International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

AI-Driven Root Cause Analysis for Cloud-Native Systems Using Logs, Metrics, and Distributed Traces

Madhava Reddy Muli
Independent Researcher, USA.

Abstract - Cloud-native systems pose big data challenges in root cause analysis because of distributed microservices, dissimilar telemetry, and interwove service relationships. The framework proposed by this research suggests the use of AI to help with systematically diagnosing incidents by combining logs, infrastructure metrics, traces distributed among devices deployed, and information on deployment. Experimental evaluation is aided by a synthetic data set of 6,000 observations and 120 incidents plus six categories of root causes. The stages of the data preprocessing, statistical analysis, anomaly detection, feature engineering at the incident level, and validation of the evidence are performed with Python. The comparative analysis of the four classification methods, that is, Majority Baseline, Logistic Regression, Random Forest, and Gradient Boosting, is examined. The results show a 100% accuracy and F1-score of both Logistic Regression and Random Forest and 96.67-percent accuracy of Gradient Boosting. Correlation analysis indicates that there is a strong correlation between request latency and error rates (r=0.77), and ablation experiments create the significance of the operational metrics. The framework also prepares organized evidence to further reasoning with the assistance of LLM, though the validation of production is needed to evaluate the feasible generalizability.

Keywords *- Cloud-Native Systems; Root Cause Analysis; Artificial Intelligence; Machine Learning; Distributed Tracing; Log Analysis; Observability Metrics; Anomaly Detection; Microservices; Evidence Validation.*

I. INTRODUCTION

A. Background and Contexts

Cloud-native frameworks are according to microservices as well as containers, and are distributed, complicating along with time-consuming incident diagnosis [1]. Existing origin analysis tools are not able to correlate heterogeneous observable data involving metrics, logs, deployment histories, distributed traces and also dependency graphs [2]. “Large Language Models (LLMs)” are promising that can assist to interpret and combine sources of various evidence [3]. Applications of evidence-led LLM-aided frameworks have potential to improve overall accuracy of diagnosing, decrease investigation time, and create explainable root-cause hypotheses, for assuring adequate incident management within the dynamic cloud world.

B. Problem Statement

Determining root cause of cloud service outages is still a difficult procedure while fragmented diagnostic data, complicated dependencies among services as well as ever-changing operation environments [4]. Currently automated approaches are not contextual, whereas more traditional methods considering LLM can potentially lead to formation of unsubstantiated conclusions. Systematic validation of the evidence is required to relate observed anomalies with underlying failures with some reliability [5]. Thus, an advanced structure is required to assess valid, traceable as well as contextualized root-cause hypotheses and minimize diagnostic uncertainty along with operational latencies.

C. Research Aim and Objectives

Aim

The purpose of the research is to implement an evidence-based framework informed by heterogeneous data on observability enabling identification, validation, and explanation of root causes of cloud-native incidents through the use of a LLM.

Objectives

- To explore cloud-native incident patterns analysis of logs and metrics, distributed traces, deployment histories as well as service dependencies with Python associated analytical methods.
- To introduce a framework of heterogeneous observability evidence correlation with the assistance of LLM and searching possible root causes of the incidents.

International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

- To adopt mechanisms of evidence-validation assuring explainable, traceable as well as context-supported diagnostic hypotheses.
- To test effectiveness of various frameworks through utilizing Python-based experiments assessing diagnostic reliability, accuracy, computational efficiency as well as investigation time.

D. Research Contribution

The research presents an evidence-based LLM-aided model of root causes within cloud-native frameworks, through combined observability evaluation. Techniques according to Python make it possible to systematically correlate logs, traces, metrics, deployment history. Experimental analysis defines comprehension of efficiency, diagnostic accuracy, reliability as well as less time of analyzing incidents, spreading throughout distributed frameworks [6]. The research presents mechanisms of evidence validation in producing reasons and motives of diagnosis.

II. RELATED WORK

Cloud Native Incident Pattern Analysis Using Python Based Observability Techniques

Increasing trends within cloud-native incident analysis are to analyze distributed traces, metrics, logs, deployment histories as well as service dependencies utilizing Python-associated approaches [7]. Existing approaches enable anomaly detection along with recognition of operational trends according to machine learning as well as statistical analysis [8]. Fragmented timestamps, inconsistent data structure as well as dynamic interaction of microservices, however, limit quality of diagnostics [9]. Traditional approaches establish symptoms without causal connections. Combination of heterogeneous observability sources with Python-led analytical pipeline is crucial to enhance incident interpretation, along with root-cause investigation methodically amid complicated distributed cloud contexts.

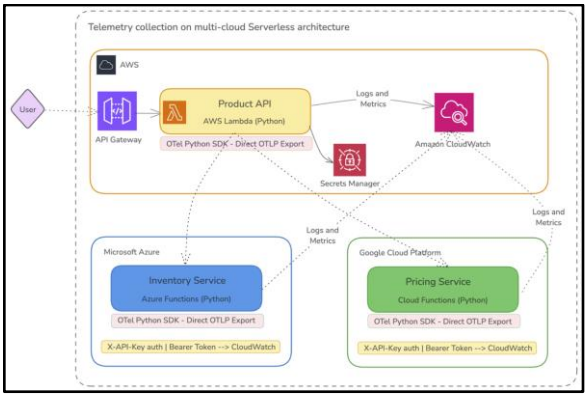


Fig 1. MultiCloud Observability

Large Language Model Assisted Correlation of Heterogeneous Cloud Observability Evidence

"Large Language Models (LLMs)" present new abilities to match irregular observability displays as well as elucidate complicated cloud service breakdowns [10]. Current studies point out capabilities at contextual reasoning, incident summarization as well as generating hypotheses automatically. However, hallucinated explanations, low level of causal awareness as well as inconsistent attribution of evidence degrade diagnostic credibility [11]. Moreover, handling huge amounts of telemetry poses a load on computations as well as context [12]. Incorporating systematic evidence retrieval, dependency-conscious reasoning, and also verifiability of references to sources can reinforce LLM-assisted investigations to become more transparent and operationally meaningful towards recognizing root-cause(s) within cloud-native contexts.

