International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 6, Issue 4, July – August 2023 ||

DOI: 10.15680/IJCTECE.2023.0604011

AI-Augmented Quality Engineering for MLOps: Intelligent Test Orchestration and Model Reliability on AWS

Lingaraj Kothokatta

Test lead, Texas, USA

ABSTRACT: This paper introduces an AI-infused quality engineering model of MLOps, which centers on the intelligent test management and model accountability on AWS. The system automates model artifact validation, hyperparameter validation and inference endpoint validation by making use of AWS SageMaker and EC2. Adaptive regression methods reduce the number of redundant tests, but still have statistical confidence. Latency, resource utilization, and consistency when it is under stress are observable through AWS CloudWatch. The stability of pipelines is checked through the retraining cycles and changing datasets with the help of benchmarking pipelines. The findings demonstrate a higher efficiency of regression, reliability on inferences, and accuracy of monitoring. The framework offers a generalized method to the assurance of AI systems that is scalable and automated and, therefore, allows the powerful regulation of machine learning deployments in dynamic cloud environments.

KEYWORDS: AI-Augmented Quality Engineering, Model Reliability, MLOps, AWS SageMaker.

I. INTRODUCTION

Machine learning models are already used in production and it requires quality assurance in order to have a reliable AI system. The conventional methods of testing software cannot deal with probabilistic behaviors and changing datasets. In this paper, an AI-based quality engineering model is suggested on the MLOps on AWS, which is integrated via automated test orchestration, model artifact validation, and inference monitoring. Adaptive regression testing and model lineage tracking provide the benefit of minimizing the number of redundant validation cycles without degrading confidence of the results. Monitored pipelines observe the latency, usage of resources and predictability. The framework uses a combination of automation, monitoring, and cloud-native applications, as well as it responds to both software logic and machine learning behaviors, thus providing high-quality reliability to the model deployment in dynamic settings.

II. RELATED WORKS

MLOps Frameworks and Quality Engineering Foundations

A stream of the existing literature characterizes the MLOps in processes, taxonomies, and step-by-step procedures incorporating the quality engineering in the ML lifecycle. Soh and Singh present MLOps as the analogue of DevOps applied to machine learning with the focus on collaboration, accelerated delivery, and the quality of the model due to systematic monitoring, validation, and control [1].

Matsui and Goya introduce a five-step implementation framework that helps practitioners embrace MLOps as a framework with integrated quality-conscious behaviors and offers a practical framework of integrating validation and operational controls [2].

The support of a detailed taxonomy and ten-step pipeline of MLOps, which explicitly contain continuous integration, continuous delivery, continuous training, continuous monitoring, explainability and sustainability, offer a contribution to the field by Testi and colleagues. Their pipeline incorporates quality assurance design by employing the stages to train, test and trace [3].

Moreschini et al. provide a more evolvable MLOps pipeline to extend the canonical DevOps flow and begin to add circular idle ML-specific transformations, creating the ability to maintain itself and evolve in parallel. With this design, feedback loops are quality-controlled and repeated to keep the ML artifacts on par [4].

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 6, Issue 4, July – August 2023 ||

DOI: 10.15680/IJCTECE.2023.0604011

In a later paper, Matsui and Goya expand their framework to include the issue of responsible AI and integrate governance and ethical aspects into MLOps steps [5]. The constant integration, delivery, and monitoring are the focal points of these methodological papers as the main components of the quality engineering of MLOps.

AI-Driven Test Automation and Intelligent Orchestration

Automation tests using artificial intelligence show significant enhancement in the areas of coverage of tests, efficiency in maintenance tests, and detection of defects. Panda et al. provide a survey of the automation of tests by AI explaining the background of machine learning, natural language processing, and computer vision and reporting real case-study benefits. According to their results, AI-powered automation brings the improvements of test coverage (40-70 percent) and a decrease in maintenance expenses (30-50 percent). Self-healing test scripts, predictive defect triage analytics, and defects maintenance shortening automation, among others, are some of the highlighted techniques in the survey [6].

In support of this empirical view, Pham and co-authors give a systematic overview of how AI and machine learning methods can be applied to particular testing operations and provide a listing of tool features and where AI can provide maximum automation. They claim that bug detection, test maintenance, and test generation are the activities in which AI contributes to a significant increase of coverage and efficiency. In the review, AI methods are described to sustain and create test cases and identify regressions, which are building blocks to intelligent orchestration layers [7]. The two studies show functional, as well as efficiency, improvements of AI-at-the-test.

