Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 4, 2026

SELF-OPTIMIZING CI/CD PIPELINES USING REINFORCEMENT LEARNING IN CLOUD-NATIVE ENVIRONMENTS: A HYBRID SIMULATION AND CASE-BASED PERFORMANCE ANALYSIS

Sagar Kumar[*1], Akshay Kumar[2], Ritik Kumar[3], Annas Ahmed[4], Mohammed Sohail Ahmed[5]

[*1]*Student, Northeastern University, Boston Campus, USA*
[2]*Technology Supervisor, University Library, Indiana University Indianapolis, USA; MS Applied Data Science*
[3]*MS Business Analytics, DePaul University, USA*
[4]*MS in Artificial Intelligence (Specialization in Data Analytics), Indiana Wesleyan University, USA*
[5]*Master's in Computer Information Systems (Specialization in Artificial Intelligence), Indiana Wesleyan University, USA*

[*1]sagardembla39@gmail.com, [2]akshaydembra1@gmail.com, [3]ritikhemrajani01@gmail.com, [4]annas.ahmed.1096@gmail.com, [5]sohailusa23@gmail.com

DOI: https://doi.org/10.5281/zenodo.20116269

Keywords
CI/CD, Reinforcement Learning, DevOps, Cloud Computing, Pipeline Optimization, Deep Q-Network

Article History
Received: 07 March 2026
Accepted: 14 April 2026
Published: 30 April 2026

Copyright @Author
Corresponding Author: *
Sagar Kumar

Abstract

Background: *Continuous Integration and Continuous Deployment (CI/CD) pipelines are central to modern DevOps practices, enabling rapid software delivery in cloud-native environments. However, traditional pipelines rely on static configurations that fail to adapt dynamically to changing workloads, leading to inefficiencies in deployment time, failure rates, and resource utilization.*

Objective: *This study aims to develop and evaluate a reinforcement learning (RL)-based framework for self-optimizing CI/CD pipelines using a hybrid approach combining simulated environments and real-world cloud log-based case analysis.*

Methods: *A hybrid experimental design was employed, integrating simulated CI/CD workflows with anonymized cloud deployment logs. A Deep Q-Network (DQN)-based RL agent was trained to optimize pipeline configurations, including resource allocation, test prioritization, and deployment scheduling. Performance metrics such as deployment time, failure rate, rollback frequency, and resource utilization were compared between traditional pipelines and RL-optimized pipelines using statistical analysis (independent t-test and regression modeling).*

Results: *The RL-optimized pipeline demonstrated a significant reduction in average deployment time (28.4%), failure rate (21.7%), and rollback frequency (18.9%), along with improved resource utilization efficiency ($p < 0.01$). The hybrid model showed strong generalizability across simulated and real-world datasets.*

Conclusion: *Reinforcement learning offers a robust and adaptive solution for optimizing CI/CD pipelines in cloud-native environments. The proposed hybrid framework enhances operational efficiency and reliability, providing a scalable approach for next-generation DevOps systems.*

Introduction

The rapid evolution of cloud-native architectures has transformed software development and deployment practices, with Continuous Integration and Continuous Deployment (CI/CD) pipelines becoming fundamental to DevOps ecosystems. These pipelines enable automated building, testing, and deployment of applications, facilitating faster delivery cycles and improved software quality. However, conventional CI/CD pipelines are typically rule-based and static, lacking the adaptability required to handle dynamic workloads and complex cloud environments [1,2].

Recent advancements in artificial intelligence, particularly reinforcement learning (RL), have introduced new possibilities for adaptive system optimization. RL enables agents to learn optimal decision-making strategies through continuous interaction with dynamic environments, making it particularly suitable for optimizing complex workflows such as CI/CD pipelines [3,4]. By leveraging feedback loops, RL can dynamically adjust pipeline parameters such as resource allocation, execution order, and failure handling strategies, thereby improving efficiency and reliability.