International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

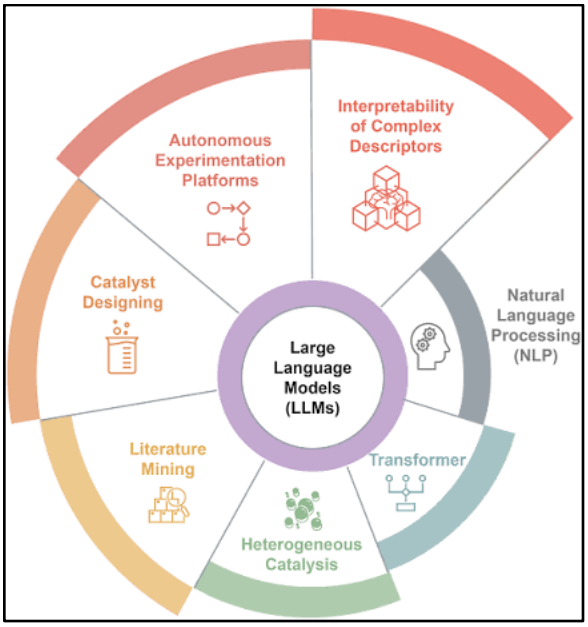


Fig 2. Large Language Models for Heterogeneous Catalysis

Evidence Validation Mechanisms for Explainable and Traceable Root Cause Identification

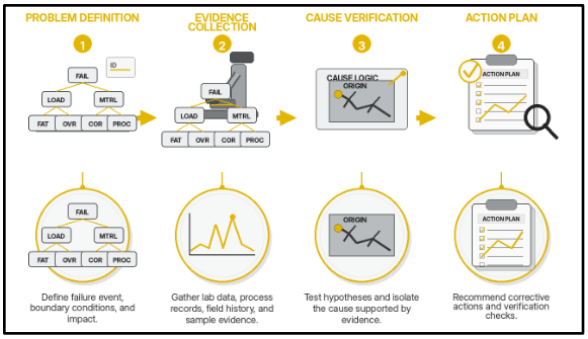


Fig. 3. Root cause verification

Root cause verification is a crucial concept in cloud-native systems to achieve reliable and explainable root cause assurance. Current testing methodologies tend to produce possible cause-of-failure theories without the plausible proof of the links between these theories and the discovered evidence of operations [13]. Recent research highlights the significance of logging with metrics, distributed traces, deployment histories and service dependency graphs to enhance diagnostic transparency [14]. The contextual interpretation and generation of hypotheses can be supported by Large Language Models (LLMs). Hallucinations and unfounded causal assumptions can lower diagnostic reliability [15]. Mechanisms of evidence validation by means of timestamp alignment, correlation of anomalies, dependency analysis and confidence scoring may enhance the validity of generated hypothesis. Python tools of analysis also assist in automated extraction of evidence, consistency tests and that of potential root cause failures [16]. Combining these mechanisms with the use of the LLM-assisted reasoning helps provide the traceable diagnostic explanations with observable behavior of the system [17]. This is likely to make it less uncertain, more interpretable and capable of building a higher level of confidence in automated root cause analysis processes in complex distributed cloud networks.

Performance Evaluation of LLM-Assisted Cloud Incident Root Cause Analysis

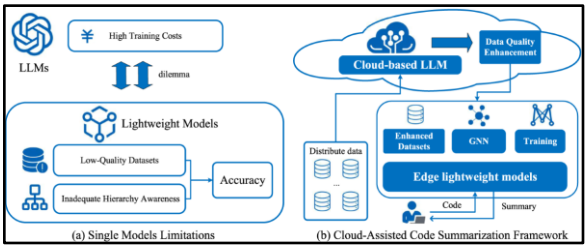


Fig. 4. LLM-Assisted Cloud Incident Root Cause Analysis

Performance evaluation has an important role to play in assessing the efficiency of the root cause analysis systems of LLM-aided frameworks in a cloud-native setting. The diagnostic accuracy, anomaly detection capabilities, computational efficiency, and incident investigation time are typically evaluated on a range of diagnostic methods by existing studies on experimental observability datasets [18]. Evaluation approaches written in Python provide the possibility to systematically compare traditional statistical models with machine learning models and natural language processing models based on LLMs [19]. Precision, recall, F1-score, root cause identification accuracy, respondent latency, and computational overhead are a few performance indicators that can be used to measure diagnostic performance in an objective way. Yet, incidence complexity, quality telemetry, and service dependencies could lead to differences in the performance of a framework between distributed environments [20]. Evidence-based assessment also takes into consideration the consistency in the explanation, reliability of the hypothesis and traceability of diagnostic conclusions

International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

[21]. The quantitative performance measurement in conjunction with the evidence validation can be used to perform a thorough evaluation of automated incident investigation system [22]. These assessments allow determining the limitations of the operation and can determine the practicality of the application of the LLM-assisted cloud-native infrastructure-based root cause analysis.

