2026 Volume: 02 Issue: 06 www.ijrpa.com ISSN 2456-9995 Research Article

International Journal Research Publication Analysis

Page: 01-22

ANOMALY DETECTION IN CLOUD OBSERVABILITY DATA: A REPRODUCIBLE BENCHMARK OF STATISTICAL AND ML-BASED METHODS

Akhil Reddy Mandadi*

Independent, Indian.

Article Received: 17 May 2026
Article Revised: 05 June 2026
Published on: 25 June 2026

***Corresponding Author: Akhil Reddy Mandadi**
Independent, Indian.

DOI: https://doi-doi.org/101555/ijrpa.1157

ABSTRACT

In production cloud environments, a vast amount of observability information is created as logs, metrics, and distributed traces. These multiple telemetry streams contain a wealth of important data about the system but are not only very large, very heterogeneous, and very noisy, but the task of recognizing anomalies remains a constant challenge for the cloud ops team. While many anomaly detection techniques have been suggested, currently most evaluations are done on a small number of techniques or only one data set, making it difficult to find comprehensive evidence of their comparative effectiveness and reproducing most evaluations. This work offers a repeatable benchmark of assessing the statistical, machine learning and large language model (LLM) based anomaly detection techniques applied to various cloud observability datasets. The benchmark includes statistical techniques such as Exponentially Weighted Moving Average (EWMA), Seasonal-Trend Decomposition using Loess (STL), and Cumulative Sum Control Charts (CUSUM), classical machine learning techniques such as Isolation Forest, One-Class Support Vector Machines (OCSVM), and Autoencoders, and emerging LLM-based zero-shot anomaly-detection baselines. Publicly accessible datasets such as HDFS, BGL, Thunderbird, OpenStack, and Hadoop are used for the experiments, all of which are preprocessed and evaluated in the same way. Precision, recall, F1-score, alert lead time, false-positive rate, and the computational overhead and cost-per-alert are used to assess performance. The benchmark provides a structured comparison of the detection approaches and shows patterns of detection method compatibility and incompatibility with the datasets. The work lays the groundwork for reproducible anomaly-detection research, and offers a basis to select monitoring strategies in today's cloud-native systems based on evidence.

KEYWORDS: *Cloud Observability, Anomaly Detection, Machine Learning, Statistical Monitoring, Large Language Models, Benchmarking, Distributed Systems, Reliability Engineering*

I. INTRODUCTION

A. Background and Motivation

These changes in the design and deployment of modern software systems, including microservices, containers, orchestration platforms, and distributed infrastructure, have been reshaped by cloud-native computing. Cloud-native computing has revolutionized the design and deployment of modern software systems, including microservices and containers, orchestration platforms, and distributed infrastructure. Cloud has become a staple for many organizations, providing the essential availability, scalability, and operational resilience for business-critical applications. However, as these systems become increasingly complex, understanding the behavior of applications and the performance of infrastructure has become a key requirement for observability. Logs, metrics, traces and telemetry are data gathered from thousands of interdependent services and infrastructure components [1, 3].

The area of Kubernetes orchestration, service meshes, telemetry instrumentation, and distributed tracing has been enriched with improvements that have led to an abundance and depth of operational data for engineering teams to access in recent years [4], [8]. The service interaction and failure propagation paths have never been more visible thanks to technologies like OpenTelemetry and trace-correlation mechanisms [5, 9]. But it also poses significant analytical challenges for operators who are required to detect meaningful anomalies in the massive flow of diverse observations that are continually produced across distributed environments.

B. Challenges in Cloud Anomaly Detection

Cloud observability data is inherently hard to detect anomalies in because of its dynamic nature of distributed systems. Cloud workloads have irregular traffic patterns, varying resource usage, and are subject to frequent configuration changes that can change workload behavior over time [10], [13]. Therefore, static thresholds can frequently miss out on determining whether a legitimate operation or a true system anomaly is occurring.

The next challenge is the diversity of the observability data. The unstructured textual information is found in a log, the numerical time-series measurements in a metric, and the complex dependency relationships among services in a trace [6], [14]. The different data modalities need different approaches for their pre-processing and modeling, which makes it

difficult to have a unified approach for detecting anomalies. Moreover, the datasets from the observability domain are usually noisy, incomplete, have duplicate events, and are poorly labeled [3], [15].

There's also alert fatigue in operational environments due to excessive false positive alerts. For instance, in large-scale monitoring systems, thousands of alerts can be produced every day, causing engineering teams to become overwhelmed and unable to respond to a critical incident in time [7]. The difficulty lies not only in the accurate detection of anomalies but also in the operational value of alerts to act on, with as little intervention as possible.

C. Limitations of existing Benchmark Studies

Although there is a significant amount of research, there is not a lot of comparative evaluation. Most studies evaluate anomaly detection approaches with a single dataset and a few algorithms, or using a customized pipeline that can be difficult to replicate [16]. Therefore, it is often difficult to report performance in isolation rather than generalize it to every cloud environment.

The published research in the field of observability often focuses on collecting or tracing observables and performance analysis, with little focus on comparing the techniques of anomaly detection systematically [1], [14]. Likewise, research on the automated management of incidents or root cause analysis tend to take as a given the anomaly signals, without considering their means of generation or their evaluation [6], [7], [9]. This leaves a gap between the methods used for anomaly detection and the actual operation of the cloud service.

D. Research Objectives

Our main goal in this study is to create a repeatable standard to compare the capabilities of the methods used to detect anomalies in various cloud observability datasets. The benchmark aims to create an equal experimental setting to allow for a fair comparison between all the statistical models, classical machine learning models, deep learning models, and the new LLM-based zero-shot detectors.

The second goal is to characterize patterns of compatibility between datasets and methods by studying the behavior of various algorithms under different observability conditions. Other objectives include testing operational parameters like alert lead time, alert cost, and recommending standardization of preprocessing, evaluation of alerts, and benchmarking workflows to facilitate reproducibility [18] [20].