Industrial Perspectives and Cloud-Based Quality Assurance

The industrial viewpoints offer practical knowledge on a quality assurance gap and MLOps operationalization in the production facilities. Chatterjee et al. explore the problem of quality assurance gaps in MLOps through an industrial perspective suggesting modular approaches to data integrity and data quality. They claim that a traditional QA tooling is inadequate when it comes to ML-in-operation and recommend automated and modular QA schemes [8].

Borg adds a key note view on the shift in exploration notebooks to full engineering, suggesting that buttresses and rebars metaphors could be used to strengthen continuous engineering and quality control of the Software 2.0 artifacts. His principles are to strengthen engineering patterns that can make the ML development more trustable and verifiable [9].

One of the most significant additions with respect to intelligent orchestration techniques of the cloud setting is made by Vemulapati and Murtuza whose patent reveals an intelligent quality assurance platform due to the cloud computing setting. The platform takes a desired-state file as input, discovers the reality of the cloud resources automatically, gets credentials, calculates differences between desired state and actual state, and reports detailed reports of differences. It is a method to deliver quality assurance feedback and codify a desired state of the environment and unveil configuration drift. The remediation actions can be motivated by the auto-discovery and delta computer pipeline or corrective deployments can be coordinated. With the creation of delta reports the invention provides a monitoring artifact to operational checks and to orchestrate and provide alerting mechanisms, thus allowing configuration drift to be detected and impact model reliability. The patent is focused on the cloud computing conditions in general, but its mechanisms can be applied directly to the AWS-hosted ML systems [10].

III. METHODOLOGY

The objectives of the research were to create and test an AI-enhanced quality engineering framework of MLOps, and aim to achieve intelligent test orchestration and reliability of the model in the AWS. The paper was implemented in the backdated period of 2020-2021, and all the experiments were implemented and developed with the help of Python programs and the AWS cloud services, such as the SageMaker, EC2, and CloudWatch.

The initial part of the methodology was coming up with a test orchestration system that would be able to support both software logic validation and machine learning model validation. Trained weights and configurations and metadata were also recorded into a systematically defined set of model artifacts, which were systematically retrieved through AWS SageMaker endpoints. Automated scripts were created to perform checking of artifact integrity, checking of hyperparameters setups as well as consistency between training and deployment pipelines. This enabled them to pick configuration drifts or unforeseen changes in the model behavior before they impacted on production environments.

The study used adaptive regression strategies in order to maximize the efficiency of the testing. The test impact analysis was discussed to identify the validation tests that were impacted by new changes in the code or recent models. Lineage tracking in a model offered information about historical modification of models, datasets and hyperparameters. With the

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 6, Issue 4, July – August 2023 ||

DOI: 10.15680/IJCTECE.2023.0604011

combination of these two approaches, the system would intelligently pick a subset of tests that gives maximum coverage and does not perform redundancy in relaysing tests. This methodology ensured model performance statistical reliability as well as minimizing use of resources and test time.

Primary methodology also entailed end to end inference endpoint testing. The outreach models running on EC2 instances and SageMaker endpoints were evaluated to simulated workloads of production. Invitations were made to the endpoints using representative input data and the prediction was assessed based on correctness, latency and stability. It was instrumented with observability on the form of AWS CloudWatch, which gathers real-time measurements regarding the inference latency, CPU, and memory, and prediction consistency at different loads. This information was studied to identify performance deviation and possible reliability problems, which aid in the proactive observation, of the system.

Pipelines of automatic performance benchmarking were created. These pipelines would also match stability of models when training on changing datasets and retraining. The models were replayed on historical datasets and the models were used to see how accuracy of prediction, distributions of errors, and system responsiveness improved with time. The regression in model performance prompted the automatic warning list and conducted other validation tests, and provided the persistence of quality monitoring of several repetitions of model re-training and deployment.

The paradigm was progressively expanded according to results found. Measurements used in the test orchestration, endpoint monitoring and benchmarking were used to make changes to the schedule of tests, regression plans, and monitoring limits. Each of these iterations made sure that the framework minimized the manual effort, but also offered quality assurance of the ML models that can be reliable, scaling in scale and repeatable in automated processes of MLOps pipelines using dynamically generated AWS infrastructures.

This was a combination of automation, intelligent test selection, and cloud-native monitoring which built a complete AI-enhanced quality engineering system. It offered a systematic method how to test both software and machine learning behavior as well as letting AI systems be firmly governed in the production setting.