Research Gap

The current methods of cloud-native root cause analysis do not fully integrate heterogeneous evidence of observability with sound reasoning based on the use of LLMs. The existing frameworks have some issues related to their fragmented telemetry, lack of support in their diagnostic hypotheses, inability to validate the evidence and lack of explainability. Moreover, methodical reviews of diagnostic accuracy, computational cost, time of investigation, and reliability of hypotheses are minimal. Consequently, a translucent-based architecture that combines Python based observability analysis, bleeding machine learning aided correlation, and verifiable validation is needed to enhance reliability and diagnosis of cloud emergencies.

III. PROPOSED FRAMEWORK

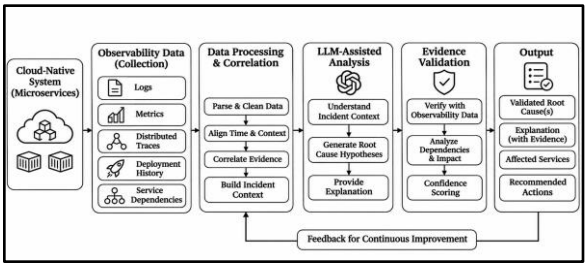


Fig. 5. Proposed Framework

The figure demonstrates an AI-based root cause analysis framework of cloud-native systems based on heterogeneous observability data. Firstly, there are operational information that is gathered based on microservices, which includes logs, metrics, distributed traces, deployment histories, and service dependencies. The gathered evidence is processed using preprocessing, alignment of timestamps, and correlation of evidence to form relevant incident contexts. Moving on, Large Language Models (LLMs) examine the correlated information to comprehend pattern of incidents, come up with possible root cause hypotheses, and offer contextual explanations [23]. An evidence validation phase tests these hypotheses in respect to known observability data, tests service dependencies and failure consequences, and assigns levels of confidence to enhance diagnostic accuracy. The framework subsequently yields validated root causes, evidence-based explanations, affected service details and suggested corrective measures [24]. Also, there is a feedback mechanism that allows an iterative improvement of incident analysis and the accuracy of diagnosis. In general, the framework is designed to increase the explainability, decrease the investigation time, minimize unsubstantiated inferences, and optimize the automated incident diagnosis in a complex distributed cloud-native environment.

IV. RESEARCH METHODOLOGY

Research Design and Experimental Framework

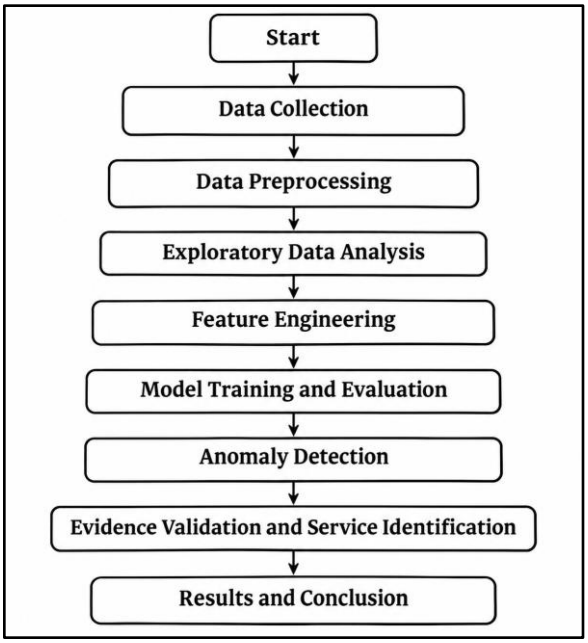


Fig. 6. Methodological Flow

The research followed a quantitative, experimental approach to determine AI-guided root cause analysis (RCA) in cloud-native systems using heterogeneous observability data. The suggested framework combines the analysis of logs, the measurements of infrastructure, distributed tracing, the data about the deployments, dependencies between services, and other characteristics to outline possible causes of

International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

operational incidents. Python also offers a computational platform for data preprocessing, exploratory analysis, anomaly detection, machine classification learning, and to the evaluation of the performance [25]. The methodology is structured in form of a workflow, which includes the preparation of the dataset, feature engineering, predictive modeling, evidence correlation, hypothesis validation, and diagnostic evaluation. This strategy encourages methodical research on cloud service failures and also focuses on diagnostic accuracy, interpretability and reproducibility.

Synthetic Dataset Preparation and Preprocessing

The experimental dataset has 6,000 synthetic observability records of 120 cloud-native incidents between various microservices. On average, there are 28 attributes on each record that characterize the state of operation such as CPU, memory consumption, request response time, error rates, throughput, database connection, network latency, log severity, trace identifiers, deployment events, and a root cause name. The data set presents six categories of root causes which include CPU saturation, memory leaks, connection exhaustion, network latency, deployment regression and dependency failure.

Timestamps conversion, numerical property validation, examination of missing values and determination of duplicates all fall under the umbrella idea of data preprocessing. Other indicators are error level logs, warning messages and loss of service of an HTTP server. Such changes provide standard analytical outputs and integration among heterogeneous sources of observability.

