficiency of the network.

D. Network In

The AWS CloudWatch 'NetworkIn' statistic, used to measure incoming network traffic, is directed toward Amazon EC2 instances. The metric properly measures the amount of data an instance receives from the network. 'NetworkIn' is one of the most important signals that CloudWatch will monitor to determine health and performance for the EC2 instance. This provides meaningful information on the quantum of inbound network traffic this instance handles, which is important for capacity planning, performance analysis, and resource optimization.

E. Network Out

"NetworkOut" is an AWS CloudWatch metric that measures the data leaving an Amazon EC2 instance, entering an instance's network. In simple words, it is a piece of information that comes out of the instance and travels either to the internet or to some other network. Another important statistic by which one can monitor the performance and behavior of CloudWatch is "NetworkOut". This allows understanding, for example, how much of the data being transferred from an instance is real; this is an essential parameter for the measurement of network utilization and optimization of data transfer, which will consequently lead to a cost reduction in data transfer out. Through the monitoring of "NetworkOut," outbound data flow analysis can be performed; bottlenecks, for example, can be pinpointed, and decisions can be made regarding the network settings and how to optimally load instances for data transmission efficiency.

F. CPU Credit Usage

The 'CPUCreditUsage' in an amazon EC2 instance uses many CPU credits, which becomes important for the case of instance types being T2, T3, and T3a under burstable performance [9]. The metric 'CPUCreditUsage' is one among the metrics coming from CloudWatch for the monitoring of EC2 instances, such as the various burstable types of instances. It provides a deeper understanding of how CPU credits are consumed by the instance, the performance characteristics, and how the instance is operating in terms of its CPU credit allocation. This metric enables perfect management of the burstable instance toward success. It places one in a position to know if their workloads are consuming more CPU credits over a certain time amount that is permitted. Such analysis helps optimize workloads in a way that maintains a balance between performance needs in workloads and is cost-effective.

G. CPU Credit Balance

The CPU credits are the unit of measurement made for handling and managing CPU utilization by the AWS EC2 instances under burstable performance instances, for example, T2, T3, and T3a. In simple terms, a CPU credit balance indicates how many credits were able to collect over a specified period after falling below baseline CPU utilization. The instance uses these accrued credits to increase the CPU performance on a temporary basis so that it can handle a larger load when the baseline is exceeded. This balance reflects the credit available to an instance that it can dip into when there is a need for more processing power. Monitoring the balance of CPU credits is quite important, since that reflects the capacity, an instance can have towards absorbing workload bursts. If the balance is low, then it can handle only small workload surges; a high balance will carry higher burstable performance. Careful management of the CPU credit level should be exercised to avoid performance degradation and the risk of depletion of CPU credit, potentially resulting in CPU performance limits until credits are restored.

IV. Anomaly Detection using MAPE-K

The proposed model uses AWS CloudWatch [7] with the integration of MAPE-K loop. This helps in forecasting as well as detecting anomaly within 180 minutes. It also provides a scalable and efficient solution for real-time monitoring and analysis of system metrics. The model can easily handle large amounts of data and adapt to forecast using an anomaly detection algorithm. This helps identify potential issues in a timely manner, allowing for proactive measures and improving the system's performance.

The proposed architecture as illustrated in Fig. 2, presents the steps for the flow of the process using MAPE-K loop in the system. The implementation uses the MAPE-K loop that

consists of Monitor, Analyze, Plan, Execute and Knowledge. It is a framework that is designed to manage and enhance system performance. It initially monitors its operations by collecting data continuously on all measurable features of performance and performs data analysis to determine

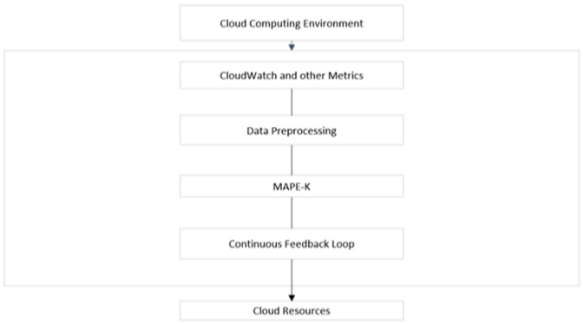

Fig 2: Architecture diagram

different patterns and tendencies and anomalies that may portend problems that arise. It then plans the appropriate actions based on the analysis for detected anomalies or optimization of performance. The performing phase does those planned actions toward placing the system in order or in better shape. Finally, the understanding component is the one that uses the insight of this process to guide and hone what will be done in future cycles of monitoring, analysis, planning, and execution. By repeating these, it continues developing and becoming resilient, able to deter problems before they happen [10]. The diagrammatic representation of the MAPE-K loop is presented in Fig. 3.