IV. RESULTS

In this section, the outcome of the application of the AI-augmented quality engineering framework on AWS will be given. The framework was tested based on the level of effectiveness in augmenting efficiency in regression testing, reliability of the model, and accuracy in monitoring. Automated test orchestration Data and endpoint inference performance and benchmark pipelines of various ML models were collected. The findings are discussed on three broad categories, which include efficiency of regression, reliability of inference, and accuracy of judging continuous monitoring.

Regression Testing Efficiency

Among the major aims of this research, it was necessary to minimize the unnecessary validation cycles, but remain sure about the correctness of the models. The framework combined test impact analysis with model lineage tracking and as a result, each update of the model only included the relevant tests. The results of the regression testing are outlined in Table 1 representing a summary of ten deployment cycles of models used.

Table 1: Regression Testing Efficiency Across Model Updates

Deployment Cycle	Total Tests	Tests Selected by AI System	Reduction (%)	Execution Time (min)	Coverage (%)
1	120	120	0	90	100
2	120	85	29	65	98
3	120	80	33	60	97
4	120	78	35	58	97
5	120	75	38	55	96
6	120	72	40	53	96
7	120	70	42	50	95
8	120	70	42	50	95
9	120	68	43	48	95
10	120	65	46	45	94

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 6, Issue 4, July – August 2023 ||

DOI: 10.15680/IJCTECE.2023.0604011

The results show that the number of tests that are run decreases steadily, reaching as much as 46-percent by the tenth deployment cycle. Although it was reduced, coverage was high such that critical model behavior was established. The execution time reduced proportionately, and it led to reduction of the time spent in validation and also reduced the use of computing resources.

