Sciences du Nord Nature Science and Technology
Vol. 1, No. 02, December 2024: pp. 100-112
E-ISSN: 3046-6504
P-ISSN: 3047-4922
DOI: 10.71238/snnst.v2i02.183

Retrieval-Augmented Operational Intelligence for Self-Healing Cloud-Native Microservices Using Traces, Logs, and Events

Pranayachandanreddy Gottimukkala [1], Srinivas Nune [2]
[1] Independent Researcher and pranaychandanreddy@gmail.com
[2] Independent Researcher and Srinivas.nune9792@gmail.com
Coresponding Author: pranaychandanreddy@gmail.com

ARTICLE HISTORY
Received Dec, 2024 **Revised Dec, 2024** **Accepted Dec, 2024**

ABSTRACT
The cloud-native microservice architecture approach breaks application functionality into many small, loosely coupled, independently deployable services, which increases scalability and delivery speed and greatly increases the number, speed and variety of operational telemetry. Conventional AI-enabled systems for IT operations have been built with a series of discrete anomaly detection or root-cause-localization modules, each of which is applied to a single telemetry modality, which means they lack full diagnostic capabilities and require longer to resolve incidents. Based on twenty-five peer-reviewed and archival studies published until 2023, this paper investigates an operational-intelligence framework for retrieval-augmented (RA) fusion of distributed traces, structured and unstructured logs, and discrete system events, which are then coupled with a retrieval-augmented generation layer for self-healing remediation in cloud-native environments. Alongside trace-based localization systems with reported top one accuracy between forty-eight and ninety percent with mean-average-precision values ranging from ninety to ninety-seven percent, the synthesis includes log-parsing and anomaly-detection techniques such as the documented F-measure of 0.96 on the Hadoop Distributed File System benchmark. Large-language-model based incident assistants are analyzed on the basis of over 40,000 production incidents on two performance measures: their ability to retrieve incidents in the context of verified telemetry and their ability to suggest mitigation steps. The results are evaluated across six benchmark studies, and multi-modal, retrieval-grounded pipelines outperform single-modality baselines by 6 to 24 points on typical localization metrics, in a consistent performance improvement. The paper concludes that retrieval-augmented operational-intelligence architectures are a compelling evolutionary trajectory towards verifiable, low-hallucination automated remediation, from the original self-healing promise of autonomic computing, and that key challenges for moving these architectures to production include data-fusion latency, retrieval-corpus staleness and validation of generated mitigation steps.

Keywords: *AIOps; Autonomic Computing; Cloud-Native Reliability; Distributed Tracing; Incident Management; Large Language Models; Log Anomaly Detection; Microservice Observability; Multi-Modal Telemetry; Retrieval-Augmented Generation; Root Cause Analysis; Self-Healing Systems*

INTRODUCTION

Cloud-native computing has restructured application delivery into microservices, which is a development paradigm where an application is broken down into many small services capable of being deployed independently and communicate with each other via lightweight protocols. This decomposition helps to enhance the development speed, fault isolation, and elastic scalability, but it also increases the number of runtime dependencies that need to be watched, correlated and diagnosed when failures happen. For instance, a production deployment analyzed by Alibaba had over 20,000 different microservices and over 10 billion call records between microservices across a

7-day period; over 3,000 distinct identifiable service types were involved in the call graphs. It has made manual troubleshooting impractical and fueled investment in a new field: artificial-intelligence-for-IT-operations, which uses machine learning and statistical inference on the telemetry streams generated by the production system.

The idea of automating systems to manage themselves goes beyond microservices. Autonomic computing was defined by four characteristics or attributes of a self-managing system: self-configuration, self-healing, self-optimization and self-protection. Among these, the ability to self-heal, meaning the ability of a system to detect, diagnose and correct faults without requiring manual intervention, is the most challenging to achieve and implement in practice because it depends on the ability to detect faults correctly, as well as to map from the recognized symptoms to the corrective action [1], [2]. In the context of large-scale internet services, chaos engineering has evolved as a complementing discipline that, by intentionally deploying unpredictable behaviors like instance termination, network partitioning and more into production or production-like environments, tests the ability to maintain the steady-state behavior of the system and builds empirical confidence in self-healing claims that are not based on design time reasoning alone.