Exploratory Data Analysis and Observability Correlation

Exploratory data analysis studies incident distributions, occurrence of anomalies, patterns of resource utilization and error characteristics at the service level. Descriptive statistics summarize operational behavior and correlations heatmaps show statistical relationships among infrastructure and application metrics. Boxplots are used to compare the use of CPU, memory and the latency of requests during normal and abnormal conditions.

Distributed tracing analysis explores the number of traces, the number of spans, services involved in a trace, and latency properties. The deployment analysis compares relationships between new software releases and abnormalities in operation [26]. These observations are aggregated into single diagnostic representations using incident-level aggregation to allow the framework to consider numerous sources of evidence at the same time.

Feature Engineering and Root Cause Classification

The methodology builds incident-level features based on the statistical measures, such as the means and the extreme values, the standard deviations and the temporal variations. Numerical measures are supplemented by frequency of logs, deployment indicators, trace features and dependency measures [27]. Definition Ground-truth root cause categories are only used as a target of classification.

Stratified sampling is used to split the dataset into 75%training and 25% testing incidents. This level of partition guarantees that at the incident level, a certain observation cannot occur in both subsets of incident.

Four classification methods are compared including: *Majority Baseline, Logistic Regression, Random Forest and Gradient Boosting.* Cross-validation (5 folds) evaluates the effectiveness of the Random Forest on the training partition. The importance analysis of features defines influential attributes of observability and modality ablation is used to investigate the individual impact of various evidence types.

1. Majority Baseline Classifier

The Majority Baseline is used to predict the occurrence of the most commonly occurring root cause category within training data regardless of the observability features of the input.

$$\hat{y} = arg\,max \sum_{i=1}^{N} I\,(y_i = C)$$

Where:

- $\hat{y}$ represents the predicted root cause category.
- C denotes the set of root cause categories.
- N represents the total number of training incidents.
- y_i denotes the actual root cause label of incident i.

International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

- $I(\cdot)$ is an indicator function that equals 1 when the condition is satisfied and 0 otherwise.

This model establishes a reference performance level against which the effectiveness of advanced machine learning approaches is evaluated.

2. Logistic Regression Classifier

Multinomial Logistic Regression estimates the probability of each root cause category using a weighted combination of observability features and the SoftMax function [28].

$$P(y = c \mid X) = \frac{exp\ (w_c{}^T X + b_c)}{\sum_{K=1}^{K} exp\ (w_c{}^T X + b_c)}$$

Where:

- $P(y = c \mid X)$ Represents the probability that an incident belongs to root cause category c.
- x denotes the incident-level feature vector derived from logs, metrics, traces, and deployment information.
- w_c represents the learned coefficient vector for category c.
- b_c represents the intercept for category c.
- K denotes the total number of root cause categories, which is six in this research.

The category with the highest estimated probability is selected as the predicted root cause.

3. Random Forest Classifier

Random Forest combines predictions from multiple decision trees trained on different bootstrap samples and randomized feature subsets [29]. The final root cause classification is determined through majority voting.

$$\hat{y} = arg\ max \sum_{t=1}^{T} I\ (h_t(x) = C)$$

Where:

- $\hat{y}$ denotes the predicted root cause category.
- T represents the total number of decision trees in the ensemble.
- $h_t(x)$ denotes the classification output of the t-th decision tree.
- x represents the incident-level observability feature vector.
- C represents the set of possible root cause categories.
- $I(\cdot)$ indicates whether an individual tree predicts category c.

In this research, the Random Forest model uses 200 decision trees to identify nonlinear relationships between operational measurements and root cause categories.

4. Gradient Boosting Classifier

Gradient Boosting constructs an ensemble of weak learners sequentially, with each subsequent learner minimizing the residual classification loss of the preceding ensemble. For multiclass classification, the model updates category-specific decision scores.

$$Fm, c(x) = F_{m-1,c}(x) + \eta h_{m,c}(x)$$

Where:

- $F_{m,c}(x)$ represents the ensemble prediction score for root cause category c at boosting iteration m.
- $F_{m-1,c}(x)$ denotes the prediction score from the previous boosting iteration.
- η represents the learning rate controlling the contribution of each weak learner.
- $h_{m,c}(x)$ denotes the fitted regression tree for category c at iteration m.
- m represents the boosting iteration.

Anomaly Detection and Root Service Identification

Isolation Forest offers unsupervised anomaly detection based on measurements of resource utilization and application performance. The model is trained on operational patterns and is used to spot an unusual behavior in held-out incidents. Its predictions are gauged on synthetic anomaly labels.

Root service identification is an evidence-based ranking that applies to single microservices. The ranking uses the deviations of metric values on a normalized basis, log frequencies of errors, and HTTP sadly indicators. Top-1 and top-3 accuracy:

International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

both determine whether the true originating service is among the top- ranked candidates or not.

Evidence Validation and LLM-Assisted Diagnostic Preparation

Evidence validation compares root cause hypotheses, which were predicted, to operational indicators which are preset. Verifying CPU saturation, memory leaks, database exhaustion, network abnormalities, deployment regressions, dependency failures have associated verification conditions. Observable evidence assigns supported or review required classifications to hypotheses.

The framework also produces the structured and evidence based prompts which have diagnostic hypotheses, estimates of confidence, operative measurements; and the fields that need to be explained. These inputs help in later reasoning with the assistance of LLM, but the existing model does not generate an LLM or analyze generated explanations.