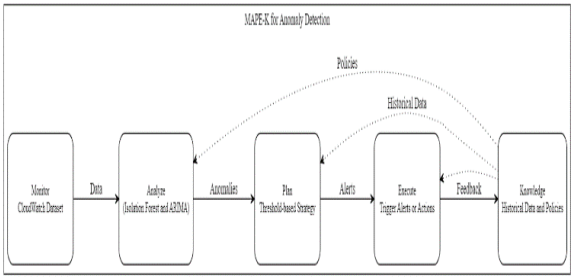

Fig 3: Diagram For MAPE-K

The analyze phase of MAPE-K loop uses two machine learning models namely ARIMA and Isolation Forest for anomaly detection which is described below:

A. ARIMA

Autoregressive Integrated Moving Average, or simply ARIMA, is a very powerful model used widely in time series data analysis and forecasting [10]. The model is made up of the moving average (MA), differencing (I), and autoregressive (AR) components of the temporal structure of the data. The goal of ARIMA is understanding and forecasting of the trends in a time series. This is put into practice by investigating the pattern and the relationship between an observation and its lag values, differencing for stationarity, and the introduction of terms of moving averages to account for the earlier forecast errors. Fundamentally, the process of differencing in ARIMA models allows the complex picture of patterns in time series to be described graphically, even when stationarity does not exist. ARIMA models have found extensive use in many forecasting applications across a wide range of disciplines and are particularly useful because they have a very broad scope of usage.

B. Isolation Forest

Anomaly detection algorithm developed based on the intuition that anomalies are few and different. In contrast to more traditional distance- or density-based approaches, Isolation Forest isolates observations by randomly selecting a feature and then selecting a random split value between the minimum and maximum of the selected feature [11]. Since, due to its time complexity, this algorithm is quite effective and can treat high-dimensional data, it is convenient to be used with large datasets. An isolation forest provides an accurate measurement of the degree of abnormality that data points have and effectively supports different applications [12-15].

Anomaly detection for various metrics like CPU usages, Network usages etc. have been applied for AWS CloudWatch.

On successful implementation, the anomaly detection in the CPU utilization using AWS CloudWatch (Fig 4) identifies 452 anomaly points within the expected range.

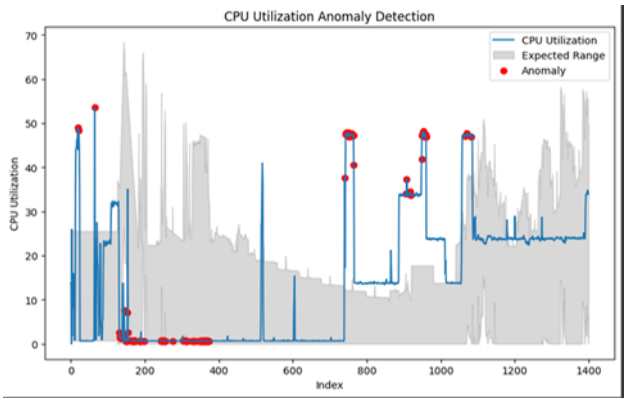

Fig 4: CPU Utilization

As per Fig. 5, isolation forest detects 14 anomaly points for the metric NetworkIn in the expected range.

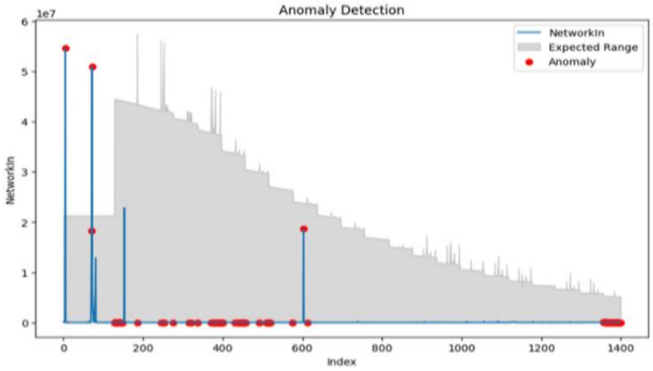

Fig 5: NetworkIn

Similarly, the metrics NetworkOut, NetworkPacketIn, NetworkPacketOut, CPU credit usage, and CPU credit balance detects 187, 1400, 1320, 814, and 994 anomaly points in the identified ranges as presented in Figs. 6 to 10 respectively.