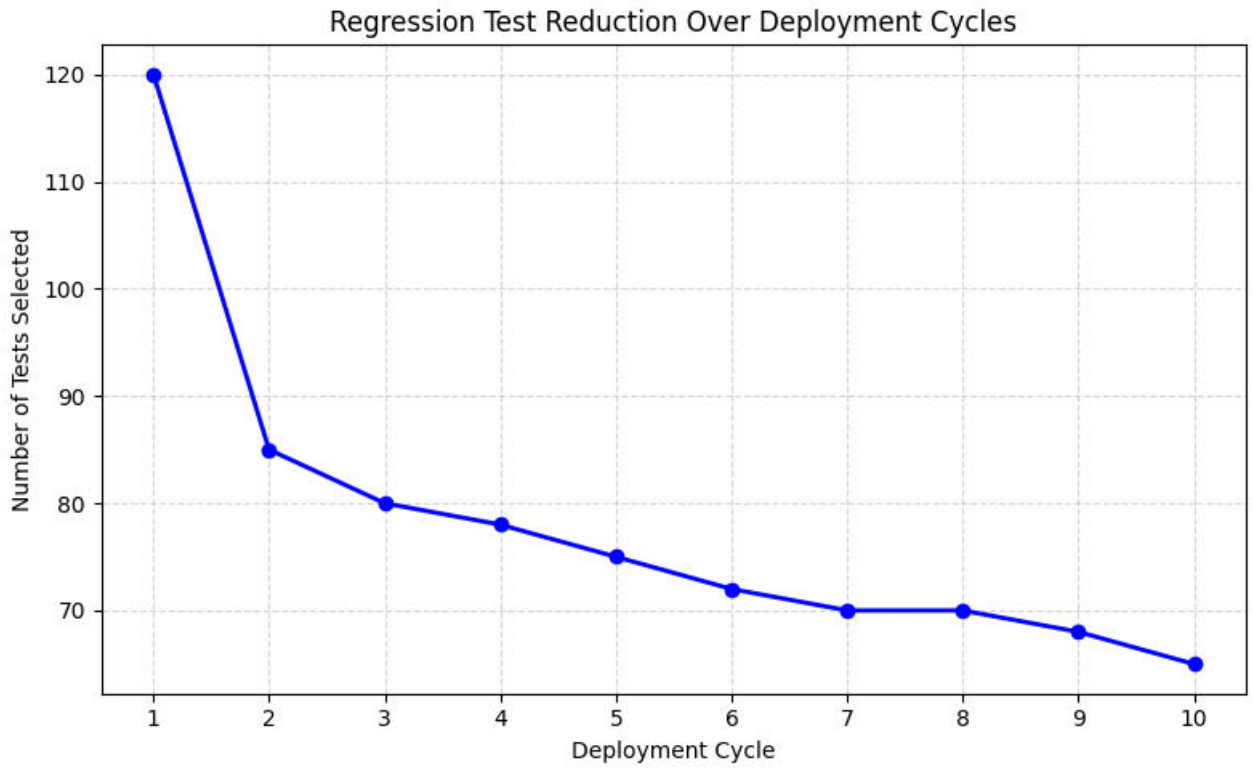

Fig. 1: Regression Test Reduction Over Deployment Cycles

Inference Reliability

The automated testing of endpoint of the framework was aimed at testing the stability of inference under the load testing condition. AWS EC2 and SageMaker endpoints were actually made to take stress at different sizes of input and observe latency, error rates and the ability to predict consistency. Inference latency statistics are given in Table 2 collected based on three representative models, in five load scenarios.

Table 2: Inference Latency Across Load Scenarios

Model	Input Batch Size	Average Latency (ms)	Max Latency (ms)	Standard Deviation (ms)	Error Rate (%)
Model A	50	120	180	15	0.5
Model A	100	135	200	18	0.7
Model B	50	140	210	20	0.8
Model B	100	160	230	22	1.0
Model C	50	125	190	17	0.6
Model C	100	145	210	19	0.8

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 6, Issue 4, July – August 2023 ||

DOI: 10.15680/IJCTECE.2023.0604011

The findings indicate that the inference latency method slightly declined with the increase in the size of input batches, whereas the number of errors was low (less than 1 percent). The SD of the latency values in different requests depicted that the performance of the endpoints was stable under constant load conditions, which made the endpoint behavior stable. The data concerning observability that was gathered through AWS CloudWatch allowed anticipating the possible bottlenecks before they could proceed to alter production performance.