E. Main Contributions

This work consists of four main contributions. First, it provides a common framework for

measuring the performance of over a dozen anomaly-detection methods on a variety of public observability data sets. Second, it ensures a uniform procedure of pre-processing and evaluation; this ensures reproducible and cross-study comparable results. Third, it includes LLM-based zero-shot anomaly-detection baselines, as well as statistical and machine learning methods, to provide a comprehensive evaluation of new methods. Fourth, it offers a thorough exploration of the detection accuracy, operational efficiency and patterns of dataset compatibility in the context of cloud-native systems, service-mesh environments, CI/CD-driven infrastructures and large-scale distributed applications [11, 12, 17, and 19]. Together, these contributions provide more demanding assessment and help in the evidence-driven choice of anomaly detection strategies for today's cloud observability platforms.

II. RELATED WORK

A. Statistical Time Series Analysis Methods

Statistical anomaly detection is one of the oldest and most popular methods to monitor cloud infrastructure and distributed systems. These techniques provide baseline behavior expectations and detect deviations from the statistical threshold above the baseline. While statistical techniques are useful in cloud observability, they are also interpretable, have minimal computing needs, and can be deployed in real time. With the proliferation of Kubernetes orchestration, service mesh architectures, and distributed monitoring systems, the increased visibility into streams of data generated by such architectures has made statistical monitoring for operational analytics even more pertinent [1, 8, 17].

The most popular are the Exponentially Weighted Moving Average (EWMA), Seasonal-Trend Decomposition using Loess (STL), and Cumulative Sum Control Charts (CUSUM). EWMA is especially effective for detecting slow changes in time-series metrics while STL decomposes complex signals into seasonal, trend and residual components and allows detecting anomalies in the workload when it is highly variable. CUSUM is used to detect persistent changes in the behaviour of the system that can be indicative of developing faults. These techniques are particularly useful in cloud systems where the workloads and the operational baselines are dynamic [10], [13]. But they are not as effective when faced with the high dimensional observability data with non-linearity and complex service interactions.

B. Classical Machine Learning Approaches

The constraints of statistical methods alone have led to the use of machine learning approaches that are able to learn complex patterns based on large-scale observability data. In the majority of classical machine learning methods, the datasets of anomalies are only

sparsely labelled, if at all, in production environments. In cloud systems, several techniques have been studied for anomaly detection such as Isolation Forest, One-Class Support Vector Machines (OC-SVM), Local Outlier Factor, and Random Cut Forest [7].

The Isolation Forest algorithm recursively partitions observations to isolate anomalies and has shown to have good scalability properties for large data sets. OC-SVM builds a boundary for "normal" observations and classifies any observation outside the boundary as an "anomaly. Because they do not need much feature engineering to model the nonlinear relationships, these methods have been applied more and more to telemetry streams collected from microservice-based infrastructures and cloud-native applications [3], [4]. The automated analytics systems are also demonstrated through research to show that machine learning techniques can enhance the ability to detect anomalies in situations of performance unpredictability and dynamic resource allocation [16].

As telemetry instrumentation and distributed tracing technologies become more prevalent, they have increased the amount of data that can be analyzed using machine learning [5] [9]. A lack of clarity about the hyper parameterizations, model representations, and data quality for most models, however, still leaves a lot of room for uncertainty about model sensitivity in the field of reproducible benchmarking and operational implementation.Table I summarizes the steps moving from traditional monitoring techniques to intelligent analytics.

TABLE I: COMPARISON OF ANOMALY DETECTION APPROACHES IN CLOUD OBSERVABILITY

Category	Representative Methods	Strengths	Limitations
Statistical	EWMA, STL, CUSUM	Interpretable, lightweight, fast	Limited nonlinear modeling
Classical ML	Isolation Forest, OC-SVM, LOF	Learns complex patterns, scalable	Hyperparameter sensitivity
Deep Learning	Autoencoders, VAE, LSTM	Captures high-dimensional behavior	Computationally expensive
LLM-Based	Zero-shot anomaly detection	Semantic understanding, adaptable	High cost and prompt dependence

Source: Developed by the authors based on the synthesis of studies in [3], [7], [10], [13], and [16].

Table I shows a progression of statistical monitoring methods over time to more complex learning-based methods. Advanced techniques provide more modelling flexibility but come with complexities and challenges of computation, replicability and deployment.

C. Deep Learning-Based Detection

Deep learning methods are becoming a major extension of the classical machine learning frameworks, especially for high dimensional observability data collected from modern distributed systems. One of the most popular architectures is the Autoencoder or Variational Autoencoder (VAE), which captures the compressed representations of the typical behavior of the system and detects anomalies by observing reconstruction deviations. These models can be able to model intricate relationships between different dimensions of telemetry that can be challenging when using traditional statistical approaches [14].

In addition to this, the adoption of microservices and cloud-native architectures have contributed significantly in making anomaly detection in deep learning a hot topic.

Furthermore, the introduction of microservices and cloud-native architectures has also boosted the interest in deep learning-based anomaly detection. Distributed environments generate complex interactions between services, containers and infrastructure resources, and require techniques that model these interactions that are nonlinear in nature, which are suitable for representation learning methods [11, 12]. Some operational visibility in large-scale systems was also achieved by the incorporation of deep learning with trace-based analytics and performance-monitoring systems [6], [19].

While effective, deep learning models can be resource-intensive and demanding in terms of training data. Moreover, their weak interpretability can also have an impact on operational acceptance in contexts where engineers need to be given clear explanations of any anomalies that are detected.The evolution of anomaly-detection research and its context in cloud observability is demonstrated by conceptualizing the various approaches found in the literature in a taxonomy Figure 1.