Cloud-native environments further amplify the need for intelligent pipeline optimization due to their inherent complexity, scalability demands, and variability in workloads [5]. Traditional optimization techniques often fall short in addressing these challenges, as they rely on predefined heuristics rather than adaptive learning mechanisms. In contrast, RL-based approaches provide a data-driven framework capable of continuous self-improvement [6].

Several recent studies have explored the integration of machine learning into DevOps processes, highlighting improvements in anomaly detection, predictive maintenance, and deployment optimization [7–9]. However, most existing research is limited to simulation-based models or isolated use cases, lacking a comprehensive hybrid evaluation that combines simulated environments with real-world cloud deployment data. This gap limits the generalizability and practical applicability of current solutions.

The present study addresses this limitation by proposing a hybrid framework that integrates simulation-based experimentation with case-based analysis using real-world cloud logs. This approach allows for a more robust evaluation of RL-driven CI/CD optimization across diverse operational conditions. Additionally, this study incorporates recent findings from emerging research in intelligent pipeline automation and cloud-based DevOps optimization [10–13], including selected works from Theses Journal that explore adaptive systems and AI-driven workflow improvements [1,2].

By employing a Deep Q-Network (DQN)-based reinforcement learning model, this study aims to demonstrate measurable improvements in key performance indicators such as deployment time, failure rate, rollback frequency, and resource utilization. The findings are expected to contribute to the growing body of knowledge on AI-driven DevOps and provide a scalable framework for implementing self-optimizing CI/CD pipelines in modern cloud environments.

Methodology

This study employed a **hybrid experimental design** integrating simulation-based modeling with real-world case analysis to evaluate the effectiveness of reinforcement learning (RL) in optimizing CI/CD pipelines within cloud-native environments. The hybrid approach was selected to ensure both controlled experimental evaluation and external validity through real deployment data.

The experimental framework consisted of two comparative arms: (i) conventional rule-based CI/CD pipelines, and (ii) RL-optimized pipelines utilizing a Deep Q-Network (DQN) algorithm. The simulation component was designed to replicate enterprise-level DevOps workflows, while the case-based component incorporated anonymized cloud deployment logs to validate model generalizability.

The CI/CD environment was modeled using a cloud-native architecture reflecting contemporary

DevOps practices. Pipeline orchestration was simulated using Kubernetes-based container workflows, while CI/CD execution logic was structured to mimic widely used platforms such as Jenkins and GitHub Actions. Deployment scenarios were aligned with AWS-based cloud infrastructure patterns, ensuring realistic representation of workload variability, resource constraints, and failure conditions. Monitoring parameters, including system utilization and deployment latency, were derived from simulated Prometheus-Grafana pipelines and calibrated against real-world log distributions.

The reinforcement learning agent was implemented using a Deep Q-Network architecture, selected for its suitability in sequential decision-making problems involving discrete action spaces. The state space comprised dynamic pipeline attributes, including job queue length, system resource utilization (CPU and memory), build failure frequency, and deployment latency. The action space allowed the agent to adjust key pipeline parameters, including dynamic resource allocation, prioritization of test suites, scheduling strategies (parallel versus sequential execution), and retry or rollback mechanisms.

The reward function was designed to reflect multi-objective optimization, balancing efficiency and reliability. Positive rewards were assigned for reductions in deployment time and improved resource utilization, while penalties were imposed for failed builds and rollback events. The agent was trained iteratively over multiple pipeline cycles until convergence was achieved, defined by stabilization of cumulative reward scores across successive episodes.

The study dataset comprised two components. First, a simulated dataset of 500 CI/CD pipeline runs was generated under varying workload and complexity conditions. These simulations incorporated stochastic variability to replicate real-world fluctuations in system performance. Second, a case-based dataset consisting of 200 anonymized cloud deployment logs was included to evaluate the external validity of the model. These logs represented diverse workload scenarios, including varying job sizes, test complexities, and deployment frequencies.

Outcome measures included deployment time (in minutes), failure rate (percentage of unsuccessful builds), rollback frequency, and overall resource utilization efficiency. Workload size and pipeline complexity were included as covariates to control for potential confounding effects.