V. Performance Measures

The proposed approach is evaluated using the standard measures which are described below.

A. RMSE

Root Mean Square Error is one of the most frequently used assessment measures in machine learning regression [15]. Root Mean Square Error is the square root of the actual minus predicted values [16]. RMSE value is inversely related to the model's performance; a lower RMSE value demonstrates that the model better follows the data.

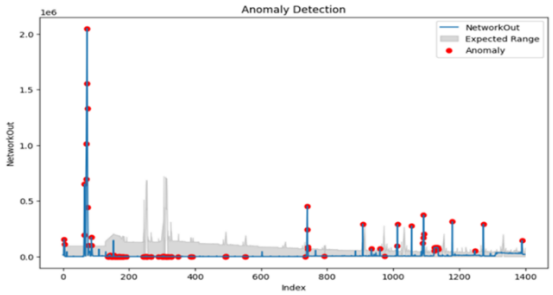

Fig 6: NetworkOut

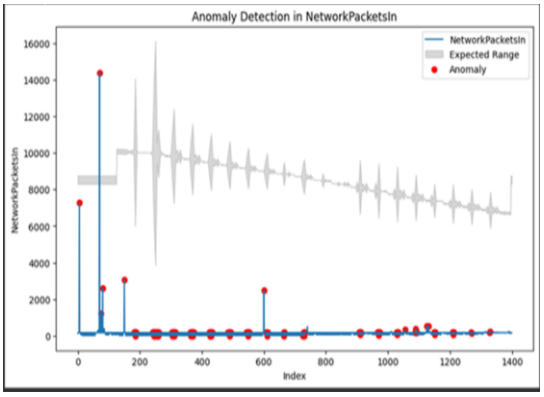

Fig 7 : NetworkPacketsIn

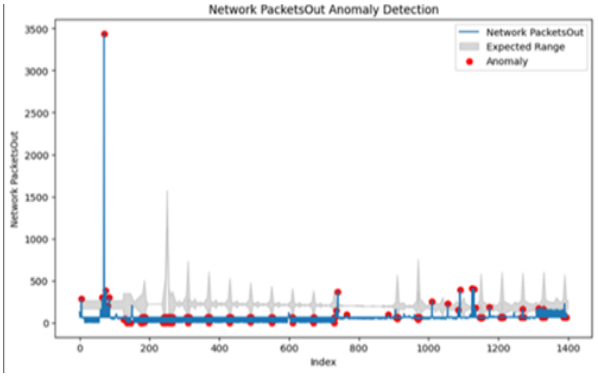

Fig 8: NetworkPacketsOut

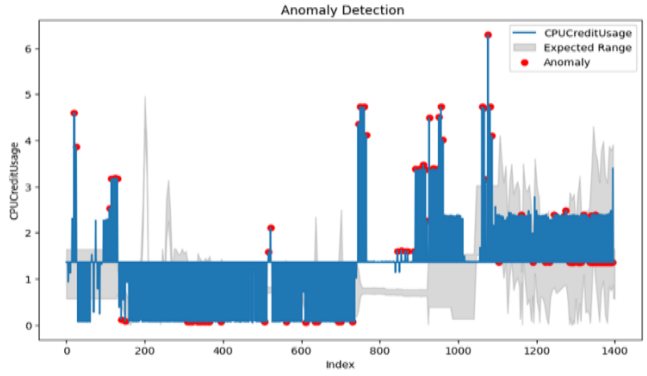

Fig 9: CPU Credit Usage

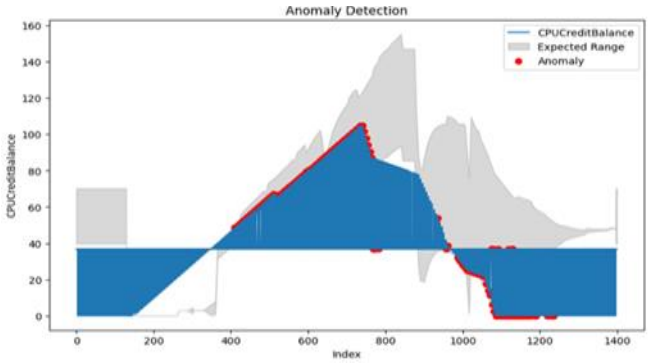

Fig 10: CPU Credit Balance

B. MSE

Mean Square Error is the mean of the squares of the variances. In machine learning, this can be seen as the most common criteria in machine-learned regression assignments [17]. A lower MSE indicates a stronger correlation between model and data.