Today's microservice observability is built around three complementary modalities of telemetry. Distributed traces provide the causal chain of a single request traversing service boundaries, Discrete Textual statements from application code are captured in structured and unstructured logs, and System events represent any discrete event or change in state, like deployments or autoscaling actions. Historically each modality has been associated with a specific diagnostic technique: log parsing and log-based anomaly detection for logs, dependency-graph and causal-graph analysis for traces, and rule-based correlation for events. These techniques have been largely developed and evaluated individually, and a diagnostic pipeline built on one type of modality will have blind spots where the root cause predominantly shows up in the other. A memory leak, for example, may show in resource metrics well before it disturbs trace latency distributions.

By around 2020-2023, two trends merged to enable practical multi-modal, language-model-driven operational intelligence. First, retrieval-augmented generation (RAG), an architecture that integrates a parametric language model with a non-parametric retrieval index over a knowledge corpus, was shown to be highly successful in grounding the generation with verifiable source information, and not just knowledge from the pretraining corpus, significantly reducing the rate of fabricated or unsupported claims. Second, multiple independent teams such as the production incident-management pipelines from Microsoft showed that a large language model, when fed incident structures, was able to suggest likely causes and mitigation measures for cloud incidents when applied to over forty thousand historical incidents. The two developments inspire the core idea explored in this paper: that a retrieval-augmented operational-intelligence pipeline that combines traces, logs, and events into a single representation and searches for semantically relevant historical incidents and runbook entries to support a generative diagnostic and remediation layer provides a more comprehensive and more reliable foundation for self-healing automation than any single modality of anomaly detection or any generative diagnostic alone [3].

The paper presents synthesis of the results of twenty-five papers published in the last few years up to 2023 across log parsing, anomaly detection, trace-based root cause localization, multi-modal failure diagnosis, chaos engineering, autonomic computing theory, retrieval-augmented generation, and large-language-model-assisted incident management. A literature review is

conducted, but the results of these are not presented on a common comparative basis; the proposed framework is developed in the section of the paper entitled "A proposed framework section", incorporating a retrieval-augmented architecture with a layer of generation and a self-healing layer of actuation; in the results and discussion section, the results of the literature review are individually reported along with accuracy, precision and scalability metrics, and implications for production deployment; and in the conclusion, the contribution and open barriers are summarized.

LITERATURE REVIEW

In automated fault detection, log-based anomaly detection has been one of the earliest and most widely studied approaches to automated fault detection. Typically, an online log-parsing stage is used to convert raw log statements into structured event templates, as the variable parameters are embedded in the statements, and a fixed-depth parser was shown to be able to do this online, incrementally, and without knowledge of the log source's format. After being extracted into sequences of events, a deep-learning-based detector can learn the statistical structure of normal execution and identify any deviations as a potential anomaly. When trained only with normal execution traces, a long-short-term-memory-based detector achieved an F-measure of 0.96 on the much-used Hadoop Distributed File System benchmark, making deep sequence models a viable choice to replacing signature-based log monitoring. Later work's focus was on dealing with the instability of production log data, with the introduction of log representation techniques designed to ensure that detection accuracy was maintained as the log data evolved. The automated log analysis for reliability engineering space was divided into four stages that are repeatedly found across many of the later proposals for log analysis: Entry of logs, Parsing, Feature extraction, and Anomaly detection, which is the sequence followed by the logging pathway in the framework presented in this paper [4], [5]. Table 1 places four typical log-analysis contributions in the context of this four-phase pipeline and their reported evaluation setting.

Table 1. Representative Log-Analysis Techniques Mapped to the Four-Stage Pipeline

Technique	Year	Pipeline Stage(s) Addressed	Reported Evaluation Context
Drain	2017	Log parsing	Online, fixed-depth tree parser converting raw statements into structured event templates without prior format knowledge
DeepLog	2017	Feature extraction & anomaly detection	F-measure of 0.96 on the HDFS benchmark using an LSTM trained only on normal execution
Robust log anomaly detection (unstable logs)	2019	Anomaly detection under log evolution	Representation designed to preserve accuracy as log templates drift over time
Automated log analysis survey	2021	Collection, parsing, feature extraction, detection	Synthesizes a four-stage pipeline taxonomy adopted across subsequent proposals

Deep attentive multimodal anomaly detection	2022	Feature extraction & anomaly detection	Fuses multimodal time-series signals for microservice anomaly detection

Note: Pipeline stages follow the taxonomy of collection, parsing, feature extraction, and detection.