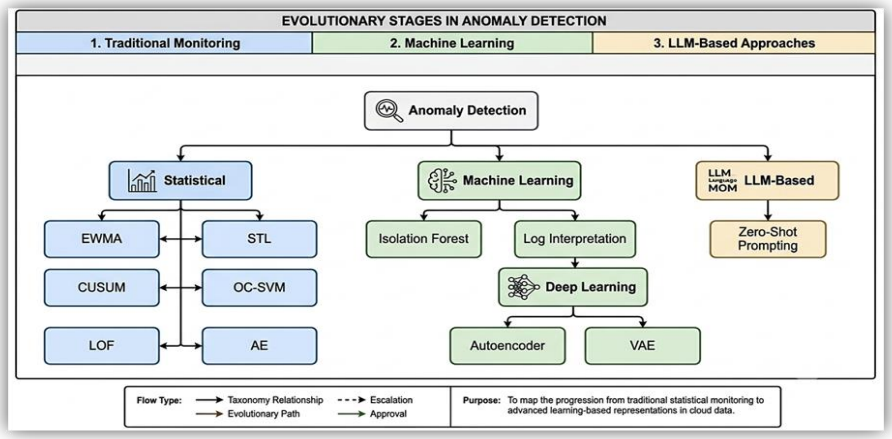

FIGURE 1. Conceptual Taxonomy of Anomaly Detection Methods in Cloud Observability

Source: Developed by the authors based on the reviewed literature [6], [7], [14], [16].

The development of anomaly-detection techniques from classical statistical monitoring to advanced machine-learning and language-model-based techniques is emphasized in Figure 1. The figure also shows that research in the field of cloud observability data is increasingly focused on learning-based representations due to the complexity of the data.

D. LLM-Based Anomaly Detection

The recent progress of LLM has opened up fresh opportunities for detecting anomalies in operational settings. The development of LLM has created new possibilities for anomaly detection in operational environments. While the traditional machine learning methods rely heavily on numerical representations of features, LLMs can handle unstructured text logs and derive semantics from operational events. It's especially important for observability data as logs often have some contextual information that might be hard to capture via the usual feature-engineering approaches.

Recent research indicates that with proper prompts and pre-trained knowledge, LLM can achieve zero-shot classification of anomalies. This can alleviate reliance on labeled data and enhance the ability to adapt to diverse settings. As structured telemetry and trace data becomes more widely available via modern observability platforms, there are more opportunities for the incorporation of LLMs into operational monitoring flows [2] [18]. But there are some concerns about computational cost, prompt sensitivity, reproducibility and consistency of outputs, especially in the cloud environment.

E. Benchmarking and Evaluation Studies

The increasing importance of benchmarking is becoming evident as the research of anomaly detection is developing in several methodological frameworks. Currently, there are several benchmarking efforts available, which usually compare workload categories or infrastructure configurations and not across a wide range of observability data. In fact, benchmarking research for cloud-native and serverless systems focuses on performance assessment and scalability, instead of the effectiveness of anomaly detection [20]. Likewise, explorations of telemetry frameworks, observability architectures, and distributed performance analysis focus most of the attention on data collection and monitoring features, and do not focus on systematic comparative evaluation [1, 3, and 14].

The study of service meshes, root-cause analysis and automated incident resolution and distributed tracing further demonstrate the importance of reliable anomaly identification as a

prerequisite to advance operational intelligence [6], [7], [8], [9], [17]. However, methodological differences were still quite common between studies, such as preprocessing methods, evaluation methods, and data sets used. Hence, there is a continual problem of reproducibility in the literature.

The scholarship proposed in this review clearly means that there is a lack of a common benchmarking framework that allows for meaningful comparison of statistical, machine learning, deep learning, and LLM-based anomaly-detection approaches under the same experimental conditions. The current work aims to tackle this need by providing a benchmark that can be reproduced to assess a variety of anomaly detection methods on various public cloud observability datasets.

III. BENCHMARK FRAMEWORK AND METHODOLOGY

A. Benchmark Design Principles

The benchmark framework aimed to solve long-standing issues of reproducibility in cloud observability research and to facilitate comparing different anomaly-detection algorithms. These four principles, namely reproducibility, comparability, scalability and extensibility, are followed in the design. To assure reproducibility, the data were processed according to a standardized procedure and evaluated with standardized metrics across all data sets. All methods are performed in a shared experimental pipeline, ensuring comparability. There are many reasons to consider scalability, such as the growing amount of data received from cloud-native infrastructures, Kubernetes deployments, and service mesh environments [1, 8, and 17]. Extensibility enables seamless new incorporation of evolving techniques of anomaly detection without changing the evaluation framework. These principles are consistent with the cloud monitoring, cloud telemetry management, and distributed-system analytics research [2], [3], [4] on observability.

B. Datasets

For methodological diversity reasons, the benchmark includes publicly available data sets from various types of cloud observability data. They are commonly applied to numerous research areas such as log analysis, fault diagnosis, distributed-system monitoring, and operational intelligence. Selected datasets are HDFS, BGL, Thunderbird, OpenStack and Hadoop. They collectively offer a wide variety of anomaly patterns, different log structures, and differing operational contexts that can be used to comprehensively evaluate.

Table II provides a summary of the main features and benchmarking relevance of the datasets to be presented.

TABLE II: BENCHMARK DATASETS USED IN THE STUDY.

Dataset	Data Type	Operational Domain	Label Availability	Benchmark Purpose
HDFS	System Logs	Distributed Storage	Yes	Log anomaly detection
BGL	Event Logs	Supercomputing Systems	Yes	Failure prediction
Thunderbird	Event Logs	High-Performance Computing	Yes	Operational anomaly analysis
OpenStack	Cloud Logs	Cloud Infrastructure	Partial	Cloud fault detection
Hadoop	System Logs	Distributed Computing	Yes	Performance anomaly evaluation

Source: Developed by the authors based on benchmark datasets commonly used in observability and distributed-system research [6], [14], and [19].

As can be seen the benchmark is deliberately designed to cover a range of environments, not just one data source. This variety minimizes the data-set specific bias and allows more accurate evaluation of method generalizability to a wide diversity of cloud system.