C. MAE

Another frequently used evaluation in machine learning for regression problems is Mean Absolute Error. The MAE is an average absolute difference between the predicted and the real values [18]. Since MAE error metric is indifferent to outliers, it is a more robust metric for the assessment of the model's performance compared to MSE.

D. R-Square

R-Square is an important measure when it comes to regression analysis, offering insights about the link between dependent and independent variables. It measures how well a model fits the observed data. If R-squared is 1, then all the variability in dependent variable has been explained by the independent variables [19]. A lower R-squared statistic means that not all variation in dependent variable can be accounted for by the independent variables, thus indicating a weaker fit [20].

VI. RESULTS

Table 1 presents the evaluation of the anomaly detection system, against the identified performance metrics described in the previous section. This demonstrates the system's capability to understand difference between operational states and potential issues, ensuring that only genuine anomalies are flagged for further checking. The score for seven different metrics of Anomaly detection has been performed, making the evaluation of overall performance for the forecasting model in view of the detected anomalies possible for both MSE and MAE, which measure average squared and absolute differences of predicted and actual values, respectively. This will help them understand what kind of anomalies might have taken effect and therefore introduce refinements or adjustments to the model that becomes more adaptive to unexpected events. RMSE is another member of the MSE family and therefore provides one more typical magnitude measure of error in the forecasts. After anomaly detection procedure, integration of RMSE quantifies variability and consistency in observing forecast errors to allow elaboration further on how a model performs

under anomalous conditions. R-Square value in every metric is found to be 1 that indicates the model perfectly predicts the variations.

Table 1: Evaluation metrics

Metrics	MSE	RMSE	MAE	R2
CPU Utilization	0.68	0.82	0.78	1
NetworkOut	0.007	0.087	0.045	1
NetworkIn	0.017	0.13	0.106	1
CPU Credit Balance	0.045	0.213	0.170	1
CPU Credit Usage	2.07	1.44	1.26	1
Network PacketsIn	0.02	0.16	0.08	1
Network PacketsOut	0.02	0.16	0.08	1

VII. Conclusion

This proposed work aims to explore the effective use of predictive and reactive techniques in auto-scaling anomaly detection. This study's reactive strategy includes a proactive step to address anomalies as they arise, providing a thorough plan for identifying and resolving issues beyond just predicting them. The MAPE-K (Monitor, Analyze, Plan, Execute, Knowledge) loop used in the study supports the reactive strategy by using acquired knowledge to improve future responses. By anticipating anomalies and implementing preventive measures, systems can become much more resilient and effective. These combined achievements mark a significant advancement toward reliable, efficient, and safe computer systems across various applications. The insights from this study lay a strong foundation for future innovations in anomaly detection techniques and procedures, driven by technological growth and the increasing importance of real-time data processing and analytics. In future, The MAPE-K loop could be experiments with additional ML models and forecasting could be done using methods such as SARIMA and Prophet.

References

[1] Dogani, Javad & Namvar, Reza & Khunjush, Farshad. Auto-scaling techniques in container-based cloud and edge/fog computing: Taxonomy and survey. Computer Communications. 209. 10.1016/j.comcom, 2023,.06.010.

[2] Thiago Pereira da Silva, Aluizio Rocha Neto, Thais Vasconcelos Batista, Flávia C. Delicato, Paulo F. Pires, Frederico Lopes, Online machine learning for auto-scaling in the edge computing, Pervasive and Mobile Computing, Volume 87, 2022, 101722, ISSN 1574-1192, https://doi.org/10.1016/j.pmcj.2022.101722.

[3] Devagopal AM, Ashwin V, Vishal Menon, "Prediction of Water Quality Parameters of River Periyar Using Regression Models," 2nd International Conference on Advance Computing and Innovative Technologies in Engineering, Greater Noida, India, 2022, pp. 53-57.

[4] G. V. Sajan, P. Kumar, "Forecasting and Analysis of Train Delays and Impact of Weather Data using Machine Learning," 12th Int. Conf. on Computing Comm. & Networking Technologies, India, 2021, pp. 1-8,

[5] S. Sonu, A. Suyampulingam, "Linear Regression Based Air Quality Data Analysis & Prediction using Python," IEEE Madras Section Conf., India, 2021, pp. 1-7.