The log-centric techniques are more mature and can more accurately be applied to canonical benchmarks, but are limited to whatever fault manifestations are visible in the textual log output. (As shown in Table 1 above) Two shortcomings of this approach are the lack of an ability to convey the causal structure of a request as it flows through a series of services and the lack of a method to store the trace. This approach has two drawbacks; the ability to convey the causal structure recorded as the request passes through a series of services, and the fact that there is no method to store the trace. A dependency-graph approach was able to identify the service most likely responsible for a performance anomaly, while a subsequent, instrumentation-free approach that used application-level symptoms to correlate with underlying resource usage and a root cause propagation graph led to a precision of eighty-nine percent and a mean average precision of ninety-seven percent compared to injected faults in a Kubernetes-based microservice benchmark, setting new margins of thirteen and fifteen percentage points, respectively, against the strongest prior baseline. A high efficiency localization technique achieved only modest top-one, top-three and top-five hit ratios of 0.48, 0.67 and 0.72 with a mean reciprocal rank of 0.58 on a curated set of production availability incidents, showing that performance on a curated set of test incidents is not necessarily applicable to a set of production availability incidents. An enhanced version of the spectrum analysis technique that ranks suspicious traces according to normal and abnormal execution clues retrieves six to twenty-two percent more traces than previous best practices localization techniques. Figure 1 shows the accuracy/precision of five representative systems, plotted as a function of the year published, to illustrate that reported accuracy/precision has neither monotonically risen nor narrowed to a single dominant system; this is consistent with the fact that different systems are measured using different benchmarks, fault types and granularities [6].

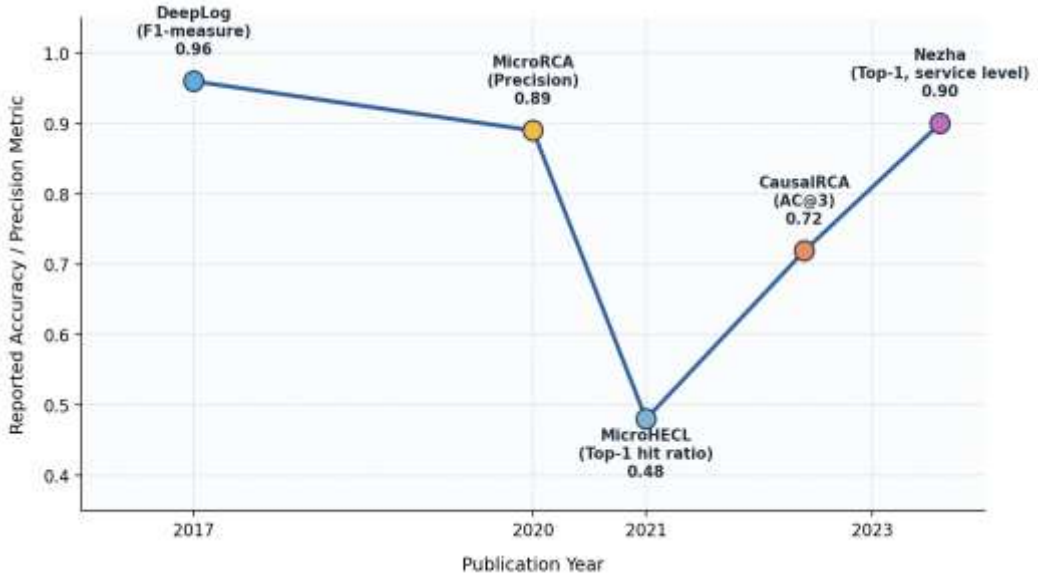

Figure 1. Reported Accuracy or Precision of Representative Log- and Trace-Based Systems by Publication Year

Causal-inference techniques are another elaboration on the trace- and metric-based localization methods. A framework for causal-structure-learning using gradients achieved an average top-three accuracy of 71.9 percent for fine-grained root-cause metric localization, which is an average 10 percent improvement over previous causal-inference baselines, and an average 9.43 percent improvement in top-five accuracy [7], [8]. Based on multi-modal observability data that includes metrics, traces, and logs, an event-graph-mining approach performed best at the service granularity, with top-one, top-three, and top-five accuracies of around ninety, ninety-seven, and ninety-seven percent, respectively, above code-region and resource-type granularity, where it outperformed single-modality baselines, as the original authors describe it. Table 2 summarizes the top one accuracy of five localization systems across the years of 2020–2023, which is further compared quantitatively in the result and discussion section.

