International Journal of Computational and Experimental Science and ENgineering (IJCESEN)

Vol. 12-No.1 (2026) pp. 1218-1228
http://www.ijcesen.com

ISSN: 2149-9144

Research Article

Automated Root Cause Analysis in Distributed Cloud Environments: An Unsupervised AIOps Approach Using BigQuery ML

Arun Harikrishnan*

UNYBRANDS LLC, USA
* **Corresponding Author Email:** arun.harikr@gmail.com - **ORCID:** 0000-0002-5247-1772

Article Info:

DOI: 10.22399/ijcesen.4996
Received : 01 December 2025
Revised : 01 February 2026
Accepted : 02 February 2026

Keywords

AIOps,
Unsupervised Learning,
Anomaly Detection,
Root Cause Analysis,
Cloud Operations

Abstract:

Modern distributed cloud environments present unprecedented challenges for operational monitoring and incident management, where traditional rule-based systems fail to effectively handle the scale and complexity of containerized microservices architectures. This article presents an advanced AIOps framework that leverages unsupervised machine learning techniques through BigQuery ML to automate anomaly detection and root cause analysis in distributed cloud systems. The framework aggregates heterogeneous telemetry data from container orchestration platforms, application traces, and network flow logs into unified analytics pipelines that establish dynamic operational baselines without requiring labeled training datasets. The implementation employs clustering algorithms and statistical deviation detection mechanisms to identify anomalous patterns across multiple system layers while suppressing operational noise and false positive alerts. Experimental evaluation in production environments demonstrates substantial improvements in detection accuracy, significant reductions in mean time to detection and resolution, and enhanced operational efficiency for site reliability engineering teams. The SQL-based machine learning implementation enables operations teams to deploy sophisticated analytics without specialized data science expertise, providing scalable anomaly detection capabilities that adapt to evolving operational patterns in dynamic cloud environments.

1. Introduction and Background

1.1 Current State of Microservices Monitoring

The contemporary landscape of enterprise cloud computing has transformed through widespread microservices adoption. Modern organizations deploy distributed applications across containerized platforms. These systems comprise numerous loosely coupled services operating within ephemeral container instances. Each service exhibits dynamic scaling behaviors and complex communication patterns. This architectural shift challenges traditional monitoring approaches designed for monolithic applications.

Distributed cloud environments present unprecedented monitoring complexity. Container orchestration platforms host thousands of microservices across multiple clusters. Individual services scale dynamically based on load conditions. Traditional monitoring frameworks struggle with these dynamic topologies. Service dependencies create intricate interaction patterns where failures propagate unpredictably [1].

The scale of modern containerized deployments exceeds conventional monitoring capabilities. Enterprise clusters routinely manage hundreds of services simultaneously. Each service generates multiple telemetry streams, including logs, metrics, and traces. Container lifecycles are ephemeral, creating constantly changing monitoring targets. This dynamic nature makes static monitoring configurations ineffective.Volume characteristics of telemetry data have grown exponentially in containerized architectures. Modern deployments produce massive quantities of operational data continuously. This includes structured application logs from microservice interactions. Performance metrics span multiple-dimensional hierarchies across infrastructure layers. Distributed trace spans capture request flows traversing service boundaries. Network communication metadata documents inter-service traffic patterns [1].Traditional rule-based monitoring systems show significant limitations in

cloud-native environments. These legacy approaches depend on static threshold configurations. Manual tuning becomes complex as the system scale increases. Static thresholds cannot accommodate natural workload variability. Container workloads exhibit fluctuating resource consumption patterns. Traffic patterns influence normal operational parameters. External dependencies affect baseline behaviors. Consequently, these systems generate excessive false positive alerts while missing subtle systemic anomalies.

1.2 Problem Definition

Alert fatigue has reached epidemic proportions in enterprise cloud operations. Production environments generate overwhelming volumes of monitoring notifications. Most alerts represent false positives rather than actionable incidents. This creates cognitive overload for operations teams. Personnel become desensitized to alert notifications. Critical issues risk being overlooked during high-volume periods. Alert fatigue fundamentally undermines incident response effectiveness.Manual root cause analysis processes exhibit inherent inefficiencies in distributed environments. Traditional investigation methods require extensive data correlation across multiple sources. Application logs from dozens of services must be analyzed simultaneously. Infrastructure metrics span compute, storage, and network layers. Distributed trace data captures request flows across service boundaries. Engineering teams invest substantial time in manual correlation tasks. This reduces capacity for strategic problem-solving activities.Conventional monitoring approaches suffer from poor signal-to-noise ratios. Threshold-based alerting systems demonstrate low precision in operational contexts. Generated alerts frequently lack actionable content for operations teams. Poor alert quality wastes engineering resources significantly. Confidence in monitoring systems erodes over time. Teams develop alert suppression behaviors as defensive mechanisms. This increases risks of missing genuine critical incidents.