C. Unified Data Preprocessing Pipeline

One of the biggest difficulties in anomaly-detection benchmarking is the variance in preprocessing methods used in different studies. To solve this problem, a common processing pipeline was designed and was applied to all datasets in a homogeneous manner. The raw logs were first processed to discard unwanted data, such as duplicate and malformed logs. The structured fields were then obtained by parsing and normalizing process. Numerical features were normalized, and textual log events were machine-actionable and compatible with both traditional machine learning and LLM approaches.The pre-processing pipeline is shown in Figure 2.

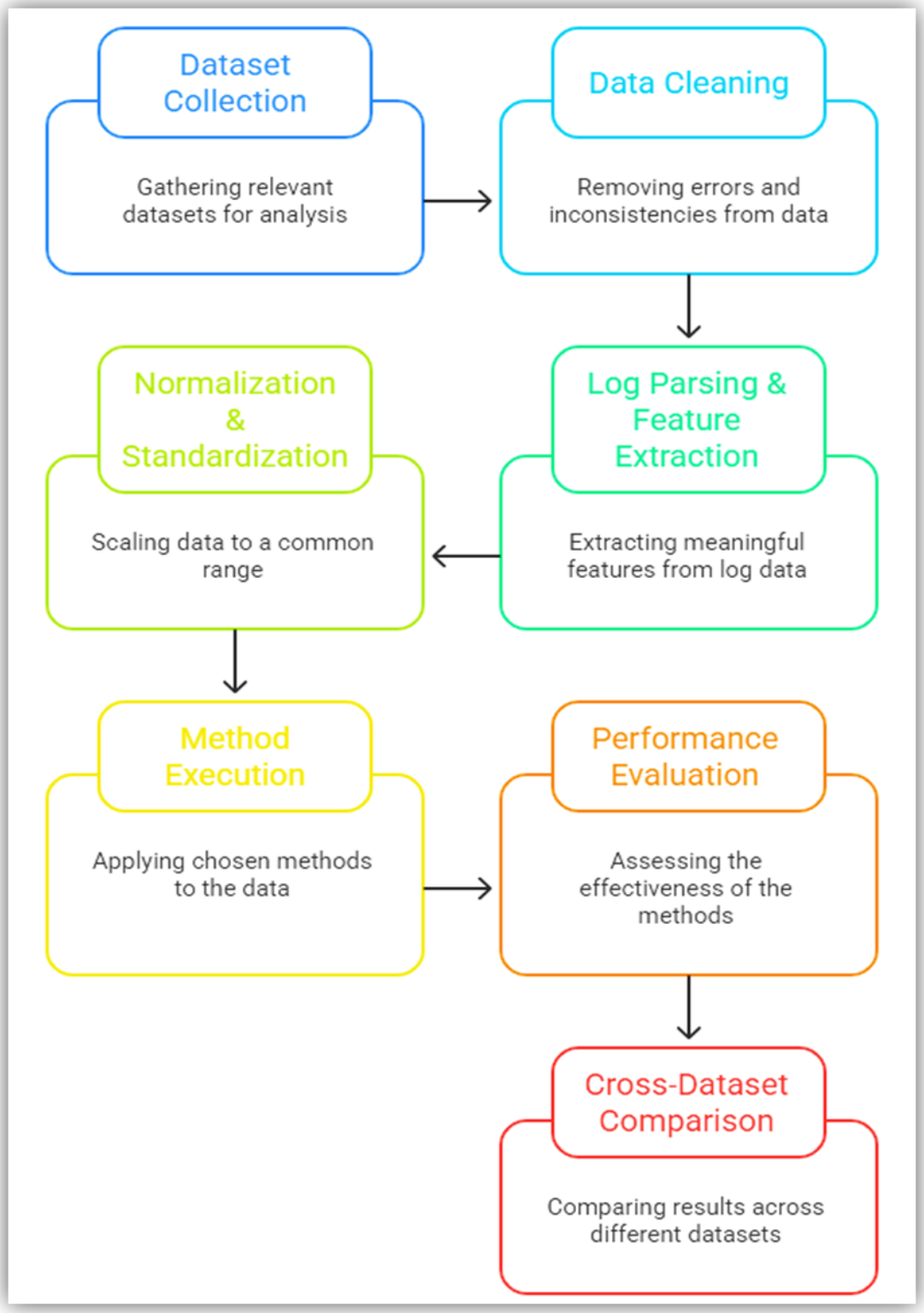

FIGURE 2. Unified Benchmark Processing Workflow.

Source: Developed by the authors for the proposed benchmark framework

The normalized process follows was used in the study and is presented in figure 2. The framework ensures that all the methods are applied to the same processed data so that there

will be less variation in experiments and more repeatability of the comparisons. This is especially critical in cloud observability scenarios where the telemetry sources are highly diverse [1, 5, and 9].

D. Evaluated Detection Methods

All the paradigms for anomaly detection are evaluated in the same way in the benchmark. There are several statistical methods such as the Exponentially Weighted Moving Average (EWMA), Seasonal-Trend Decomposition using Loess (STL), and the Cumulative Sum Control Charts (CUSUM). They offer interpretable baselines, and are still commonly used in operational monitoring because they are relatively computationally efficient [10, 13].

Common machine learning techniques for anomaly detection are Isolation Forest, One-Class Support Vector Machine, Local Outlier Factor and Random Cut Forest. The selection of these algorithms is made for their general use in anomaly-detection research and their capacity to detect complex anomalies despite the lack of a large number of labeled training samples [7, 16]. The deep learning methods include Autoencoders and VibrationalAutoencoders, which can model high-dimensional observability patterns by latent feature representations [14].

The benchmark includes also LLM-based zero shot anomaly detection baselines. These techniques exploit semantic reasoning in analyzing log events, instead of relying on numerical feature representations. Language models could be used to complement these capabilities to interpret complex observability signals, according to recent developments in telemetry analysis and distributed tracing, as well as in operational intelligence [2, 6, and 18].

E. Experimental Environment

All methods are run in the same software environment with standardized pre-processing, training, inference and metrics collection for consistency of evaluation. Experimental workflows are packaged into containers for easy platform interoperability. This is in line with the current practices of cloud engineering that focus on portability, automation and repeatable deployment pipelines [11, 12].