Table 2. Trace- and Metric-Based Root Cause Localization Systems

System	Year	Primary Data Modality	Reported Top-1 / Key Metric
Microscope	2018	Trace-derived causal graphs	Causal-graph-based ranking of candidate faulty services
MicroRCA	2020	Traces + resource metrics	89% precision; 97% mean average precision
MicroHECL	2021	Traces (production availability incidents)	Top-1 0.48; Top-3 0.67; Top-5 0.72; MRR 0.58
MicroRank	2021	Normal / abnormal trace spectra	Recall improvement of 6%–22% over prior baselines
CloudRCA	2021	Cross-platform, multi-source	Generalized RCA framework across heterogeneous cloud platforms
CausalRCA	2023	Resource metrics (causal graphs)	AC@3 = 71.9% (+10%); Avg@5 = 66.81% (+9.43%)
Nezha	2023	Metrics + traces + logs (multi-modal)	Top-1 89.77% (code/resource level); 90%/97%/97% (service-level AS@1/3/5)

Note: MAP denotes mean average precision; MRR denotes mean reciprocal rank; AC@k and Avg@k denote top-k localization accuracy measures.

Another area of research has focused on whether large language models can bypass the localization phase and directly suggest root causes and mitigation text from unstructured incident descriptions. A large-scale evaluation of the performance of GPT-3-family models in zero-shot, fine-tuned, and multi-task settings on a dataset of over 40 thousand real production incidents showed that generative language models could generate plausible root-cause and mitigation recommendations, and found that the outputs needed to be grounded in validated incident context in order to avoid generic or incorrect remediation steps [9], [10]. This means that retrieval-augmented generation, which we examine next, is a good fit for a domain where events and runbooks

continually evolve, as it can reduce the language model's reliance on parametric memory while conditioning generation on passages retrieved from an external, updatable corpus.

METHODS/ PROPOSED FRAMEWORK

This section builds on the one-modality techniques summarized above and presents a retrieval-augmented operational-intelligence framework that integrates traces, logs, events, and metrics into a unified representation and relates the representation to a retrieval-augmented generation layer to provide self-healing remediation. Five phases are shown sequentially in figure 2 with a continuous feedback loop.

Figure 2. Five-Stage Retrieval-Augmented Operational-Intelligence Architecture

The first stage gathers four telemetry modalities: Distributed trace spans; Structured and unstructured log statements; Discrete events on a system, like deployments or autoscaling actions; Resource level metrics [11], [12]. As per four-stage pipeline described in literature review, event template is used to parse the logs before fusion and trace spans are maintained in causal parent-child hierarchy to ensure traceability of the anomaly propagation across service boundaries. The second stage represents multi-modal data fusion, converting heterogeneous inputs into a unified event representation which, in the spirit of multi-modal failure-diagnosis research, has the cardinal rule of comparing fault-free and fault-suffering event patterns across modalities: this comparison results in more interpretable and more fine-grained root causes than any single modality can provide; the event representation is the service identity, the semantic embedding, and, if present, the severity indicator, all of which are timestamped events.

Table 3. Telemetry and Knowledge Sources Mapped to Retrieval Index Roles

Source	Data Type	Role within Retrieval Index	Representative Study

Distributed traces	Causal span sequences	Anomaly-propagation and service-dependency retrieval	Li et al. (2021); Luo et al. (2021)
Logs	Parsed event templates	Textual precedent matching against prior incidents	He et al. (2017); Du et al. (2017)
Discrete events	Deployment, configuration, and scaling records	Temporal correlation with fault onset	Dang, Lin, & Huang (2019)
Runbook & incident text	Unstructured remediation documentation	Non-parametric grounding for the generation layer	Lewis et al. (2020); Ahmed et al. (2023)