Site reliability engineering productivity suffers under ineffective monitoring systems. Teams allocate the majority time responding to false positive alerts. Manual root cause analysis consumes substantial engineering effort. Limited capacity remains for proactive reliability improvements. Strategic infrastructure optimization activities receive reduced attention. This reactive operational model increases incident resolution times. Overall system reliability decreases through inadequate preventive maintenance practices.

1.3 Research Motivation and Objectives

Intelligent automated anomaly detection frameworks have become critically necessary. Large-scale distributed systems require algorithmic monitoring approaches. Manual methods cannot scale with exponential complexity growth. Data volumes consistently exceed human processing capability. Algorithmic approaches maintain consistent analytical quality regardless of scale. These systems process vast telemetry quantities while preserving detection accuracy.Unsupervised learning provides compelling advantages in operational contexts. Failure modes evolve continuously in dynamic cloud environments. Historically labeled datasets may not represent future anomaly patterns accurately. Supervised approaches require extensive training data preparation efforts [2]. Ongoing model maintenance demands specialized expertise. Unsupervised methods adapt dynamically to changing operational patterns. These techniques identify novel anomaly patterns without manual configuration requirements.Contemporary research objectives center on practical methodology development. Cloud platform capabilities should be leveraged effectively. Implementation complexity must remain minimal for operational teams. Overhead requirements should not burden existing infrastructure. Machine learning techniques must demonstrate measurable improvements. False positive alert rates require a significant reduction. Anomaly detection sensitivity needs enhancement. Root cause attribution accuracy demands improvement in production systems [2].The proposed solution architecture integrates with native cloud observability infrastructure. Heterogeneous telemetry streams are processed through unified analytics pipelines. Operations teams implement sophisticated capabilities without specialized expertise. External tool integration requirements are eliminated. Dynamic operational baselines are established through continuous clustering analysis. Statistical deviations indicate potential systemic issues automatically. Normal operational variations receive appropriate noise suppression.

2. Related Work and Technical Foundation

2.1 AIOps Evolution and Current Approaches

Algorithmic operations management emerged from traditional data center environments where manual processes dominated system administration. Early implementations relied on simple automation

scripts and rule-based expert systems. These systems encoded operational knowledge into decision trees and workflow automation. System administrators defined explicit conditional logic for handling routine operational tasks. Rule-based approaches worked effectively in static environments with predictable failure patterns. However, cloud computing introduced dynamic complexities that exceeded static rule capabilities.

Contemporary operational settings call for complex analytical techniques beyond simple rule-based systems. Machine learning algorithms increasingly replace manual decision processes in operational contexts. Supervised learning methods train on historical incident data to classify future operational events. These approaches achieve high accuracy when training datasets comprehensively represent operational scenarios. Classification algorithms learn from labeled examples of normal and anomalous system behaviors. Regression models predict system performance based on historical patterns and current operational states [3].

Unsupervised learning techniques provide advantages in operational environments where failure patterns evolve continuously. These methods identify anomalies without requiring pre-labeled training examples from operational history. Clustering algorithms automatically discover natural groupings within operational telemetry data. Statistical outlier detection identifies deviations from established operational baselines without manual threshold configuration. Density-based anomaly detection excels at discovering irregularly shaped anomaly patterns in high-dimensional operational data spaces.

Contemporary root cause analysis frameworks employ graph-theoretic approaches to model complex system dependencies. These frameworks construct dependency graphs representing relationships between distributed system components. Causal inference algorithms trace failure propagation paths through interconnected service architectures. Probabilistic models estimate causation likelihood based on temporal correlation patterns between system events. Commercial platforms integrate multiple analytical techniques to improve root cause attribution accuracy in complex distributed environments [3].