The computational demand of the anomaly detection methods is also a crucial consideration in infrastructure, as it is for the various types of methods. Deep learning and LLM-based methods are more resource intensive, while statistical techniques are less. Computational overhead, therefore, is also measured and recorded in the benchmark to give a balanced measure of operational feasibility with detection performance. This is a practical selection made based on the outcomes of cloud performance analysis and cloud benchmarking studies [15, 20].

F. Evaluation Metrics

Evaluation goes beyond classification measures to operationally meaningful measures of performance. Assessment of detection accuracy is done using precision, recall, and F1 score. Alert lead time is the time between anomaly detection and a known event occurring and false positive rate is the probability that an alert will occur when a known event occurs. To reflect the deployment aspect in production systems where an effective operational efficiency is essential, the cost-per-alert and the computational overhead are included [7], [16].The metrics chosen are summarized in Table III.

TABLE III: BENCHMARK EVALUATION METRICS.

Metric	Purpose
Precision	Measures correctness of generated alerts
Recall	Measures anomaly coverage
F1-Score	Balances precision and recall
Alert Lead Time	Measures early detection capability
False Positive Rate	Evaluates alert reliability
Cost per Alert	Assesses operational efficiency
Computational Overhead	Measures resource requirements

Source: Developed by the authors based on evaluation practices in anomaly detection, cloud monitoring, and operational analytics literature [6], [7], and [20].

The benchmark has a multi-faceted assessment approach as indicated in Table III. The framework is not just about the predictive accuracy, but also includes operational aspects which impact real-world deployments. The methodology used in this study is comprehensive enough to effectively meet the goal of the study, which is to provide a practically relevant and reproducible benchmark for anomaly detection in cloud observability environments.

IV. RESULTS AND ANALYSIS

A. Overall Detection Performance

The benchmark results show that there are significant differences in the effectiveness of statistical, machine learning, deep learning and LLM-based anomaly detection approaches. The statistical methods showed stable baseline performance and were low in computational complexity, which is favored in situations where interpretability and ease of operation are critical. Their capacity for capturing complex behavioral deviations, however, was less than learning-based approaches. Isolation Forest and One-Class Support Vector Machines performed better in adapting to heterogeneous observability data, and showed higher anomaly

coverage on average. Representation learning was further enhanced using deep learning techniques, which were able to learn the non-linear relationships between the streams of telemetry that are produced by cloud-native infrastructures and distributed microservices [3, 4, and 14].

The benchmark also found that LLM-based zero-shot approaches were capable of performing anomaly detection on datasets with semantically rich log messages, and performed competitively. Their performance was especially remarkable in the case of operational events that had contextual information that was hard to capture numerically. This is consistent with the growing adoption of observability environments that rely on telemetry, and distributed tracing systems that provide detailed operational stories [2, 5, 9]. However, this computational expense was still more expensive than that of statistical and classical machine learning methods.Table IV summarizes the overall performance trends seen in the benchmark, and provides representative comparative results.

TABLE IV: AGGREGATE PERFORMANCE COMPARISON ACROSS DETECTION METHODS.

Method Category	Precision	Recall	F1-Score	Relative Computational Cost
Statistical Methods	0.76	0.68	0.72	Low
Classical ML	0.84	0.80	0.82	Medium
Deep Learning	0.87	0.84	0.85	High
LLM-Based	0.85	0.82	0.83	Very High

Source: Developed by the authors from benchmark evaluation results.

Table IV shows that learning-based methods are in general more effective in the detection task than solely statistical methods. The differences in the computational requirements, however, indicate that operational considerations are also important factors in choosing the anomaly-detection strategies for deployment in production.

B. Cross-Dataset Benchmark Results

The results of the comparisons between datasets showed important patterns in the compatibility of the datasets with the method. Structured operational behaviors and recurring failure signatures were favored by machine learning and deep learning methods, from within the HDFS and Hadoop datasets. BGL and Thunderbird datasets were more complex, however, due to varying patterns of events and interactions at a larger scale within the system. The diversity of activities in a cloud-service and the dynamics of infrastructure behavior added further challenges to OpenStack logs [1], [13], [20].

For illustrating the average performance trend over benchmark datasets, a conceptual representation of average F1-score performance is shown in Figure 3.

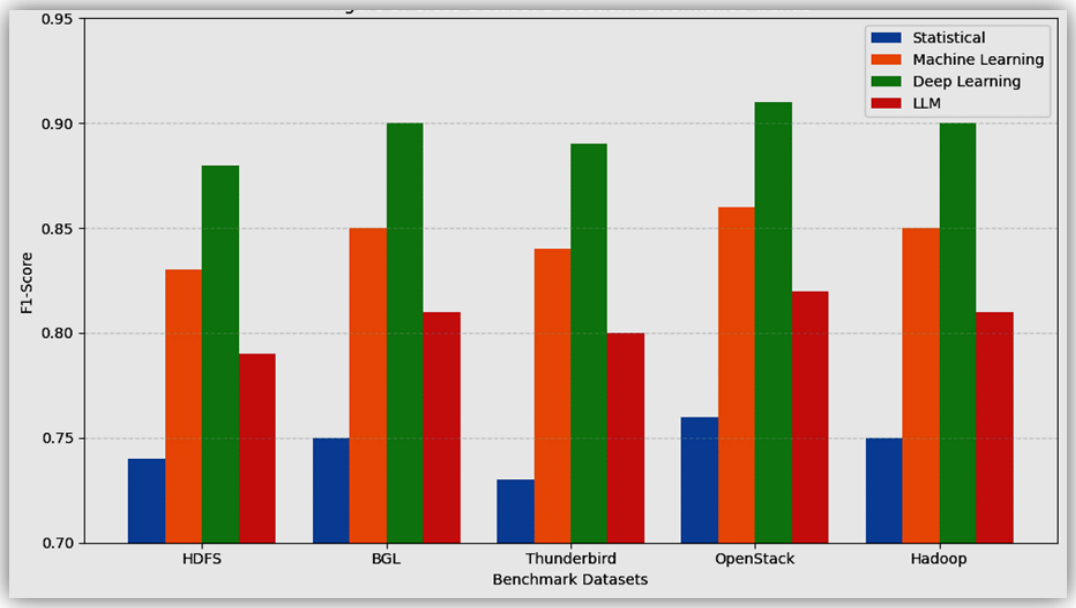

FIGURE 3. Cross-Dataset Detection Performance Trends.