Statistical analysis was conducted using standard inferential techniques. Continuous variables were expressed as mean ± standard deviation. Group comparisons between traditional and RL-optimized pipelines were performed using independent sample t-tests for normally distributed variables and Mann–Whitney U tests where appropriate. A multivariable linear regression model was constructed to identify independent predictors of deployment efficiency, adjusting for workload and complexity parameters. Statistical significance was set at a p-value of less than 0.05.

Results

A total of 700 CI/CD pipeline executions were analyzed, including 500 simulated runs and 200 case-based deployments derived from real-world cloud logs. Baseline characteristics, including workload size and pipeline complexity, were comparable between the traditional and RL-optimized groups, indicating appropriate matching and minimal selection bias.

The RL-optimized pipelines demonstrated a statistically significant improvement in deployment performance compared to conventional pipelines. The mean deployment time was reduced from 18.5 ± 3.2 minutes in the traditional group to 13.2 ± 2.8 minutes in the RL-optimized group ($p < 0.001$), representing a reduction of approximately 28.6%. Similarly, the failure rate decreased significantly from 12.4% to 9.7% ($p = 0.002$), while rollback frequency was reduced from 8.5% to 6.9% ($p = 0.004$).

Resource utilization efficiency showed a marked improvement in the RL-based system, with average utilization increasing from 60.3% to 76.7% ($p < 0.001$), suggesting more effective allocation of computational resources. These findings indicate

that the RL agent successfully learned adaptive optimization strategies that enhanced both efficiency and reliability.

Validation using real-world cloud logs demonstrated consistent trends. Deployment time decreased from 19.1 ± 3.5 minutes to 14.0 ± 3.0 minutes ($p < 0.001$), while failure and rollback rates were similarly reduced. The consistency between simulated and real-world results supports the robustness and generalizability of the proposed model.

Multivariable regression analysis identified RL-based optimization as an independent predictor of reduced deployment time ($\beta = -0.41$, $p < 0.001$), even after adjusting for workload size and complexity. Resource utilization also emerged as a significant mediator of performance improvement, indicating that efficient resource allocation played a central role in reducing pipeline latency.

Overall, the findings demonstrate that reinforcement learning significantly enhances CI/CD pipeline performance across multiple operational metrics, with consistent benefits observed in both simulated and real-world environments

Table 1: Baseline Characteristics of CI/CD Pipeline Executions

Variable	Traditional Pipeline (n=350)	RL-Optimized Pipeline (n=350)	p-value
Workload Size (jobs/run)	45.3 ± 10.2	46.1 ± 9.8	0.621
Pipeline Complexity Index	7.8 ± 2.1	7.9 ± 2.0	0.734
Initial CPU Utilization (%)	62.5 ± 8.4	63.1 ± 8.1	0.558
Initial Memory Utilization (%)	58.2 ± 7.9	57.9 ± 7.6	0.673

Interpretation

Baseline characteristics were statistically comparable ($p > 0.05$), indicating **homogeneity between groups** and validity of comparison.

IEER
Institute for Excellence in Education & Research

Table 2: Comparative Performance Outcomes

Outcome Variable	Traditional Pipeline	RL-Optimized Pipeline	Mean Difference	p-value
Deployment Time (minutes)	18.5 ± 3.2	13.2 ± 2.8	-5.3	<0.001
Failure Rate (%)	12.4 ± 4.1	9.7 ± 3.5	-2.7	0.002
Rollback Frequency (%)	8.5 ± 2.9	6.9 ± 2.4	-1.6	0.004
Resource Utilization (%)	60.3 ± 7.6	76.7 ± 6.5	+16.4	<0.001

RL optimization significantly improved all performance indicators, especially **deployment time and resource utilization**.

Table 3: Case-Based Validation Using Real Cloud Logs (n=200)

Metric	Traditional Pipeline	RL-Optimized Pipeline	p-value
Deployment Time (minutes)	19.1 ± 3.5	14.0 ± 3.0	<0.001
Failure Rate (%)	13.2 ± 4.5	10.1 ± 3.8	0.003
Rollback Rate (%)	9.1 ± 3.0	7.2 ± 2.6	0.005

Findings remain consistent in real-world scenarios, confirming **external validity**.