2.2 Machine Learning in Infrastructure Monitoring

Clustering algorithms form the foundation of modern operational data analysis in distributed computing environments. K-means clustering partitions multidimensional telemetry data into distinct operational state clusters. These clusters represent different modes of normal system operation under varying load conditions. Hierarchical clustering techniques reveal nested relationships within complex operational behavior patterns. Density-based clustering methods identify anomalies in sparse regions of high-dimensional telemetry feature spaces. Gaussian mixture models accommodate overlapping operational states that traditional hard clustering approaches cannot handle effectively.

Time-series analysis techniques specifically address temporal characteristics inherent in infrastructure monitoring data. Seasonal decomposition algorithms separate regular operational patterns from anomalous behavioral variations. These methods distinguish between expected cyclical patterns and genuine system anomalies. Trend analysis identifies gradual performance degradations that develop over extended operational periods. Change point detection algorithms automatically identify specific temporal moments when system behavior undergoes significant shifts. Autoregressive integrated moving average models capture complex temporal dependencies in sequential metric observations [4].

Correlation analysis methodologies enable a comprehensive understanding of relationships across multidimensional telemetry streams from distributed systems. Linear correlation analysis quantifies relationships between different infrastructure metrics collected simultaneously. Rank-based correlation techniques capture monotonic relationships that traditional linear methods cannot detect. Cross-correlation analysis identifies temporal delays between causally related events occurring across different system components. Mutual information measures quantify non-linear dependencies between telemetry variables that traditional correlation approaches miss entirely.

Statistical baseline establishment requires adaptive techniques that accommodate natural variability in dynamic cloud computing environments. Traditional fixed threshold approaches fail when operational parameters exhibit significant natural variation. Exponential smoothing algorithms adaptively adjust baselines based on recent operational history patterns. Seasonal adjustment techniques account for regular traffic variations and scheduled maintenance activities. Robust statistical estimation methods minimize the influence of transient outliers on baseline calculations [4].

2.3 Cloud Platform Integration

Contemporary cloud platforms provide comprehensive observability infrastructure through integrated analytics and machine learning

capabilities. SQL-based machine learning workflows democratize advanced analytical techniques for operations teams without specialized data science backgrounds. These platforms eliminate traditional barriers to implementing sophisticated analytical approaches in operational contexts. Declarative query interfaces enable complex analytical operations through familiar database management paradigms. Automated feature engineering capabilities reduce manual data preprocessing requirements significantly. Integrated visualization frameworks support exploratory data analysis and analytical model interpretation activities.

For thorough telemetry collecting across service boundaries, container orchestration systems automatically instrument distributed applications. Centralized logging systems combine structured as well as unstructured data streams from containerized applications. Multidimensional measures gathered throughout the computer, storage, and network infrastructure levels from performance metric collection frameworks are gathered. Distributed tracing systems capture complete request execution paths spanning multiple microservice boundaries. These diverse telemetry streams integrate through unified data pipeline architectures for centralized analytical processing.

Network observability frameworks provide essential visibility into communication patterns within complex distributed system architectures. Flow-based analysis techniques document traffic volumes and connection establishment patterns between distributed system components. Network performance monitoring identifies bandwidth utilization constraints and communication latency issues affecting application performance. Security-focused network analysis enables the detection of anomalous communication behaviors that indicate potential security incidents. Deep packet inspection capabilities provide application-layer protocol visibility for complex distributed system troubleshooting scenarios.

Comprehensive monitoring platform integration enables unified operational workflows across diverse analytical and operational tools. Application programming interfaces facilitate seamless bidirectional data exchange between analytical platforms and existing operational toolchains. Event-driven integration mechanisms trigger automated operational responses based on analytical findings and confidence assessments. Unified dashboard architectures provide consolidated operational visibility across multiple monitoring and analytical domains. Intelligent alert routing systems ensure appropriate incident escalation based on analytical model confidence levels and operational impact assessments.

3. Proposed AIOps Framework Architecture

3.1 Data Collection and Aggregation Pipeline

The proposed architecture implements a comprehensive data ingestion system that consolidates telemetry from diverse containerized cloud sources. Container orchestration platforms generate extensive operational data through runtime interfaces and event mechanisms. Individual containers produce structured log streams documenting application activities and system interactions. Log entries capture lifecycle events, resource allocation decisions, and application-specific operational messages. Container runtime systems automatically instrument workloads to collect telemetry without requiring application code modifications.