[6] C. -I. Chang, "Target-to-Anomaly Conversion for Hyperspectral Anomaly Detection," in IEEE Transactions on Geoscience and Remote Sensing, vol. 60, pp. 1-28, 2022, Art no. 5540428, doi: 10.1109/TGRS.2022.3211696.

[7] AWS, https://aws.amazon.com/cloudwatch/"

[8] Singh, A., Singh, V., Aggarwal, A. (2024). Improving the Application Performance by Auto-Scaling of Microservices in a Containerized Environment in High Volumed Real-Time Transaction System. In: Bhardwaj, A., Pandey, P.M., Misra, A. (eds) Optimization of Production and Industrial Systems. CPIE 2023. Lecture Notes in Mechanical Engineering. Springer, Singapore. https://doi.org/10.1007/978-981-99-8343-8_27.

[9] R. Doukha, S. A. Mahmoudi, M. Zbakh and P. Manneback, "A comparative analysis of Convolution Neural Network models on Amazon Cloud Service," AIIPCC 2022; The Third International Conference on Artificial Intelligence, Information Processing and Cloud Computing, Online, 2022, pp. 1-7.

[10] Malburg, Lukas & Hoffmann, Maximilian & Bergmann, Ralph. Applying MAPE-K control loops for adaptive workflow management in smart factories. Journal of Intelligent Information Systems. 61, 2023. 1-29. 10.1007/s10844-022-00766-w.

[11] A. Ashok and C. P. Prathibhamol, "Improved Analysis of Stock Market Prediction: (ARIMA-LSTM-SMP)," 2021 4th Biennial Int. Conference on Nascent Technologies in Engineering (ICNTE), India, 2021, pp. 1-5,

[12] A. Garlapati, D. R. Krishna, K. Garlapati, N. m. Srikara Yaswanth, U. Rahul and G. Narayanan, "Stock Price Prediction Using Facebook Prophet and Arima Models," 2021 6th International Conference for Convergence in Technology (I2CT), Maharashtra, India, 2021, pp. 1-7, doi: 10.1109/I2CT51068.2021.9418057.

[13] R. R and S. Babu, "Anomaly Detection using User Entity Behavior Analytics and Data Visualization," 2021 8th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 2021, pp. 842-847.

[14] T. Lalitha, N. K. Anushkannan, S. Shreepad, S. Sasireka, H. Anandaram and S. Razia, "Deep Learning-based Automatic 3D Printer Anomaly Detection during the Printing Process," 2022 3rd International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 2022, pp. 1343-1348, doi: 10.1109/ICOSEC54921.2022.9951903.

[15] A. Saji, A. Prakash and M. S. Krishnan, "Electricity Demand Forecasting In Kerala Using Machine Learning Models," 2023 3rd International Conference on Intelligent Technologies (CONIT), Hubli, India, 2023, pp. 1-6, doi: 10.1109/CONIT59222.2023.10205925.

[16] S. RamPrakash and P. Sivraj, "Performance Comparison of FCN, LSTM and GRU for State of Charge Estimation," 2022 3rd International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 2022, pp. 47-52, doi: 10.1109/ICOSEC54921.2022.9951916.

[17] Reena Panwar, M. Supriya, Dynamic resource provisioning for service-based cloud applications: A Bayesian learning approach, Journal of Parallel and Distributed Computing,Volume 168, 2022, Pages 90-107, ISSN 0743-7315

[18] H. Paul, A. Rajendran, C. Sunil, A. T and S. Abhishek, "CycleFit: An Analysis of Regression Models for Caloric Expenditure Prediction in Cycling Activities," 2024 IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI), Gwalior, India, 2024, pp. 1-6, doi: 10.1109/IATMSI60426.2024.10503071.

[19] A. M. Almohammedi, A. Zerguine, M. Deriche and S. M. Sait, "Artificial Bee Colony DLMS Beyond Mean Square Error Boundary in Ad-hoc WSN," 2022 14th International Conference on Computational Intelligence and Communication Networks (CICN), Al-Khobar, Saudi Arabia, 2022, pp. 572-576, doi: 10.1109/CICN56167.2022.10008253.

[20] Shinu, M.R., Supriya, M. (2021). Performance Comparison of VM Allocation and Selection Policies in an Integrated Fog-Cloud Environment. In: Kumar, N., Vinodhini, M., Venkatesha Prasad, R.R. (eds) Ubiquitous Communications and Network Computing. UBICNET 2021.