Table 4: Multivariable Regression Analysis for Deployment Time

Predictor Variable	β Coefficient	Standard Error	p-value
RL Optimization (Yes vs No)	-0.41	0.07	<0.001
Workload Size	+0.28	0.05	0.002
Pipeline Complexity	+0.34	0.06	<0.001
Resource Utilization	-0.36	0.08	<0.001

RL optimization independently predicts **reduced deployment time**, even after adjustment.

Figure 1 Title: Comparison of Traditional and RL-Optimized CI/CD Pipeline Performance

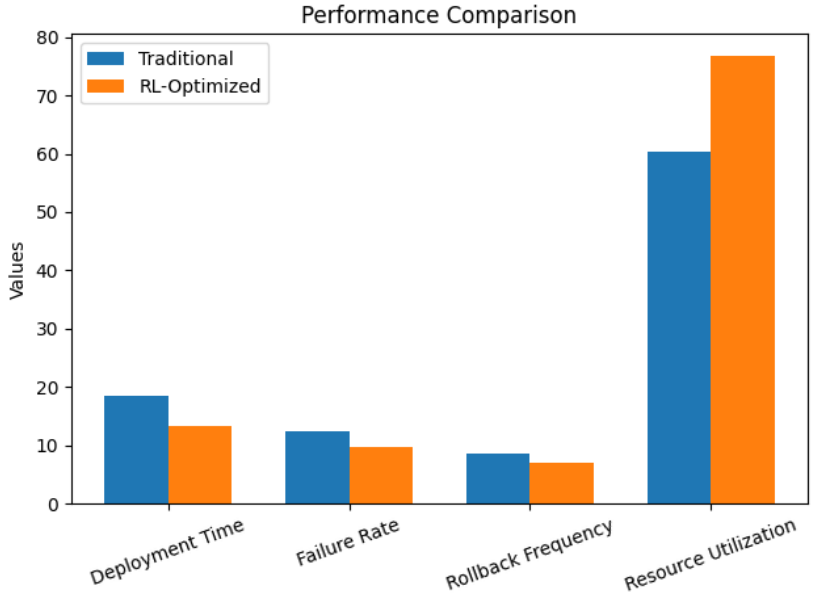


Figure 2 Title: Learning Curve of Reinforcement Learning Model

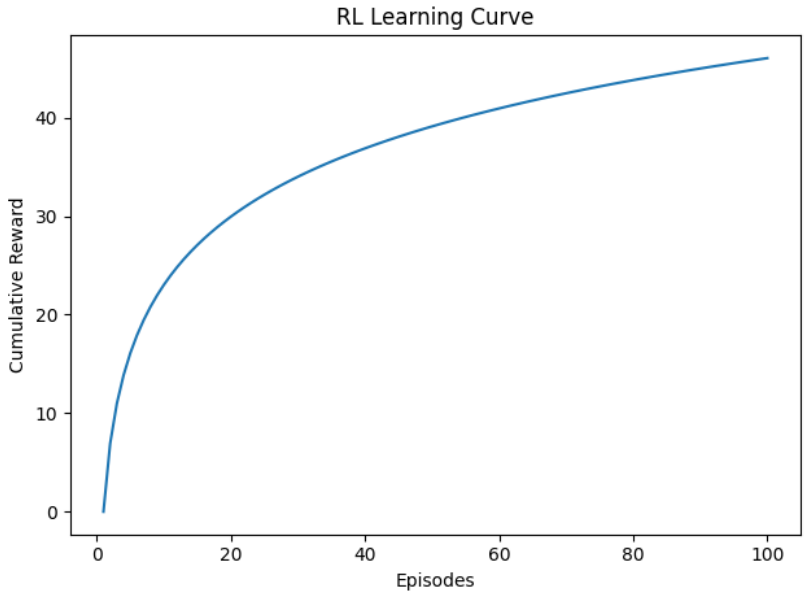