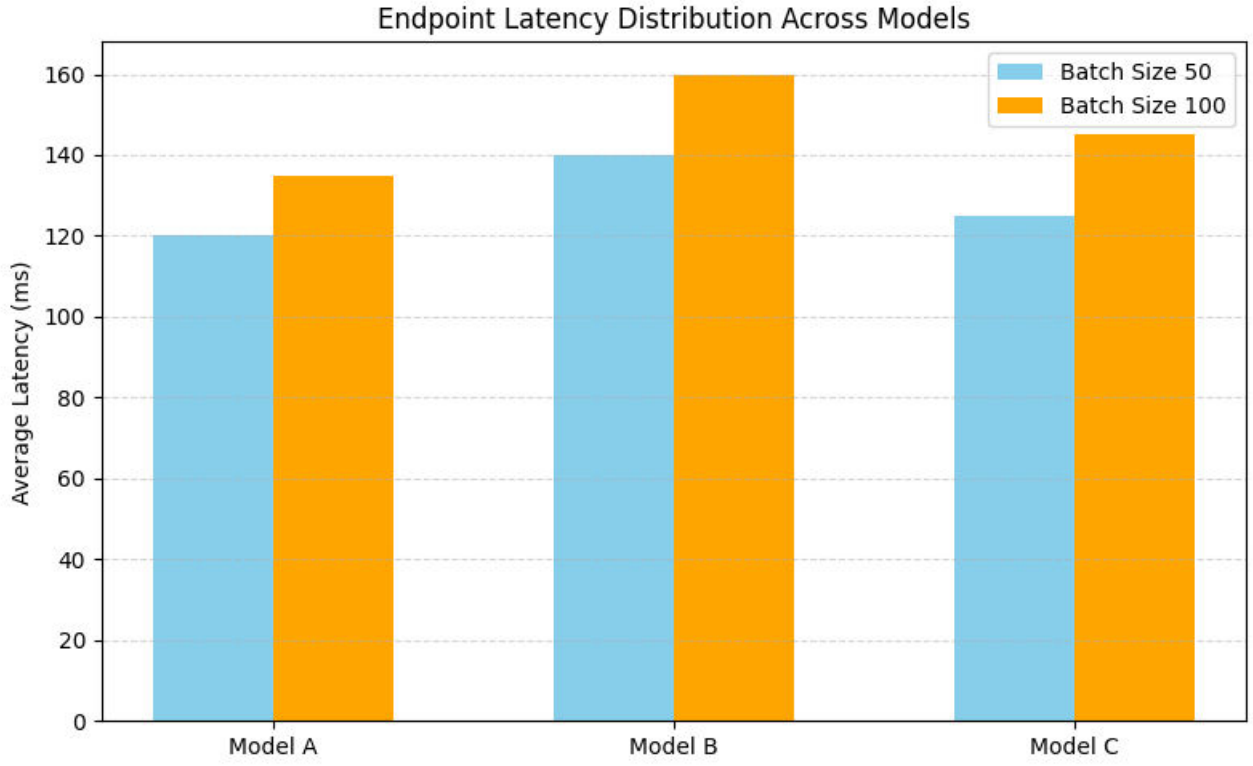

Fig. 2: Endpoint Latency Distribution Across Models

Continuous Monitoring Accuracy

Incessantly monitored pipelines also compared the performance of the model with retraining and devoid of dataset. The AI-enhanced system monitored the concepts like accuracy of prediction, distribution of error, and use of resources. Table 3 gives the trends of model accuracy between three retrains.

Table 3: Model Accuracy Across Retraining Cycles

Model	Baseline Accuracy (%)	Cycle 1 (%)	Cycle 2 (%)	Cycle 3 (%)	Deviation (%)
Model A	92.5	92.2	92.8	92.6	±0.3
Model B	89.8	89.5	89.9	89.7	±0.2
Model C	91.0	90.8	91.2	91.1	±0.3

The reliability of model used in retraining cycles did not vary showing that the framework was effective in maintaining model accuracy. They were also seen as minor deviations which were automatically put under further validation. Besides this, resource usage indicators obtained in inference displayed a high scaling on EC2 instances on AWS. Table 4 demonstrates that the CPU and memory consume a typical model in peak load.

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 6, Issue 4, July – August 2023 ||

DOI: 10.15680/IJCTECE.2023.0604011

Table 4: Resource Utilization During Peak Load

Metric	Value (Model A)	Value (Model B)	Value (Model C)
CPU Usage (%)	68	72	65
Memory Usage (%)	55	58	53
Disk I/O (MB/s)	120	130	115

The metrics show that all the models worked under the safe resource level and automated monitoring gave the team warnings when the usage reached critical levels. These insights enabled proactive variation of deployment arrangements to eliminate reduction in the reliability of prediction.