Container log processing employs advanced parsing techniques to extract structured information from heterogeneous log formats across distributed applications. Natural language processing algorithms identify operationally relevant events within unstructured log messages generated by different application frameworks. Regular expression patterns capture structured data elements from application-specific logging formats and custom message structures. Log normalization processes standardize timestamp formats, severity classifications, and metadata schemas across diverse application environments. Schema inference algorithms automatically detect and parse data structures within JSON and XML-formatted log entries [5].

Application trace correlation establishes comprehensive visibility into distributed request execution paths spanning multiple microservice boundaries within complex architectures. Distributed tracing systems automatically instrument application code to capture detailed span information for each inter-service interaction and database query. Trace correlation algorithms reconstruct complete request flows by analyzing span relationships based on trace identifiers and hierarchical parent-child connections. Request topology analysis identifies critical execution paths and performance bottleneck services within distributed transaction flows. Performance attribution techniques isolate latency contributions from individual service components to enable targeted optimization efforts.Network behavior characterization analyzes communication patterns through comprehensive flow-based telemetry collection mechanisms across distributed system

infrastructures. Flow log analysis documents detailed traffic volumes, connection establishment patterns, and protocol-level communication statistics between distributed system components and external dependencies. Network topology discovery algorithms automatically map service dependency relationships based on observed communication patterns and connection metadata. Anomaly detection algorithms applied to network flows identify unusual communication behaviors that potentially indicate security incidents or underlying infrastructure failures [5].

3.2 BigQuery ML Model Implementation

Unsupervised clustering model design focuses on automatically identifying natural groupings within multidimensional operational telemetry datasets collected from distributed systems. K-means clustering algorithms partition high-dimensional telemetry feature vectors into distinct operational state clusters representing different modes of normal system behavior under varying load conditions. Gaussian mixture models accommodate overlapping operational states where traditional hard clustering boundaries prove inadequate for capturing operational complexity. Density-based clustering techniques identify anomalous data points located in sparse regions of high-dimensional feature spaces that indicate unusual system behaviors.

SQL-based machine learning pipeline development democratizes advanced analytical capabilities for operations teams without requiring specialized data science expertise or external tooling dependencies. Declarative query interfaces enable complex feature engineering operations through familiar database management paradigms and existing operational workflows. Automated data preprocessing pipelines handle missing value imputation, outlier treatment strategies, and feature scaling operations across diverse telemetry data types. Window functions enable temporal aggregation operations and rolling statistical calculations across time-series telemetry streams from multiple system components [6].

Hyperparameter optimization employs automated techniques to determine optimal clustering configurations for operational datasets without manual intervention requirements. Grid search algorithms systematically evaluate clustering quality metrics across multiple parameter combinations and configuration spaces. Silhouette analysis quantifies clustering effectiveness by measuring intra-cluster cohesion levels and inter-cluster separation distances across different parameter settings. Elbow method analysis identifies optimal cluster quantities by evaluating within-cluster sum of squares metrics across different clustering configurations. Cross-validation techniques ensure clustering stability and reproducibility across different temporal subsets of operational training data.

Dynamic baseline establishment algorithms continuously adapt to evolving operational patterns without requiring manual threshold configuration or expert domain knowledge. Exponential smoothing techniques automatically weight recent operational observations more heavily than historical data points to capture current system behavior. Seasonal adjustment algorithms account for regular cyclical patterns in operational metrics related to business cycles and usage patterns. Robust statistical estimation methods minimize the influence of transient outliers and anomalous events on baseline calculations and threshold determination processes [6].

3.3 Anomaly Detection and Correlation Engine

Multi-dimensional anomaly scoring methodologies combine sophisticated statistical analysis techniques with machine learning model outputs to generate comprehensive anomaly assessment scores. Individual metric anomaly scores quantify the statistical significance of observed deviations from established operational baselines across different system layers. Composite scoring algorithms intelligently weight individual anomaly indicators based on historical correlation patterns with genuine operational incidents and their resolution outcomes. Confidence estimation techniques provide uncertainty quantification for anomaly assessments to support informed operational decision-making processes and resource allocation strategies.

Cross-stack signal correlation algorithms identify complex relationships between infrastructure events and application performance anomalies across multiple system layers and architectural tiers. Temporal correlation analysis detects time-delayed relationships between causally related events occurring across different system components and service boundaries. Graph-based correlation techniques model intricate dependency relationships between distributed system elements based on observed interaction patterns. Statistical correlation measures quantify the strength of relationships between different telemetry streams to identify potential causation patterns and dependency chains [7].