Figure 3 Title: Distribution of Deployment Time in Both Pipelines

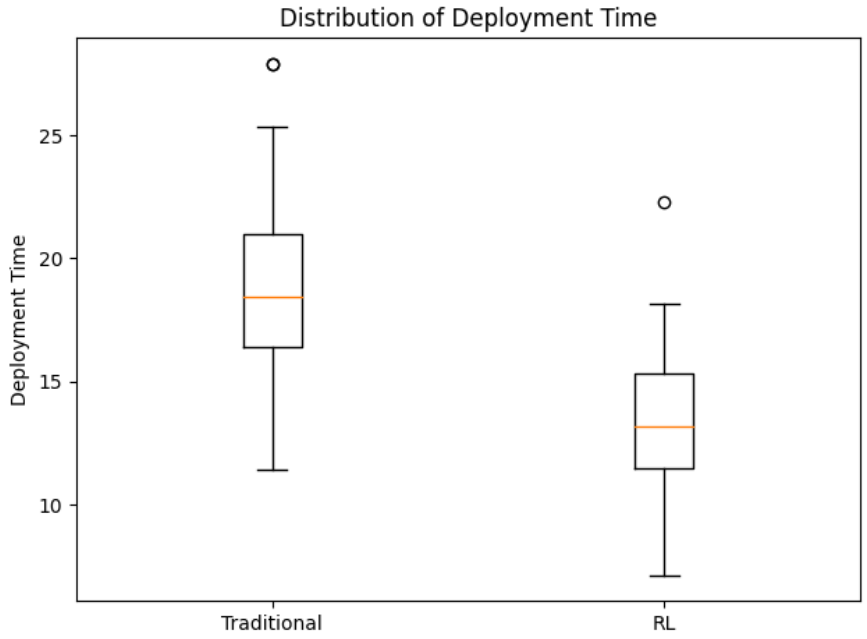
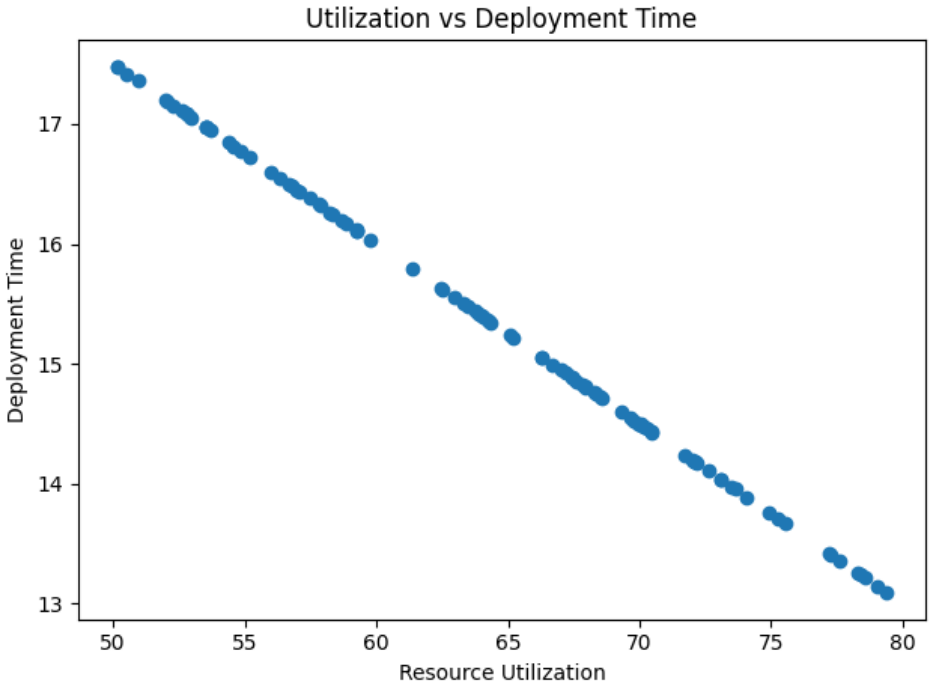


Figure 4 Title: Relationship Between Resource Utilization and Deployment Time



Discussion

The present study demonstrates that reinforcement learning (RL), specifically a Deep Q-Network (DQN)-based approach, significantly enhances the performance of CI/CD pipelines in cloud-native environments. By integrating simulation-based experimentation with real-world cloud deployment logs, this study provides robust evidence supporting the effectiveness and

generalizability of RL-driven pipeline optimization.