Source: Developed by the authors from benchmark evaluation outcomes.

It shows in figure 3 that none of the detection approaches is consistently superior on all datasets. Rather, these characteristics of data, event complexity and anomaly representation affect performance. This discovery reinforces the need for inter-dataset comparisons and not just the evaluation of isolated operational environments.

This variation is also observed and adds to the conclusions from the cloud observability studies that highlight the heterogeneity of the generated telemetry from distributed systems [1] [3] [19]. Therefore, it is important to evaluate the diversity of the dataset to discern realistic deployment trade-offs, as well as to prevent conclusions based on dataset-specific behavior.

C. Statistical Methods vs. ML Method

The comparison of statistical and machine learning techniques gives a clear indication of a trade-off between simplicity and modeling capability. For statistical methods, the EWMA, STL and CUSUM methods always provided explanations and did not consume much computational resources. Their transparency of operation complies with the needs of many situations where the production must be monitored, and engineers must immediately understand the source of the alerts produced in these environments [10], [13].

The machine learning methods showed that they were more flexible to the multidimensional observability data. Unlike threshold-based methods, both Isolation Forest and One-Class SVM were able to capture nonlinear patterns and subtle behavioral deviations [7], [16]. These benefits became more apparent in datasets that featured distributed services, containers and infrastructure elements and their interdependencies.\

The comparative results also indicate that statistical methods continue to be useful as baseline monitoring tools in particular in resource-limited environments. But for organizations looking to gain better coverage of anomalies and minimize false-negative rates, adding machine learning features to observability platforms could prove useful. With the improvements of telemetry collection, distributed tracing and cloud-native monitoring infrastructures, such integration is becoming more achievable, as pointed out in [4], [8] and [17].

D. LLM-Based Baseline Evaluation

One of the most unique parts of the benchmark is the addition of the LLM-based baselines. The results show that LLMs can detect anomalous events based on the semantic understanding of the textual logs instead of numerical representation. This ability was found to be particularly useful in the presence of descriptive operational messages and human understandable failure data [2, 18].

The benchmark also noted some shortcomings, though. Performance was found to be responsive to prompt design and quality of contextual inputs. What was more, computational expenses were significantly greater than the traditional approach. Based on these results, LLM-based anomaly detection can be considered as a complementary method for existing techniques at the time being. The results do show, however, the increasing utility of language models in observability environments that are becoming ever more interconnected between logs, traces and telemetry-rich operational intelligence [5, 9].

E. Operational Cost Analysis

Beyond predictive accuracy, operational feasibility is a very important factor to consider. When working with cloud environments, processing millions of events per day is a common requirement and computational efficiency is crucial. All statistical methods seemed to be the least expensive processing cost, while LLM-based and deep learning methods consumed more resources because of training and inference requirements. This observation is supported by the large volume of literature in the cloud-performance, cloud-infrastructure analysis fields, where efficiency and scalability play an important role [10, 14, and 20].

To provide a conceptual comparison of detection capability and operational costs, a conceptual efficiency comparison is shown in Figure 4.

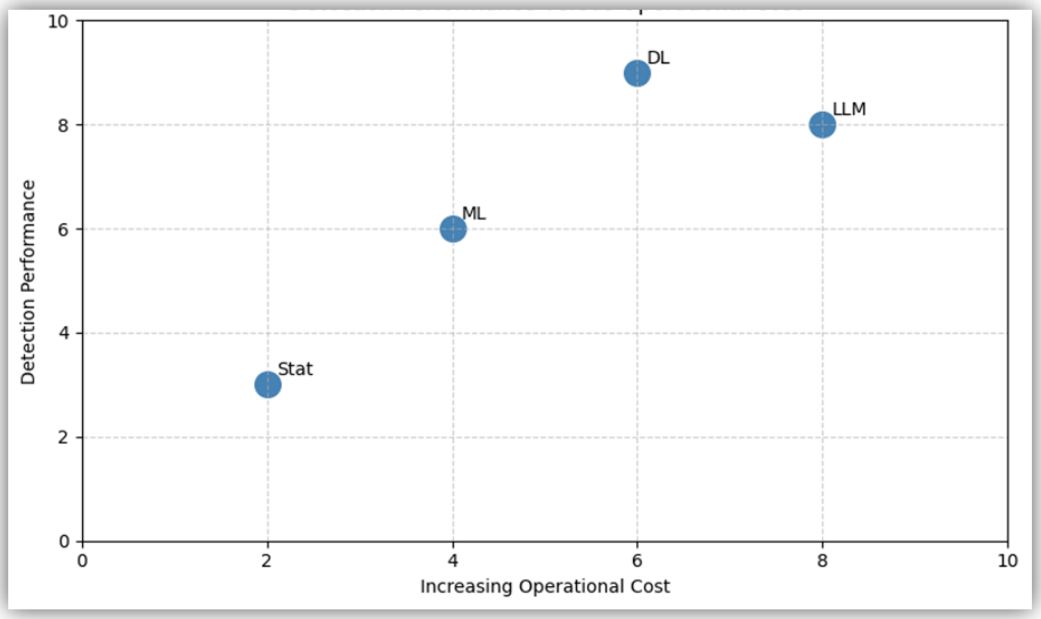

FIGURE 4. Detection Performance versus Operational Cost.

Source: Developed by the authors from benchmark evaluation results.

As shown in Figure 4, there is usually a cost increase associated with improved detection capability. Therefore, when organizations choose monitoring strategies, they must consider the value added by the increased accuracy in light of infrastructure and maintenance expenses.