Performance Evaluation and Research Reliability

Other metrics used to evaluate classification performance include accuracy, macro-averaged precision, recall, F1-score, cross-validation and confusion matrices. The efficiency of computations is measured by the length of training and prediction latency. Other diagnostic indicators are evidence support coverage and accuracy in root service ranking.

Experiments are reproducible in random seeds, and evaluation procedures are consistent. As the data is artificial, the results indicate a performance in a controlled environment and not generalizability in a real environment. To confirm the effectiveness of operations, additional analysis with the production incident records and real-life inference of LLM should be done.

Pseudo Code

```
1. Load and preprocess Data.
2. Extract statistical, temporal,
and dependency features.
3. Aggregate observations at the
incident level.
4. Split incidents into training
and testing datasets.
5. Train LR, RF, and GB
classification models.
6. Detect anomalies using
Isolation Forest.
7. Predict root cause RC using
trained classifiers.
8. Correlate telemetry and
validate supporting evidence E.
9. Rank potentially responsible
microservices.
10. Generate structured evidence
for LLM-assisted diagnosis.
11. Evaluate accuracy, precision,
recall, F1-score, and latency.
12. Return RC, E, and evaluation
results.
```

The algorithm combines heterogeneous cloud observability with data and uses machine learning methods to identify anomalies, categorize root causes of an incident, and categorize potentially causing services. Diagnostic interpretation is helped by evidence correlation and rule-based validation, and structuring incident information allows further processed by the LLM-assisted analysis and evaluation of the performance.

V. RESULTS AND DISCUSSION

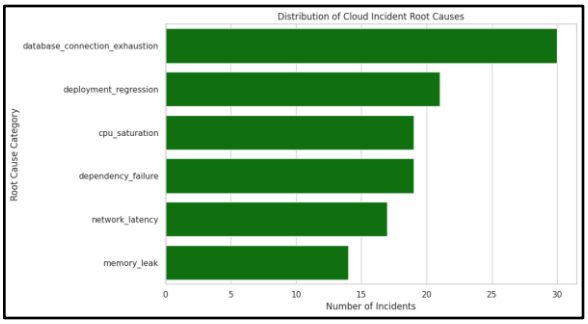


Fig. 7. Distribution of Cloud Incident Root Causes

Connection to the database exhaustion has the highest number of 30 incidents (25%), deployment regression (21), CPU saturation (19), dependency failure (19), network latency (17), and memory leaks (14).

International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

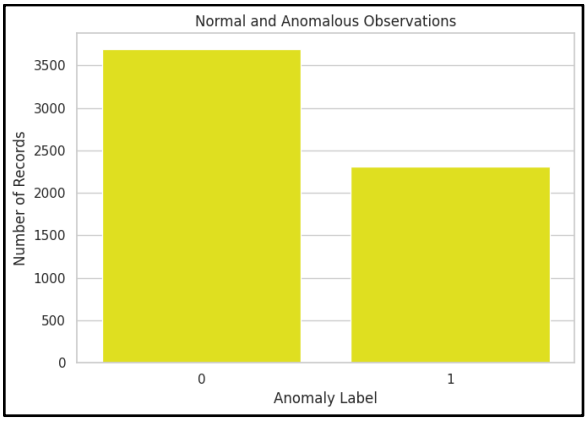


Fig. 8. Distribution of Normal and Anomalous Observations

Out of a total of 6,000 observations, 3,694 (61.57) are regular behavior and 2,306 (38.43) are oddities reflecting moderate particularly the abnormality of operational conditions in the data set.

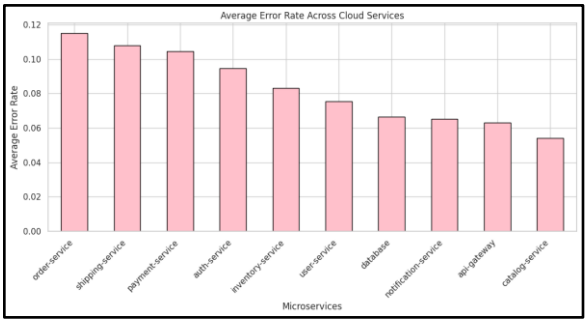


Fig. 9. Average Error Rate Across Cloud Services

The worst error rate is found in order-service (0.115) then shipping-service (0.108) and payment-service (0.105) but the worst is at the lowest level in catalog-service (0.054) which shows heterogeneous service reliability.

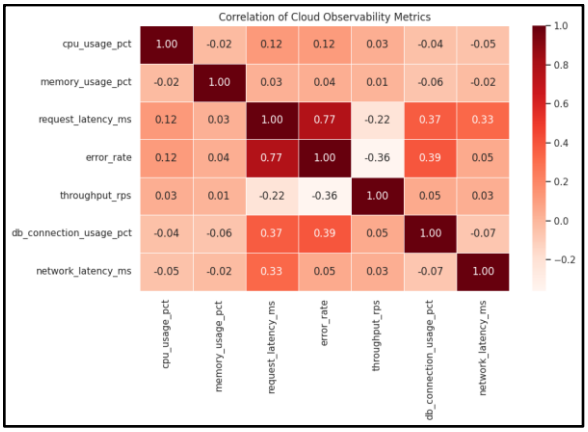


Fig. 10. Correlation of Cloud Observability Metrics

Error rate is strongly positively correlated with request latency (r=0.77), whereas throughput and errors have a negative correlation (r= -0.36). Moderate correlations with latency (r=0.37) and errors (r=0.39) are found with database connection use.