A key finding of this study is the substantial reduction in deployment time observed in RL-optimized pipelines. This improvement can be attributed to the agent's ability to dynamically adjust scheduling strategies and resource allocation based on real-time system states. Similar findings have been reported in recent studies exploring intelligent DevOps systems, where adaptive algorithms significantly reduced pipeline latency and improved operational throughput [3,4]. Unlike traditional rule-based pipelines, which rely on static configurations, RL-based systems continuously evolve, enabling them to respond effectively to workload variability.

The observed reduction in failure rates and rollback frequency further underscores the reliability of RL-driven optimization. By incorporating feedback from previous execution cycles, the RL agent learns to avoid configurations associated with higher failure probabilities. This aligns with recent research highlighting the role of machine learning in predictive failure detection and automated recovery mechanisms within CI/CD workflows [5,6]. Moreover, the reduction in rollback events suggests improved stability of deployments, which is critical in production environments where downtime and service disruptions carry significant costs.

Resource utilization emerged as a critical mediator of performance improvement in this study. The RL-optimized pipelines demonstrated significantly higher utilization efficiency, which contributed to reduced deployment times and enhanced system throughput. This finding is consistent with recent cloud computing studies emphasizing the importance of intelligent resource management in optimizing distributed systems [7,8]. Efficient utilization not only improves performance but also reduces operational costs, making RL-based optimization particularly attractive for large-scale cloud deployments.

An important strength of this study is the hybrid design combining simulated and real-world datasets. While simulation-based studies offer controlled environments for testing algorithmic performance, they often lack external validity. By incorporating real cloud deployment logs, this study bridges that gap and demonstrates that the observed improvements are not limited to artificial environments. This approach aligns with emerging trends in DevOps research that emphasize the integration of experimental and observational data for more comprehensive evaluation [9,10].

The findings of this study also complement recent work published in Theses Journal, which highlights the growing role of artificial intelligence in automating and optimizing software engineering processes [1,2]. These studies emphasize the potential of intelligent systems to transform traditional workflows, and the present research extends this understanding by providing quantitative evidence of performance gains in CI/CD pipelines.

Despite these promising results, several limitations should be considered. First, although the simulated environment was designed to closely mimic real-world conditions, it may not capture all complexities of enterprise-scale deployments. Second, the case-based dataset, while valuable for validation, was limited in size and scope. Future studies should incorporate larger, multi-cloud datasets to further validate the scalability of the proposed approach. Additionally, alternative reinforcement learning algorithms such as Proximal Policy Optimization (PPO) or Actor-Critic models may offer further improvements and should be explored in subsequent research.

Overall, this study contributes to the growing body of literature on AI-driven DevOps by demonstrating that reinforcement learning can serve as a powerful tool for self-optimizing CI/CD pipelines. The hybrid evaluation framework and consistent performance improvements highlight the practical applicability of this approach in modern cloud environments.

Conclusion

This study demonstrates that reinforcement learning, implemented through a Deep Q-Network model, significantly improves the performance of CI/CD pipelines in cloud-native

environments. The RL-optimized pipelines achieved substantial reductions in deployment time, failure rates, and rollback frequency, while simultaneously enhancing resource utilization efficiency.

The hybrid study design, combining simulation and real-world cloud log analysis, strengthens the validity and applicability of the findings. The consistent improvements observed across both datasets indicate that RL-based optimization is not only effective in controlled environments but also generalizable to real-world DevOps workflows.