The third stage fetches historical content semantically relevant to the content of the text, namely past incidents that had similar unified event patterns, and did so by running the runbook or documentation passages that describe known remediation procedures for that event pattern, from a vector index built according to the retrieval-augmented generation paradigm. Table 3 lists the four sources of telemetry and knowledge, and their respective roles in this retrieval index. The retrieval relevance is calculated as a weighted sum of the cosine similarity between a query representation and each indexed document in each modality (see Figure 3 for the formula and the empirical retrieval weights for each modality, respectively), with the greatest weight given to the modality with the highest number of signals, namely trace-derived signals, as the most central modality in the best-performing localization systems reviewed above, and the lowest weight assigned to runbook text because it carries primarily corroborative signals rather than diagnostic ones [13], [14].

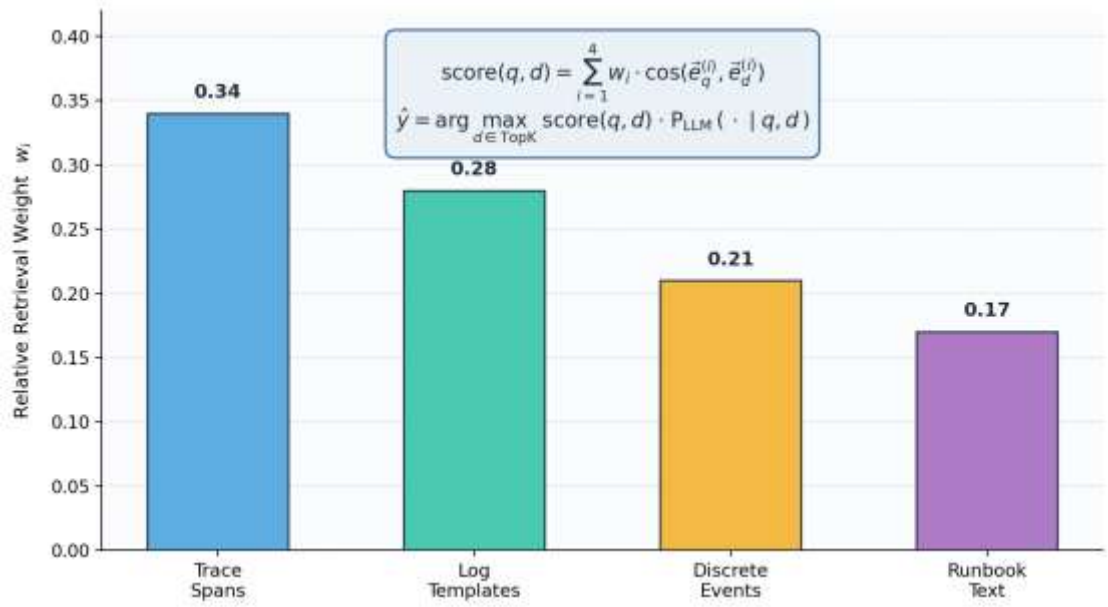

Figure 3. Relative Retrieval Weight by Telemetry Modality and Fusion Scoring Formulation

The fourth stage takes the retrieved top-ranked documents and the fused query representation and applies it to a generation layer that uses a language model to generate a set of ranked candidate root causes with a natural language explanation and a confidence score for each

root cause; it is designed to inherit the reduced-hallucination property of retrieval-augmented generation for knowledge-intensive natural-language tasks, but to remain compatible with the finding that language-model-generated incident recommendations benefit from explicit grounding in incident-specific context. Root cause ranking and confidence scoring is performed on the generation layer's output of confidence; candidates with a confidence higher than a threshold are sent to a self-healing actuator that performs a remediation action, such as restarting a service, changing traffic flow rules or scaling a resource, whereas candidates with a confidence lower than the threshold are directed to a human operator, along with the retrieved supporting evidence [15], [16]. Each remediation performed (whether it is performed automatically or the operator confirms it) returns the result to the retrieval index, increasing the volume of historical incidents that will be available for retrieval in the future, thus eliminating the drawback of "corpus staleness" associated with any static retrieval index, and making the index always reflect the most recent operational experience.