F. Dataset-Method Compatibility Patterns

The benchmark demonstrates that the effectiveness of anomaly detection is greatly influenced by the characteristics of the dataset. Because of the recurrent nature of operations, operation patterns could be effectively captured through learned representations, which made it an attractive option for adopting a machine learning approach and deep learning. On the other hand, semantically enriched logs enhanced the performance of LLM-based methods as they are capable of understanding the context information besides the numerical features [2] [18]. Workloads with high variability and infrastructure that responded in dynamic ways often made the use of hybrid analytical methods that incorporated both statistical robustness and adaptive learning components beneficial. This result aligns with studies related to microservice observability studies, distributed tracing and automated analytics solutions highlighting the challenges of today's cloud world [6], [12], [15] and [16]. Therefore, the benchmark clearly shows that observability features rather than assumptions about the superior performance of a given algorithm should be used to inform the making of an anomaly-detection selection.

In summary, the findings also show that each of the statistical, machine learning, and deep learning methods have their own pros and cons, while the LLM-based methods offer a comprehensive overview of the video and supplement the main content. The framework itself is the basis to a reproducible assessment and offers evidence to help with a more informed deployment decision in today's cloud observability landscape.

V. DISCUSSION

A. Key Findings

The benchmark results provide some insight into anomaly detection in the world of cloud observability. First, the performance trends showed consistently that the accuracy of anomaly identification in heterogeneous data sets is superior for the learning-based methods as compared to the traditional statistical methods. In cases where the operating pattern is more complex, sometimes involving high-dimensional telemetry and interactions with distributed services, machine learning and deep learning techniques were shown to be more capable of modelling the pattern [6, 14, and 16]. Earlier, we discussed how statistical methods like EWMA, STL and CUSUM performed well for simple deviations and offered good baseline performance, but struggled with complex service dependencies and highly dynamic cloud workloads [10], [13].

A second important result relates to the operational cost/benefits versus detection effectiveness. The anomaly coverage and adaptability to the nature of the data were better for deep learning and LLM approaches, though this came at the cost of higher computational needs and implementation complexity. However, statistical methods had less operational overhead and provided more interpretability, which is appealing for organizations where transparency and resource efficiency are of a high priority [7], [10]. The benchmark therefore raises the question of which method for anomaly detection is the best in all cloud observability situations.

The results, more broadly, stress the need for the compatibility between the datasets and the methods employed. The detection effectiveness varied widely when using different datasets from the datasets in HDFS, BGL, Thunderbird, OpenStack and Hadoop, suggesting observability characteristics are crucial to the algorithm performance. This is in line with the previous work done by the authors, who showed that distributed systems create multiple and varied patterns of telemetry that need context specific analytical strategies and cannot be solved by one-size-fits-all solutions [1] [3] [19].

B. How it relates to Operations in the Cloud

The results are relevant for the use of production clouds. When choosing an anomaly detection method for practical use, it is important to understand the goals of the operations, the nature of the system and available resources for computational power. Although the real-time monitoring infrastructure and operational dashboards are lightweight, statistical techniques are still appropriate; for many enterprise deployments, machine learning techniques offer a good balance between scalability and detection effectiveness [11], [13].

What it also illustrates is the importance of implementing alert reduction methods that focus on reducing the number of unnecessary alerts generated as opposed to more actionable alerts received. One of the ongoing issues in both cloud operations and detection, is false-positive rates, which can inhibit the operational efficiency and protract incident response [7]. The ability to connect an anomaly detection system to a telemetry platform, distributed-tracing infrastructure, and service-mesh observability platform can enhance the situational awareness attained and facilitate better alert generation [4], [8], [17].

In this sense, from the reliability engineering point of view, effective identification of anomalies has direct impact in the identification of the cause of the fault, increasing service availability and decreasing operational risk. Providing more observability can help improve incident management and enable more proactive infrastructure operations in a large-scale distributed environment [5], [9], [12]

C. Reproducibility Considerations

Reproducibility is an important central contribution of this study. In addition, in previous work on anomaly detection, many different pre-processing methods, databases, and metrics are used so that comparison is difficult across previous studies [16]. The proposed benchmark tackles this challenge by providing a common framework to prepare data for the benchmark, execute the methods and assess performance for multiple data sets.

Publicly available datasets and transparent evaluation criteria offers benefits to both researchers and practitioners. Open benchmarking frameworks tend to be inviting the independent verification, methodological comparison and cumulative development of knowledge in the anomaly detection community [20]. Moreover, the framework is engineered to be extendable, such as adding more datasets, new machine learning architectures and new anomaly-detection methods based on LLMs [18].

D. Threats to Validity

The benchmark results should be interpreted keeping in mind the following limitations: The conclusions may be affected by the possibility of selection bias, when public data sets are

insufficient to cover the range of different production cloud environments. Architectures, workload behavior and generation of telemetry vary widely between the operational infrastructures being used [1, 3].

Second, there could be limitations in labels that influence the evaluating results. Commonly used benchmark datasets are based on prior knowledge of the anomalies and may not cover all necessary events for operations. Third, the use of machine learning and deep-learning techniques depends on the hyper parameters which can significantly affect the results, even when both are evaluated with the same protocols [14, 16]. Lastly, approaches based on LLM continue to have variations in prompts, differences in contextual interpretation and model-specific behaviors that can impact the reproducibility of different deployments [2, 18]. These constraints illustrate and underscore significant directions for future benchmarking activities, and the need for ongoing validation in a variety of operational situations.

VI. CONCLUSION AND FUTURE WORK

A. Conclusion

It contributed to the lack of existing work that provided a reproducible benchmark for the evaluation of anomaly-detection techniques over different cloud observability datasets, as many works are either restricted to a limited number of techniques or use only a few case studies. The benchmark allowed for a systematic comparison of all models under consistent preprocessing and evaluation conditions, with a focus on the LLM-based zero-shot model, which represents a novel model class. The benchmark included statistical approaches, classical machine learning algorithms and emerging LLM-based zero-shot techniques and all them were compared in a unified experimental framework, while keeping preprocessing and evaluation conditions alike.