Incident prioritization mechanisms systematically rank detected anomalies based on potential operational impact assessments and statistical confidence levels. Business impact modeling

estimates potential service disruption scenarios based on affected system components and their documented dependency relationships. Historical incident analysis provides empirical data for calibrating prioritization algorithms based on past incident resolution outcomes and resource requirements. Resource constraint modeling considers available engineering capacity and expertise when determining optimal incident handling priorities and response strategies.

Noise suppression techniques systematically reduce false positive rates by filtering transient anomalies and normal system variations that do not indicate genuine operational issues. Statistical significance testing eliminates detected anomalies that fall within expected operational variation ranges based on historical baselines. Persistence filtering requires anomalies to maintain statistical significance over minimum duration thresholds before generating actionable alerts for operations teams. Contextual analysis algorithms consider scheduled maintenance activities and planned system changes when evaluating anomaly significance and operational relevance [7].

3.4 Root Cause Analysis Automation

Causal inference techniques employ sophisticated probabilistic graphical models to identify likely causation sequences during complex operational incidents across distributed systems. Directed acyclic graphs model dependency relationships between system components based on observed interaction patterns and configuration metadata. Bayesian inference algorithms estimate causation probabilities based on temporal correlation patterns and system topology information derived from operational telemetry. Counterfactual analysis techniques systematically evaluate whether incidents would have occurred without specific contributing factors or environmental conditions.

Timeline reconstruction algorithms aggregate temporally related events into coherent incident narratives that support rapid operational understanding and response coordination. Event correlation techniques identify related activities across multiple system components and diverse data sources using temporal and semantic analysis. Temporal ordering algorithms sequence complex events based on precise timestamp analysis and causal dependency constraints derived from system architecture. Narrative generation capabilities produce human-readable incident timelines that facilitate knowledge transfer processes and comprehensive post-incident analysis activities.

Resource dependency mapping constructs detailed models of system interdependencies to support comprehensive impact assessment and strategic remediation planning. Service dependency discovery algorithms analyze communication patterns and configuration data to identify critical service relationships and potential failure points. Resource utilization modeling predicts cascading failure scenarios based on current system loading conditions and documented capacity constraints. Impact propagation analysis estimates potential service disruption scenarios based on individual component failure probabilities and dependency chain analysis.

Automated runbook generation produces detailed procedural guidance for incident remediation based on historical resolution patterns and accumulated system knowledge. Template-based documentation systems generate comprehensive step-by-step remediation procedures tailored to specific incident characteristics and system configurations. Knowledge base integration gives quick access to proven operational best practices and historical incident resolution methods. Operations teams may always perfect and improve automated remediation guidance based on operational experience and changing system architectures, thanks to cooperative editing tools.

4. Experimental Evaluation and Results

4.1 Experimental Setup and Dataset Characteristics

The experimental evaluation was conducted within a production-grade containerized environment hosting diverse microservices across multiple geographical regions. The infrastructure comprised numerous container instances distributed across orchestrated clusters to provide comprehensive coverage of real-world operational scenarios. Container deployment architectures included various service types ranging from stateless web applications to stateful data processing workflows. This diversity created representative operational complexity suitable for thorough evaluation purposes. The production environment maintained continuous operation throughout the evaluation period without planned downtime.

Telemetry data collection spanned an extended evaluation period covering multiple operational cycles and seasonal business patterns affecting system load. The comprehensive dataset encompassed heterogeneous data types, including structured application logs from microservices interactions. Performance metrics across multiple-dimensional hierarchies captured system behavior at various operational layers. Distributed trace information documented request flows across

service boundaries and external dependencies. Network communication metadata provided visibility into inter-service traffic patterns and external system interactions [8].

The production deployment generated substantial telemetry volumes through continuous monitoring instrumentation across all containerized workloads and supporting infrastructure components. Container lifecycle events documented creation, scaling, and termination activities across different operational contexts. Resource utilization patterns captured compute, memory, storage, and network consumption across varying load conditions. Application performance indicators measured response times, throughput, and error rates across different service categories. Inter-service communication data documented API calls, message passing, and database interactions throughout the distributed system architecture.