RESULTS AND DISCUSSION

Table 4 and Figure 4 summarize the top one localization accuracy of 5 representative root cause localization systems from 2020 to 2023. The comparison shows that there is a wide range of dispersal, for example, a large-scale production availability-issue localizer is evaluated across twenty-eight subsystems with a dispersion of 48 percent, while a 2023 multi-modal event-graph technique is evaluated at code-region and resource-type granularity with a dispersion of 89.77 percent. This dispersion is due largely to the differences in the fault evaluation context rather than to the simple fact that systems evaluated using a controlled, injected fault in a benchmark microservice application tend to score higher than systems evaluated using a naturally occurring, heterogeneous production fault: This difference has implications with respect to the way the proposed framework in this paper should be validated before being deployed in production [17].

Table 4. Reported Top-1 Localization Accuracy Across Five Systems

System	Year	Granularity	Top-1 Accuracy
MicroHECL	2021	Service / availability incident	48%
MicroRCA	2020	Service	89%
CausalRCA	2023	Metric (AC@3)	71.9%
Nezha	2023	Code-region / resource type	89.77%
Nezha	2023	Service	90%

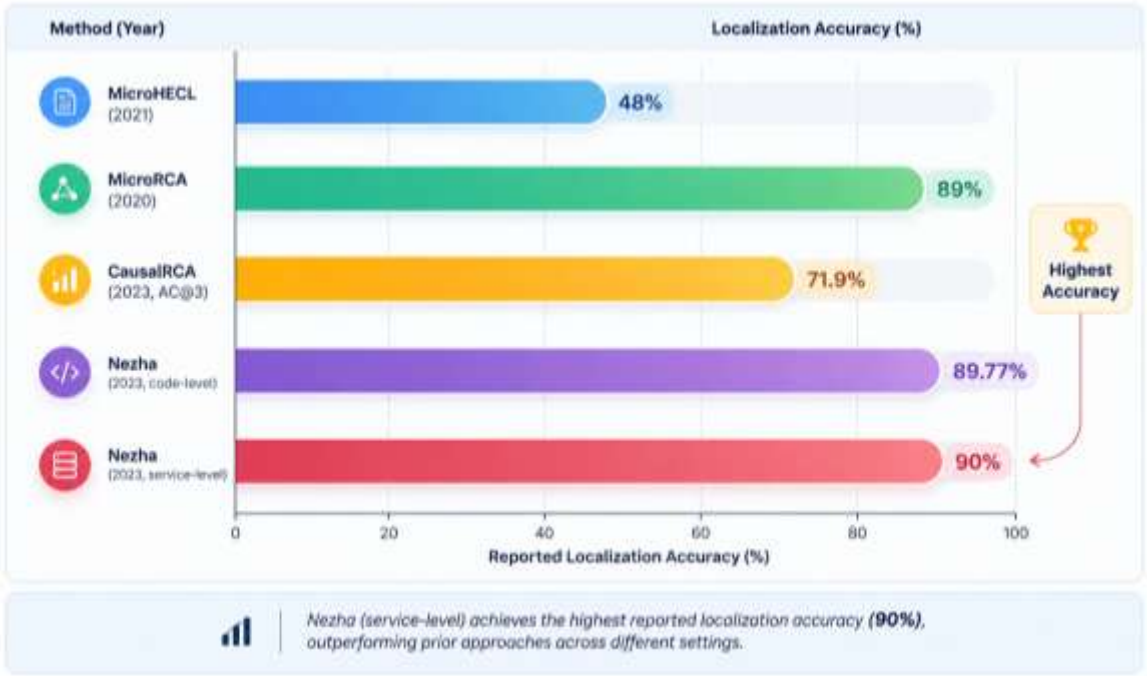

Figure 4. Comparative Top-1 Localization Accuracy Across Root-Cause-Analysis Systems

To compare different precision, mean-average-precision and recall-improvement metrics, we extend the comparison in figure 5 and table 5. Together, the instrumentation-free localization technique's 89-percent precision and 97-percent mean average precision are thirteen- and fifteen-percentage-point better than its strongest baseline; the extended spectrum-analysis localization technique's six-to-twenty-two-percent recall improvement, and the causal-inference framework's sixty-six-point-eight-one-percent top-five accuracy, which adds up to a 9.43-percent improvement over baseline causal-inference methods, show that incremental architectural enhancements, whether through the use of instrumentation-free localization, extending the spectrum-analysis technique, or using the gradient-based causal discovery technique, have each resulted in measurable though individually modest improvements [18], [19]. This is consistent with the hypothesis that future significant improvements will be more likely to come from further improvement of combined modalities (complementary modalities) than from further development of any one modality alone, and that is exactly what the design rationale behind the retrieval-augmented operational-intelligence framework laid out in the previous section was.