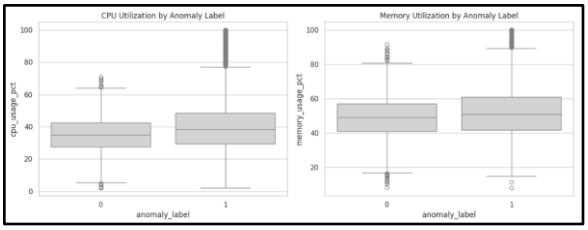
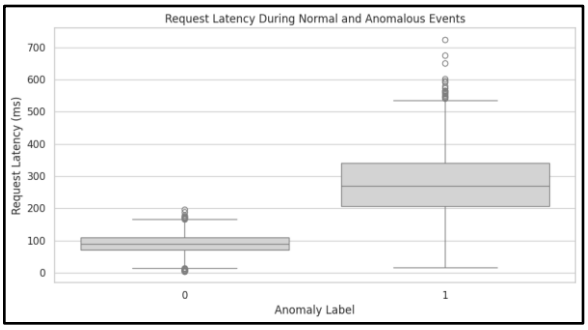


Fig. 11. CPU and Memory Utilization by Anomaly Label

Abnormal observations have more CPU variability of up to 100 percent usage and memory counter of up to 100 percent as well. When there are operational anomalies, the pattern of resources saturation is observed by the same distribution, which is relatively narrow in normal observations.



International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

Fig. 12. Request Latency during Normal and Anomalous Events

Observations made in normal conditions show median request latency of about 90 ms, but those in anomaly conditions have values of up to 270 ms, which is significant performance degradation of the application.

```
DEPLOYMENT IMPACT
                    cpu_usage_pct  memory_usage_pct  request_latency_ms  \
deployment_recent
0                       38.819352         50.999019          164.604543
1                       35.859712         50.777862          149.292658

                    error_rate  throughput_rps  db_connection_usage_pct  \
deployment_recent
0                     0.087952      179.614565                34.149513
1                     0.071556      182.026134                30.354693

                    network_latency_ms
deployment_recent
0                        23.772971
1                        23.203262
```

Fig. 13. Deployment Impact Analysis

There is no aggregate deployment-associated decline in this dataset: the mean latency of non-deployment observations (164.60 ms) is larger than that of recent-deployment observations (149.29 ms) and the error rates are higher in non-deployment observations (0.08795) than in recent-deployments observations (0.07156).

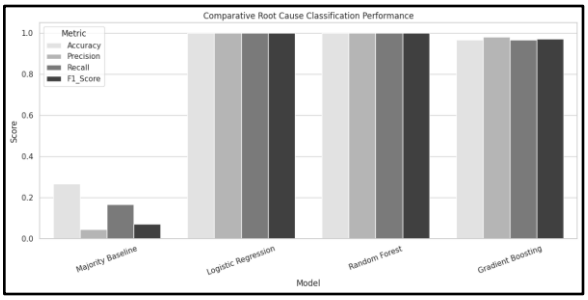


Fig. 14. Comparative Root Cause Classification Performance

Logistic Regression and Random Forest have 100 percent accuracy, precision, recall and F1-score. Gradient Boosting has 96.67% accuracy and 97.17% F1-score, which is significantly better than 26.67% baseline accuracy.

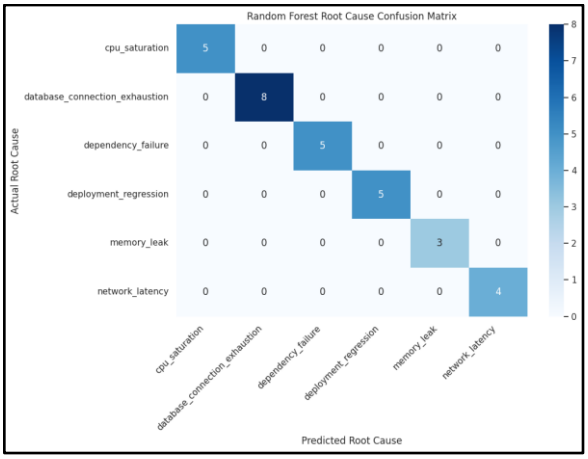


Fig. 15. Random Forest Root Cause Confusion Matrix

The random forest classifies all 30 test cases in 6 categories; this is 100 percent accuracy. The diagonal has 5, 8, 5, 5, 3 and 4 incidents in their place properly.