Baseline comparison utilized traditional threshold-based monitoring systems with manually configured alert rules developed through extensive operational experience over multiple years. The conventional monitoring approach employed static threshold configurations calibrated by experienced site reliability engineers. Performance benchmarking methodology emphasized practical operational metrics including detection accuracy, response time improvements, and resource efficiency measurements. Evaluation criteria focused on operational relevance rather than theoretical performance indicators to ensure practical applicability [8].

4.2 Anomaly Detection Performance Analysis

False positive reduction analysis demonstrated substantial improvements compared to conventional threshold-based detection systems across multiple evaluation dimensions and operational contexts. The unsupervised learning approach consistently outperformed traditional monitoring systems across different service categories and operational loading conditions. Statistical significance testing confirmed the reliability and consistency of performance improvements across various operational scenarios. Cross-validation techniques validated performance consistency across different temporal periods of the evaluation dataset. The machine learning approach showed robust performance characteristics regardless of seasonal operational variations.

True positive detection rate analysis revealed performance characteristics that varied across different incident categories and operational complexity levels. Infrastructure-related anomalies demonstrated higher detection accuracy compared to complex application-level performance degradation scenarios. The system showed particular effectiveness in identifying gradual performance issues that developed over extended periods. Traditional monitoring approaches typically missed these subtle degradations until critical impact thresholds were exceeded. Resource exhaustion scenarios showed consistently high detection rates across different resource types and consumption patterns [9].

Precision and recall evaluation in operational contexts provided comprehensive assessment of detection quality across diverse incident scenarios and system configurations. The evaluation methodology considered both immediate detection accuracy and sustained performance over extended operational periods. Temporal analysis examined detection consistency across different time windows and operational conditions. The system maintained stable performance characteristics despite varying operational loads and system modifications. Cross-validation ensured performance reliability across different subsets of the evaluation dataset.

Comparison with rule-based detection systems highlighted significant advantages of the adaptive machine learning approach in dynamic operational environments. Traditional threshold-based systems showed degraded performance when operational patterns deviated from historical baselines used for calibration. Manual threshold adjustment proved time-consuming and error-prone across complex distributed system environments. The proposed approach demonstrated adaptive capabilities that maintained detection effectiveness despite evolving operational characteristics. System modifications and deployment changes did not require manual recalibration of detection parameters [9].

4.3 Root Cause Analysis Effectiveness

Mean Time to Detection improvements showed substantial reductions across various incident categories, with infrastructure-related issues demonstrating the most significant performance gains. The automated approach consistently identified anomalous patterns earlier in incident development cycles compared to traditional monitoring approaches. Early detection capabilities enabled proactive response strategies that prevented full service impact in numerous evaluation scenarios. Detection timing analysis showed consistent improvement across different incident severity levels. The system proved particularly effective at identifying cascading failure scenarios before widespread service impact occurred.

Mean Time to Resolution analysis revealed substantial improvements attributable to enhanced initial diagnosis accuracy and streamlined investigative workflows. The automated system provided structured incident analysis that reduced manual correlation efforts required for complex distributed system failures. Diagnostic accuracy improvements accelerated the investigation process by highlighting probable root causes based on statistical analysis. System dependency modeling enhanced causal attribution capabilities across complex service interaction patterns. Integration with existing operational procedures maintained workflow compatibility while improving analytical effectiveness [10].

Accuracy assessment of automated causal attribution was validated through comprehensive comparison with detailed post-incident analysis conducted by experienced operational personnel. The system demonstrated high attribution accuracy for single-point failures and linear causation chains across different system components. Complex multi-service interaction scenarios showed reduced but still acceptable attribution accuracy levels. Validation methodology employed objective assessment techniques to ensure unbiased evaluation of diagnostic capabilities. Blind evaluation processes confirmed attribution accuracy across various incident categories and operational contexts.

Site reliability engineering workflow efficiency improvements were measured through detailed time allocation analysis and productivity assessment methodologies. Operations teams reported substantial reductions in time spent investigating false positive alerts and conducting manual correlation activities. The automated approach freed significant engineering capacity for strategic system reliability improvements and proactive optimization activities. Productivity gains enabled focus on preventive measures rather than reactive incident management tasks. Team satisfaction metrics improved significantly due to reduced alert fatigue and enhanced diagnostic confidence [10].