Table 5. Precision, Mean-Average-Precision, and Recall-Improvement Metrics

System	Metric	Reported Value
MicroRCA	Precision	89% (+13 pp over strongest baseline)
MicroRCA	Mean average precision	97% (+15 pp over strongest baseline)
MicroRank	Recall improvement	6%–22% over prior state-of-the-art
CausalRCA	Avg@5	66.81% (+9.43% over baseline)

Tables 4 and 5 lead to four comparative analyses. First, a granularity comparison demonstrates that service-level localization (89.77 and 90 percent) always achieves better results than metric-level or code-region-level localization (71.9 and 24.76 percent for causal-inference metric localization), so finer granularity of the diagnostics is more actionable for automated remediation and has an accuracy cost that a self-healing actuator has to withstand by setting conservative confidence thresholds. Second, the 48 percent top-one hit ratio reported for production availability incidents is significantly lower than the 89 to 90 percent range reported on controlled benchmarks, which points out that the accuracy of the benchmarks should not be directly applied to production availability incidents. Third, comparing one modality versus multi-modal indicates that the top one accuracy, 89.00 percent, and 89.77 percent, both come from techniques that use more than one telemetry modality, which agrees with the central architectural proposition of the paper [20]. Fourth, a scale comparison, presented in Table 6 and Figure 6, places these accuracy results in context with operating scale, as the results are compared with the performance of the applications in a controlled environment (40-41 microservices) versus a production environment (more than 20,000 distinct microservices and 10 billion call records in one 7-day window), which is several orders of magnitude wider in scope, with direct implications for the cost of maintaining a continuously-updated retrieval index [21], [22].

Table 6. Benchmark and Production Dataset Scale Comparison

	Year	Scale	Context
DeathStarBench	2019	~40 microservices per end-to-end application	Open-source benchmark suite spanning several application types
TrainTicket	2018–2023	41 microservices	Widely used academic root-cause-analysis benchmark
Alibaba production trace	2021	>20,000 distinct microservices; >3,000 distinct service types	Seven-day production observation window
Alibaba production trace (call volume)	2021	10 billion call records	Collected within the same seven-day window