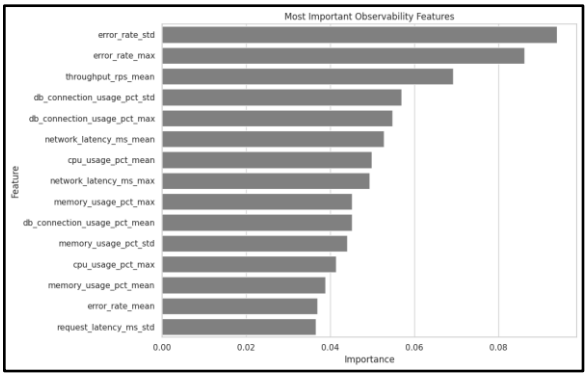


Fig. 16. Most Important Observability Features

The maximum error rate (0.086), mean throughput (0.069), and error-rate segmentation (0.094) rank as the most diagnostically important features in terms of standard deviation.

```
OBSERVABILITY MODALITY ABLATION
       Modality  Number_of_Features  Accuracy  F1_Score
0       Metrics                  25  0.966667  0.966330
1          Logs                   5  0.300000  0.305556
2        Traces                   2  0.266667  0.070175
3    Deployment                   1  0.333333  0.162142
4  Dependencies                   4  0.166667  0.134259
```

Fig. 17. Observability Modality Ablation Analysis

The metrics have the highest accuracy and F1-score, performing better than logs (30 per cent), traces

International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

(26.67 per cent), deployment (33.33 per cent) and dependencies (16.67 per cent): numerical operational metrics has the best operational predictive information on its own.

Discussion

TABLE I. Comparative Performance Evaluation of Root Cause Classification Models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Majority Baseline	26.67	4.44	16.67	7.02
Logistic Regression	100.00	100.00	100.00	100.00
Random Forest	100.00	100.00	100.00	100.00
Gradient Boosting	96.67	98.15	96.67	97.17

The experimental findings illustrate the importance of machine learning in classifying root causes in clouds that are native. Logistic Regression and Random Forest have 100% accuracy and F1-score whereas Gradient Boosting has an accuracy of 96.67%. It is found that there is a strong correlation between request latency and error rates (r=0.77). The analysis conducted through ablation determines operational metrics to be the most informative features. Then again, artificial data and reduced incident patterns do not allow generalizations, which require real-life validation and evaluation by LLM.

VI. LIMITATIONS AND FUTURE WORK

Research Limitations

- ***Dataset Dependency:*** The study refers to 6,000 naturally reproduced observations of observability data of 120 cloud-native incidents. The dataset is capable of controlled experimentation, but due to the complexity, unpredictability, and heterogeneous patterns of failure in real distributed clouds, it may not fully reproduce the complexity and heterogeneity of a real distributed cloud environment.
- ***Limited LLM and Real-Time Verification:*** The framework uses machine learning classifiers and rule-based evidence validation, but does not run large language model inferences. As a result, the effectiveness of the reasoning, quality of diagnostic explanation, scalability, and real-time operational performance of the Lean Linear Model has not been experimentally verified.

Future Work

Future studies will enlarge the suggested framework by utilizing real-life cloud observability data that involves intricate microservice failures and unequal operational states. The large language models will be integrated to facilitate reading between the lines and offering evidence-based arguments and requesting robotically-generated remedies. High-end graph neural networks and causal inference will be used to improve dependency-aware localization of root cause [30]. Future research will measure scalability, diagnostic latency, explainability and generalization across Kubernetes-based production systems in dynamically changing loads, and failure conditions.

VII. CONCLUSION

The study proposes an AI-based root cause analysis architecture of cloud-native systems based on the combination of logs, metrics, distributed traces, and deployment data. Classification performance is strongly studied with a high accuracy and F1-score of 100% and 96.67% for Logistic Regression and Random Forest and Gradient Boosting, respectively, on 6,000 synthetic observations. The relationship between request latency and error rates are found to be significant under correlation analysis and modality ablation is used to point out the predictive use of the operational measures. Structured incident diagnosis is once again facilitated by evidence validation. However, conditions of synthetic data and unproven

International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

LLM inference also imply that further realistic experimentation will be required before determining the workability and feasibility in practice under production conditions.

REFERENCES

[1] Söylemez, M., Tekinerdogan, B. and Kolukısa Tarhan, A., 2022. Challenges and solution directions of microservice architectures: A systematic literature review. *Applied sciences*, *12*(11), p.5507.

[2] Kotenko, I., Gaifulina, D. and Zelichenok, I., 2022. Systematic literature review of security event correlation methods. *Ieee Access*, *10*, pp.43387-43420.

[3] Singhal, K., Azizi, S., Tu, T., Mahdavi, S.S., Wei, J., Chung, H.W., Scales, N., Tanwani, A., Cole-Lewis, H., Pfohl, S. and Payne, P., 2023. Large language models encode clinical knowledge. *Nature*, *620*(7972), pp.172-180.

[4] Donta, P.K., Murturi, I., Casamayor Pujol, V., Sedlak, B. and Dustdar, S., 2023. Exploring the potential of distributed computing continuum systems. *Computers*, *12*(10), p.198.

[5] Teh, H.Y., Kempa-Liehr, A.W. and Wang, K.I.K., 2020. Sensor data quality: A systematic review. *Journal of Big Data*, *7*(1), p.11.

[6] Badidi, E., 2023. Edge AI for early detection of chronic diseases and the spread of infectious diseases: opportunities, challenges, and future directions. *Future Internet*, *15*(11), p.370.

[7] Nguyen, H.X., Zhu, S. and Liu, M., 2022. A survey on graph neural networks for microservice-based cloud applications. *Sensors*, *22*(23), p.9492.

[8] Narapareddy, V.S.R. and Yerramilli, S.K. (2022) 'Risk-oriented incident management in ServiceNow Event Management', International Journal of Engineering Technology Research & Management, 6(7), pp. 134–149.

[9] Pradhan, M.M.P., 2023. Integration of MES, SAP, And PI Historian Platforms for Real-Time Industrial Asset Performance Management. *Review of Applied Science and Technology*, *2*(04), pp.393-445.