4.4 Scalability and Resource Utilization

Machine learning model performance evaluation demonstrated consistent analytical quality across varying dataset sizes and operational scales without degradation. Training operations scaled effectively with telemetry volume increases without proportional computational overhead growth. Model inference performance remained stable during peak telemetry ingestion periods while maintaining real-time anomaly detection capabilities. Horizontal scaling capabilities accommodated system growth without requiring architectural modifications. The analytical framework maintained consistent performance characteristics across different cluster sizes and operational complexities.

Computational resource requirements analysis showed moderate overhead for continuous model training and inference operations across the evaluation infrastructure. The analytical workload consumed minimal cluster computational capacity while providing substantial operational benefits and performance improvements. Memory utilization remained stable across varying telemetry volumes and model complexity levels. Processing overhead scaled linearly with data volume rather than exponentially, ensuring sustainable performance growth. Cost-benefit analysis demonstrated significant economic advantages through reduced manual investigation time and improved incident resolution efficiency.

Real-time processing capabilities maintained consistent low latency for anomaly detection operations even during maximum telemetry ingestion rates and system stress conditions. Stream processing architecture scaled horizontally to accommodate traffic bursts while maintaining analytical quality and detection accuracy. Buffer management strategies effectively handled temporary capacity constraints without data loss or detection delays. Load balancing mechanisms distributed processing workload across available computational resources to optimize performance. System responsiveness remained consistent across different operational loading patterns and traffic variations.System reliability assessment showed consistent availability and operational stability throughout the extended evaluation period without service interruptions or performance degradation. Automated failover mechanisms ensured continuous analytical operation during infrastructure maintenance activities and system updates. The framework demonstrated robust performance characteristics suitable for mission-critical operational environments and high-availability requirements. Fault tolerance mechanisms maintained analytical capabilities despite individual component failures or network connectivity issues. Recovery procedures restored full analytical functionality quickly following any temporary service disruptions.

4.5 Case Studies and Practical Applications

Representative incident scenarios provided detailed validation of automated analysis capabilities across diverse failure modes and distributed system configurations. Database connection pool

exhaustion incidents demonstrated the system's ability to trace application-level symptoms back to underlying infrastructure root causes. The analytical framework correctly identified resource bottlenecks and dependency relationships that contributed to service degradation. Memory leak scenarios validated gradual anomaly detection capabilities that traditional monitoring approaches consistently missed. Performance degradation incidents showed effective correlation between multiple system components and their causal relationships.Comparison between manual and automated root cause identification processes highlighted significant improvements in investigation efficiency and diagnostic accuracy. Automated analysis consistently identified relevant system components and probable causation chains faster than manual correlation processes. The structured analytical approach reduced investigation time while improving diagnostic accuracy across various incident categories and complexity levels. Human investigators validated automated findings, confirming high accuracy rates across different operational scenarios. Integration with existing troubleshooting workflows enhanced rather than replaced human expertise and domain knowledge.Integration success with existing operational procedures showed seamless workflow incorporation without disrupting established incident response processes and organizational practices. The system enhanced existing operational capabilities while maintaining compatibility with familiar tools and procedures. Alert integration preserved existing escalation procedures and communication channels used by operations teams. Dashboard integration provided consolidated analytical insights within established operational interfaces. Training requirements remained minimal due to intuitive interfaces and integration with existing operational workflows.Site reliability engineering team feedback indicated substantial improvements in operational confidence and a significant reduction in alert fatigue across different operational roles. Teams appreciated improved signal-to-noise ratios in operational alerts and enhanced initial incident analysis quality. Productivity improvements enabled strategic focus on system reliability enhancements rather than reactive incident management activities. Job satisfaction metrics improved due to reduced repetitive investigation tasks and increased focus on meaningful reliability engineering work. Knowledge transfer improved through automated documentation and structured incident analysis capabilities.

Table 1: *Comparison of Monitoring Approaches in Distributed Systems. [2]*

Monitoring Approach	Detection Method	Operational Characteristics
Rule-based Systems	Static threshold configuration	Manual calibration is required for each metric
Supervised Learning	Labeled training datasets	High accuracy with concept drift limitations
Unsupervised Learning	Statistical clustering analysis	Adaptive baseline establishment without labels

Table 2: *Data Sources and Processing Pipeline Components. [7]*

Telemetry Source	Data Type	Processing Mechanism
Container Orchestration	Lifecycle events and resource metrics	Real-time stream processing with schema normalization
Application Tracing	Distributed request spans	Correlation algorithms for topology reconstruction
Network Flow Logs	Communication patterns and traffic statistics	Anomaly detection for unusual behavior identification