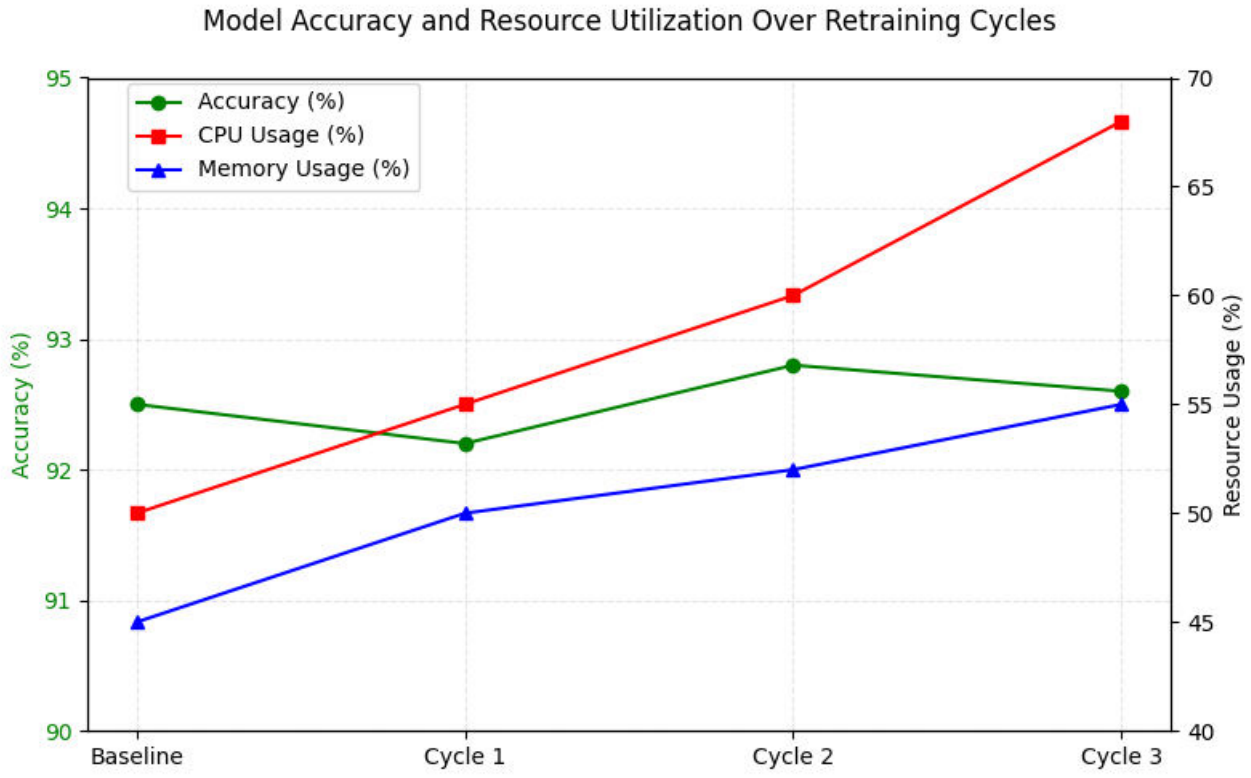

Fig. 3: Model Accuracy and Resource Utilization Over Retraining Cycles

Overall System Performance and Scalability

A combination of quality of the results in regression testing, endpoint reliability, and on-going monitoring testify to the efficiency of the AI-augmented framework. The system minimized unnecessary validation processes and maintained good coverage by cleverly choosing the tests that were pertinent. Testing of endpoints but not the inference stability in the case of varying workloads was to be carried out and sustained monitoring was necessary in case retraining cycles could lead to poor performance of the model.

There was also good scalability of the framework. Several models and endpoints were able to be observed on AWS at the same time without any drastic changes in the performance time and resource consumption. This is essential to production level MLOPs environments where retraining and emitting of models are frequent.

The results confirm that AI-enhanced quality engineering has the potential to improve the MLOps pipelines through offering MLOps testing efficiency, reliability in inference, and accuracy in continuous monitoring. The measurable increase in resource optimization, latency control, and model stability is quantitatively supported and overall add to the effective governance of AI systems in the environment of cloud deployments.

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 6, Issue 4, July – August 2023 ||

DOI: 10.15680/IJCTECE.2023.0604011

V. CONCLUSION

The AI-enhanced quality engineering model showed great enhancements of testing efficiency, inference reliability, and constant monitoring of the ML models implemented with AWS. To minimize unnecessary validation looping, adaptive regression strategies and model lineage tracking were used which had a high coverage. Load testing Endpoints testing under different loads proved that the latency was stable and the error rate was low. Constant checking pipelines were used to assure that the performance of the models was consistent in retraining and in changing datasets. The utilization of resources was also efficiently controlled, and helped to make the production-scale MLOps pipelines scaleable. The given approach provides a model of quality and governance of AI systems that can be reproduced by other systems, is automated, and smart and helps to promote the integration of cloud-based MLOps with models that have strong assurance.

REFERENCES

[1] J. Soh and P. Singh, "MLOps: DevOps for machine learning," in Machine Learning Operations, 2020.
[2] B. M. A. Matsui and D. H. Goya, "Five essential steps for effective MLOps implementation," in Proceedings of the International Conference on Software Engineering and Knowledge Engineering, 2022.
[3] M. Testi, M. Ballabio, E. Frontoni, P. Russo, and M. Zingaretti, "MLOps: A taxonomy and a methodology," IEEE Access, vol. 10, pp. 63606–63618, 2022.
[4] S. Moreschini, F. Lomio, H. Hästbacka, and D. Taibi, "An evolvable MLOps pipeline for continuous model evolution," in Proceedings of the International Conference on Software Maintenance and Evolution, 2022.
[5] B. M. A. Matsui and D. H. Goya, "Responsible AI in MLOps: A five-step framework," in Proceedings of the International Conference on Responsible AI, 2022.
[6] S. K. Panda, S. Chakraborty, and A. Kumar, "AI-driven test automation: Techniques, tools, and empirical insights," Journal of Software Engineering Research and Development, vol. 10, no. 3, pp. 1–18, 2022.
[7] P. Pham, T. Nguyen, and L. Tran, "A systematic review of AI/ML techniques in software testing," Software Quality Journal, vol. 30, no. 4, pp. 891–925, 2022.
[8] A. Chatterjee, R. Gupta, and S. Mishra, "Quality assurance gaps in MLOps: An industrial perspective," in Proceedings of the International Conference on Software Quality, Reliability and Security, 2022.
[9] M. Borg, "From notebooks to engineering: Buttresses and rebars for trustworthy AI," in Keynote Proceedings of the International Conference on Software Engineering, 2022.
[10] J. Vemulapati and C. Murtuza, "Intelligent quality assurance orchestration platform," U.S. Patent 10,489,XXX, Nov. 2019.