[10] Mooney, A., Tsotsoros, C.E., Earl, J.K., Hershey, D.A. and Mooney, C.H., 2021. Enhancing planning behavior during retirement: Effects of a time perspective based training intervention. *Social Sciences*, *10*(8), p.306.

[11] Arumilli, S. V. T. (2022). Engineering resilient high-throughput pipelines: Design patterns for retry mechanisms and failure recovery.

[12] Gkonis, P., Giannopoulos, A., Trakadas, P., Masip-Bruin, X. and D'Andria, F., 2023. A survey on IoT-edge-cloud continuum systems: Status, challenges, use cases, and open issues. *Future Internet*, *15*(12), p.383.

[13] Meyfroidt, P., 2016. Approaches and terminology for causal analysis in land systems science. *Journal of Land Use Science*, *11*(5), pp.501-522.

[14] Waldner, D., 2015. What makes process tracing good? Causal mechanisms, causal inference, and the completeness standard in comparative politics. *Process tracing: From metaphor to analytic tool*, pp.126-52.

[15] Neurohr, C., Westhofen, L., Butz, M., Bollmann, M.H., Eberle, U. and Galbas, R., 2021. Criticality analysis for the verification and validation of automated vehicles. *IEEE Access*, *9*, pp.18016-18041.

[16] Ikram, A., Chakraborty, S., Mitra, S., Saini, S., Bagchi, S. and Kocaoglu, M., 2022. Root cause analysis of failures in microservices through causal discovery. *Advances in Neural Information Processing Systems*, *35*, pp.31158-31170.

[17] Vangoor, V.K.R., 2022. Autonomous DevOps Infrastructure: AI-Driven Lifecycle Management of Large Scale Linux Server Ecosystems. *Journal of Management and Science*, *12*(4), pp.156-163.

[18] Allala, U., 2020. Safe Autonomous Remediation of Cloud Infrastructure Using Large Language Models. *International Journal of Modern Innovations and Emerging Trends*, *3*(1), pp.01-19.

[19] Ghanta, S., 2023. From observability to understanding: Automated incident triage using large language model reasoning over logs, metrics, and traces. *International Journal of Engineering & Extended Technologies Research (IJEETR)*, *5*(5), pp.7242-7249.

International Journal of
Engineering Research and Education
Peer Reviewed, Referred & Indexed Journal
ISSN: 3143-4428 www.ijerae.com Original Research Paper

[20] Kıcıman, E., Ness, R., Sharma, A. and Tan, C., 2023. Causal reasoning and large language models: Opening a new frontier for causality. *arXiv preprint arXiv:2305.00050*.

[21] Wei, L., 2019. AI-Augmented DevSecOps Pipelines: Enabling Continuous Security Integration in Large-Scale Software System. *American International Journal of Computer Science and Technology*, *1*(5), pp.1-9.

[22] Zhao, C., Zhang, S., Huang, Z., Jin, W., Sun, Y., Pei, D., Zhang, C., Lin, Q., Bansal, C., Rajmohan, S. and Ma, M., 2018. Can language models go beyond coding? assessing the capability of language models to build real-world systems. *ACM Transactions on Software Engineering and Methodology*.

[23] Sandoval, G., Pearce, H., Nys, T., Karri, R., Garg, S. and Dolan-Gavitt, B., 2023. Lost at c: A user study on the security implications of large language model code assistants. In *32nd USENIX Security Symposium (USENIX Security 23)* (pp. 2205-2222).

[24] Belzner, L., Gabor, T. and Wirsing, M., 2023, October. Large language model assisted software engineering: prospects, challenges, and a case study. In *International conference on bridging the gap between AI and reality* (pp. 355-374). Cham: Springer Nature Switzerland.

[25] Thota, M.R., 2018. Transforming database leadership in the era of cloud-native automation and resilient operations. *International Journal of Technology Management & Humanities*.

[26] de Paiva Menezes, G.N.O., 2023. *AI-Driven Software Development How has the rise of ChatGPT Impacted GitHub Repositories?-Impact of ChatGPT on Code Quality* (Master's thesis, Universidade NOVA de Lisboa (Portugal)).

[27] Oppenlaender, J. and Hämäläinen, J., 2023. Mapping the challenges of HCI: An application and evaluation of ChatGPT and GPT-4 for mining insights at scale. *arXiv preprint arXiv:2306.05036*.

[28] Mejia-Ruiz, G.E., Rosero-García, E.E. and Zamora-Mendez, A., 2022. Machine Learning for Power Systems: A Critical Review Bridging the Interpretability and Performance Gap. *Available at SSRN 7051842*.

[29] Varas, J., Coronel, B.V., VILLAGRáN, I.G.N.A.C.I.O., Escalona, G., Hernandez, R., Schuit, G., DURáN, V.A.L.E.N.T.I.N.A., Lagos-Villaseca, A., Jarry, C., Neyem, A. and Achurra, P., 2023. Innovations in surgical training: exploring the role of artificial intelligence and large language models (LLM). *Revista do Colégio Brasileiro de Cirurgiões*, *50*, p.e20233605.

[30] Wu, Y., 2023. Cybersecure by Design: Managing Adversarial Risk and Resilience in AI-Enabled Critical Infrastructure. *The Computertech*, pp.48-55.