Table 3*: Performance Metrics Comparison Across Detection Systems. [8]*

System Type	Precision Performance	Recall Effectiveness
Threshold-based Monitoring	Limited precision in dynamic environments	Inconsistent recall across incident categories
Machine Learning Framework	Enhanced precision through adaptive baselines	Improved recall for gradual performance issues
Hybrid Integration	Optimized precision with contextual analysis	Comprehensive recall across multiple failure modes

Table 4: *Operational Impact Assessment and Efficiency Improvements. [10]*

Operational Metric	Traditional Systems	Proposed Framework

Mean Time to Detection	Extended detection periods for complex issues	Accelerated identification through automated correlation
Mean Time to Resolution	Manual investigation workflows	Streamlined analysis with causal attribution
Engineering Productivity	Reactive incident management focus	Proactive reliability optimization emphasis

5. Conclusions

This article demonstrates the practical effectiveness of unsupervised machine learning techniques for automating anomaly detection and root cause analysis in complex distributed cloud environments. The proposed BigQuery ML-based framework addresses critical operational challenges, including alert fatigue, manual investigation inefficiencies, and scaling limitations inherent in traditional monitoring systems. The framework achieves substantial improvements in operational efficiency through automated baseline establishment, dynamic anomaly detection, and intelligent incident correlation across multiple system layers. The SQL-based implementation approach democratizes advanced analytics capabilities, enabling operations teams to leverage sophisticated machine learning techniques without requiring specialized expertise or external tooling dependencies. Experimental validation confirms significant reductions in false positive alerts, improved detection sensitivity for subtle performance degradations, and enhanced root cause attribution accuracy across diverse operational scenarios. The framework's emphasis on seamless integration with existing operational workflows ensures practical adoption while providing measurable improvements in incident response effectiveness and engineering productivity. The demonstrated capabilities represent a practical pathway for organizations to enhance operational resilience and system reliability through intelligent automation, while maintaining compatibility with established operational procedures and infrastructure investments. Future developments should focus on extending the framework to multi-cloud environments and incorporating advanced causal inference techniques to further improve root cause attribution accuracy in complex distributed system scenarios.

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.
- **Use of AI Tools:** The author(s) declare that no generative AI or AI-assisted technologies were used in the writing process of this manuscript.

References

[1] David Bernstein, "Containers and Cloud: From LXC to Docker to Kubernetes," IEEE Cloud Computing, 2014. Available: https://ieeexplore.ieee.org/document/7036275

[2] Varun Chandola et al., "Anomaly Detection: A Survey," ACM Computing Surveys, 2009. Available: https://www.researchgate.net/publication/220565847_Anomaly_Detection_A_Survey

[3] Yingnong Dang et al., "AIOps: Real-World Challenges and Research Innovations," IEEE Network, 2018. Available: https://ieeexplore.ieee.org/document/8802836

[4] Xue Yang et al., "ADT: Time series anomaly detection for cyber-physical systems via deep reinforcement learning," ScienceDirect, 2024. Available: https://www.sciencedirect.com/science/article/pii/S0167404824001263

[5] Hui Kang et al., "Container and Microservice Driven Design for Cloud Infrastructure DevOps," IEEE International Conference on Cloud Computing Technology and Science, 2016. Available: https://ieeexplore.ieee.org/document/7484185

[6] Albert Bifet et al., "Machine Learning for Data Streams with Practical Examples in MOA," ResearchGate, 2019. Available: https://www.researchgate.net/publication/323993210_Machine_Learning_for_Data_Streams_with_Practical_Examples_in_MOA

[7] I. Gethzi Ahila Poornima et al., "Anomaly detection in wireless sensor network using machine learning

algorithm," ScienceDirect, 2020. Available: https://www.sciencedirect.com/science/article/abs/pii/S0140366419309673

[8] Miguel G. Xavier et al., "Performance Evaluation of Container-Based Virtualization for High Performance Computing Environments," IEEE Xplore, 2013. Available: https://ieeexplore.ieee.org/document/6498558

[9] Mohiuddin Ahmed et al., "A survey of network anomaly detection techniques," ScienceDirect, 2016. Available: https://www.sciencedirect.com/science/article/abs/pii/S1084804515002891

[10] Robin Sommer, Vern Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," IEEE Symposium on Security and Privacy, 2010. Available: https://ieeexplore.ieee.org/document/5504793