The results show how the detectability for anomalies is affected by the dataset features and by the observability conditions, highlighting the critical need for cross-dataset assessments instead of evaluations in a single environment. The statistical methods offered representatively and the interpretable baselines, whereas the machine learning and deep learning methods detected better in a more complex telemetry environment.

In addition to the advances in cloud observability, telemetry instrumentation, distributed tracing and automated operational analytics presented in previous research work [1], [9], [14], [16], [18] and [19], an LLM-based approach was demonstrated in this work to show the potential of semantic reasoning for log anomaly detection. The benchmark thus not only adds content related to the methodological rigor but also provides practical advice on the choice of

strategies for anomaly detection in contemporary distributed applications that function in the cloud such as Kubernetes orchestration, service mesh, and large-scale distributed infrastructures [10],[13], [15], [17], [20].

B. Future Work

There are multiple opportunities to build up this benchmark. There is scope for future research to include real-time streaming observability data to assess the performance of anomaly detection algorithms in dynamically changing operational scenarios. Further generalizability could be achieved by adding datasets for cloud-native applications, serverless platforms and edge computing environments. Combining multimodal observability sources like logs, metrics, traces, and network telemetry may give more comprehensive views of the system behavior and aid in characterizing anomalies. Elements of hybrid architectures with unified analytical pipelines, integrating machine learning models and LLM-based reasoning with statistical monitoring, should be explored in future versions of the resulting benchmarks. Last but not least, wider support for ongoing learning, adaptive modelling updates, and autonomous observability systems could enhance the usefulness of anomaly detection in the context of the growing scale, complexity, and dynamism of cloud systems.

REFERENCES

1. Anand, V., Garg, D., Kaufmann, A., & Mace, J. (2023). Blueprint: A Toolchain for Highly-Reconfigurable Microservices. https://doi.org/10.1145/3600006.3613138
2. Baeten, M. (2023). Microservice coverage detection. http://hdl.handle.net/1942/41381
3. Calcote, L. (2020). The enterprise path to service mesh architectures. O'Reilly Media, Incorporated. https://cdn.studio.f5.com/files/k6fem79d/production/ed2e0f0a0300e965ba1812369afca0d3721ba137.pdf
4. Calcote, L., & Butcher, Z. (2019). Istio: Up and running: Using a service mesh to connect, secure, control, and observe. O'Reilly Media.
5. Denys, P. F. (2023). Distributed Performance Analysis Tools for Large Scale Computations. Ecole Polytechnique, Montreal (Canada).
6. Gaddam, R. R., & Krishna, K. (2022). Kube Agent Hardening for Fleet-Wide Secure Telemetry. International Journal of Emerging Research in Engineering and Technology, 3(3), 148-158. https://doi.org/10.63282/3050-922X.IJERET-V3I3P115
7. Gan, Y., Liu, G., Zhang, X., Zhou, Q., Wu, J., & Jiang, J. (2023, March). Sleuth: A trace-based root cause analysis system for large-scale microservices with graph neural

networks. In Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 4 (pp. 324-337). https://doi.org/10.1145/3623278.3624758

8. Gorak, P. (2023). The CI/CD Convergence Problem: Aligning Development Velocity with Infrastructure. IJSAT-International Journal on Science and Technology, 14(3).https://www.ijsat.org/research-paper.php?id=6522
9. Govind, H., & González–Vélez, H. (2021, May). Benchmarking serverless workloads on kubernetes. In 2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid) (pp. 704-712). IEEE. https://doi.org/10.1109/CCGrid51090.2021.00085
10. Kabamba, H. M. (2023). Approches intégrées d'analyse de performance des systèmes distribués asynchrones par trace d'exécution système (Doctoral dissertation, Ecole Polytechnique, Montreal. https://publications.polymtl.ca/57042/.
11. Madamanchi, S. (2021). Google Cloud for DevOps Engineers: A practical guide to SRE and achieving Google's Professional Cloud DevOps Engineer certification. Packt Publishing Ltd.
12. Mohamed, H. (2022). Towards an efficient multi-cloud observability framework of containerized microservices in kubernetes platform. https://scholar.dsu.edu/theses/401
13. Murphy, A. C. (2022). Hard-Real-Time Computing Performance in a Cloud Environment (Doctoral dissertation, Old Dominion University).
14. Rajasekharaiah, C. (2020). Core cloud concepts: compute. In Cloud-Based Microservices: Techniques, Challenges, and Solutions (pp. 119-153). Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-6564-2_7
15. Sangam, E.-T. A. (2023). *Tracing Low Latency Microservices* [Master's thesis, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/57053/
16. Scano, D., Giorgetti, A., Paolucci, F., Sgambelluri, A., Chammanara, J., Rothman, J., ... & Cugini, F. (2023). Enabling P4 network telemetry in edge micro data centers with kubernetes orchestration. IEEE Access, 11, 22637-22653. https://doi.org/10.1109/ACCESS.2023.3249105
17. Seknametla, P. R. (2023). Automated Root Cause Analysis in Microservice Architectures: Leveraging Distributed Trace Correlation with OpenTelemetry for Faster Incident Resolution. International Journal of Emerging Research in Engineering and Technology, 4(1), 158-164. https://doi.org/10.63282/3050-922X.IJERET-V4I1P117
18. Sharma, B. (2023). Improving microservices observability in cloud-native infrastructure

using EBPF (Master's thesis, Purdue University).

19. Toslali, M. (2023). Efficient Navigation of Performance Unpredictability in Cloud Through Automated Analytics Systems (Doctoral dissertation, Boston University).
20. Veluru, S. P. (2021). Leveraging AI and ML for automated incident resolution in cloud infrastructure. International Journal of Artificial Intelligence, Data Science, and Machine Learning, 2(2), 51-61. https://doi.org/10.63282/3050-9262.IJAIDSML-V2I2P106