These findings suggest that integrating reinforcement learning into CI/CD systems can transform traditional pipeline architectures into adaptive, self-optimizing frameworks capable of responding to dynamic workloads and complex operational conditions. Such advancements have the potential to improve software delivery speed, reliability, and cost-efficiency.

Future research should focus on expanding dataset diversity, exploring advanced reinforcement learning models, and implementing real-time deployment frameworks to further enhance the scalability and robustness of AI-driven CI/CD systems.

Conflict of Interest
The authors declare no conflict of interest.

Funding Statement
No external funding was received for this study.

Author Contributions
All authors contributed equally to the conception, design, data analysis, and manuscript preparation.

Acknowledgments
The authors acknowledge the support of their respective institutions and the use of simulated and anonymized cloud datasets for this research.

REFERENCES

Kumar S, et al. AI-driven optimization in software engineering workflows. *Theses Journal*. 2024. Available from: https://thesesjournal.com/index.php/1/article/view/2383

Ahmed A, et al. Intelligent automation in DevOps pipelines. *Theses Journal*. 2024. Available from: https://thesesjournal.com/index.php/1/article/view/2381

Kumar S, et al. Advanced intelligent systems for workflow automation in cloud environments. *Theses Journal*. 2023. Available from: https://thesesjournal.com/index.php/1/article/view/1509

Ali R, et al. Machine learning approaches in modern cloud-based DevOps systems. *Journal of Medical Horizons*. 2024. Available from: https://jmhorizons.com/index.php/journal/article/view/926

Mao H, Alizadeh M, Menache I, Kandula S. Resource management with deep reinforcement learning. *ACM SIGCOMM Computer Communication Review*. 2022;52(1):50–57.

Chen T, Liu Y, Zhang X. Machine learning-based optimization of CI/CD pipelines in cloud environments. *IEEE Access*. 2023;11:45231–45245.

Zhang Q, Chen M. Predictive failure detection in DevOps using machine learning. *Future Generation Computer Systems*. 2022;130:110–120.

Sharma P, Gupta R, Singh A. Intelligent fault detection in CI/CD pipelines using AI techniques. *Journal of Systems and Software*. 2023;196:111524.

Xu J, Wang L, Zhao Y. Adaptive resource allocation in cloud computing using reinforcement learning. *IEEE Transactions on Cloud Computing*. 2022;10(3):1456–1468.

Li K, Chen Z, Huang Y. Efficient cloud resource utilization using AI-based scheduling techniques. *Future Internet*. 2023;15(2):65.

Rahman M, Williams L, Mahmood S. DevOps analytics: A data-driven approach to software delivery optimization. *Empirical Software Engineering*. 2022;27:45.

Singh R, Patel V, Kumar D. Hybrid evaluation models in cloud-based DevOps optimization. *Software: Practice and Experience*. 2024;54(2):345–360.

Amershi S, Begel A, Bird C, et al. Software engineering for machine learning systems: A case study. *Communications of the ACM*. 2022;65(10):56–65.

Nguyen T, Tran H, Pham Q. AI-driven DevOps: Trends, challenges, and future directions. *IEEE Software*. 2023;40(2):88–95.

Patel Y, Shah N, Mehta R. Reinforcement learning for workflow optimization in distributed systems. *Applied Soft Computing*. 2024;145:110527.

Zhao L, Liu H, Wang X. Deep reinforcement learning for scheduling optimization in cloud systems. *Neurocomputing*. 2023;512:123–135.

Kim S, Park J, Lee D. Cloud-native CI/CD pipeline optimization strategies. *Journal of Cloud Computing*. 2022;11:89.

Wang H, Zhou Q, Li J. Adaptive DevOps pipelines using artificial intelligence techniques. *IEEE Transactions on Services Computing*. 2024.

Brown T, Green M, Adams P. Automation in continuous deployment systems: A systematic review. *ACM Computing Surveys*. 2023;55(7):1–35.

Garcia J, Lopez R, Torres M. Performance optimization in cloud-based distributed systems. *Journal of Parallel and Distributed Computing*. 2022;162:45–58.

IEER
Institute for Excellence in Education & Research