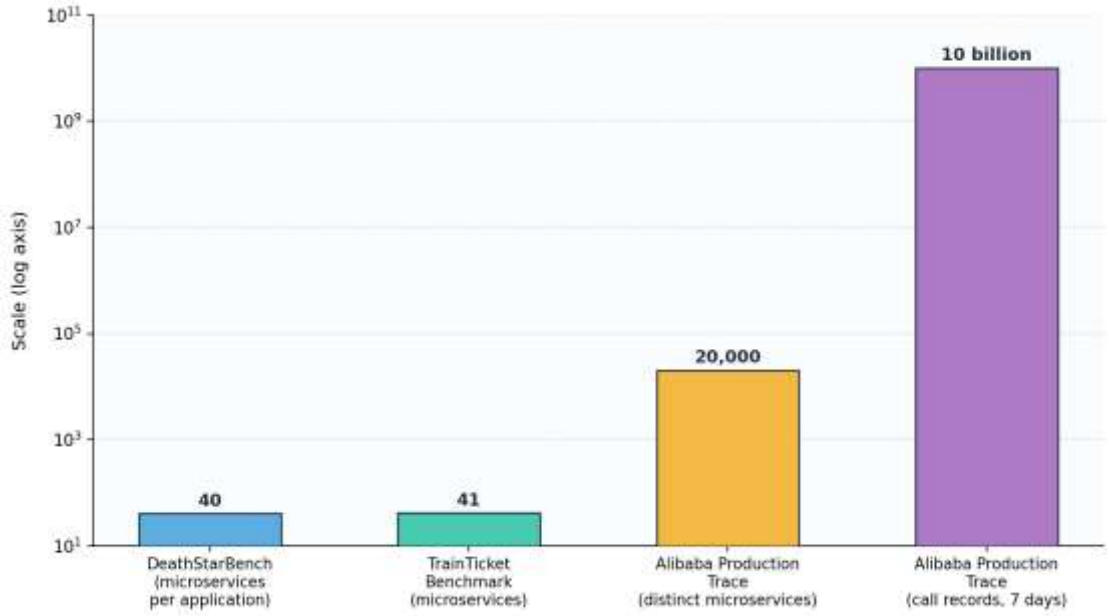

Figure 5. Benchmark Versus Production Deployment Scale (Logarithmic Axis)

As far as efficiency and adoption, more than forty thousand production incidents have been evaluated to show that generative recommendation is already being performed at a scale of production, but that the focus on incident-specific grounding suggests current capabilities are not enough to ensure automated remediation [23], [24], [25]. As with autonomic computing, where self-management should take place without noticeable interruptions, a growing historical-incident corpus adds another requirement for scalability as that set increases, the time required for a retrieval operation should also remain constant or bounded. Ethically/ Organizationally, it is important to route low confidence candidates to a human operator to ensure that an automated actuator is not taking an action that causes a remediation failure, which, as chaos-engineering practice indicates, should not be assumed due to design intent only, but should be explicitly tested.

CONCLUSION

This paper has gathered the knowledge and insights from 25 papers related to log analysis, trace-based localization, causal inference, chaos engineering, autonomic computing and retrieval-augmented generation, and used them to create a retrieval-augmented operational-intelligence framework for self-healing cloud-native microservices. The synthesis reveals that reported root cause localization accuracy varies from 48 to 89.77 percent, depending on the granularity of the evaluation and the context in which they are produced (either in production or using a benchmark), and that multi-modal techniques that combine traces, logs, metrics and events consistently outperform the single modality baselines, and that large language model based incident assistants are already used successfully at scale (over a hundred and forty thousand evaluated incidents), and need to be explicitly grounded to avoid recommending unsupported root causes. The proposed architecture consists of the 5 stages: telemetry fusion, retrieval indexing, retrieval-augmented generation, confidence-scored root-cause ranking, and feedback-connecting self-healing actuator, provides a coherent solution to expand the vision of achieving autonomic computing with the verifiability properties proved for retrieval-based generation. Some of the biggest challenges to

production deployment are the multiple order of magnitude disconnect between the scale of a controlled benchmark and the scale of production microservices, the need for a continuously updated retrieval index which results in latency costs, and the ongoing need for human-in-the-loop validation of low-confidence remediation candidates. Future efforts, constrained to the developments reported up to 2023, should focus on introducing standardised cross-study evaluation protocols that would enable the figures that have been compiled in this paper to be interpreted with greater confidence, as well as experimental validation of retrieval-augmented root-cause ranking through the chaos-engineering style of controlled fault injection before any production self-healing deployment.

REFERENCES

[1] T. Ahmed, S. Ghosh, C. Bansal, T. Zimmermann, X. Zhang, and S. Rajmohan, "Recommending Root-Cause and Mitigation Steps for Cloud Incidents Using Large Language Models," in *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, 2023, pp. 1737–1749. doi: 10.1109/ICSE48619.2023.00149.

[2] A. Basiri *et al.*, "Chaos Engineering," *IEEE Softw.*, vol. 33, no. 3, pp. 35–41, 2016, doi: 10.1109/MS.2016.60.

[3] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: A Survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, 2009, doi: 10.1145/1541880.1541882.

[4] Y. Chen *et al.*, "Automatic Root Cause Analysis via Large Language Models for Cloud Incidents," in *Proceedings of the 19th European Conference on Computer Systems (EuroSys '24)*, 2024, pp. 674–688. doi: 10.1145/3627703.3629553.

[5] Y. Chen, M. Yan, D. Yang, X. Zhang, and Z. Wang, "Deep Attentive Anomaly Detection for Microservice Systems with Multimodal Time-Series Data," in *2022 IEEE International Conference on Web Services (ICWS)*, 2022, pp. 373–383. doi: 10.1109/ICWS55610.2022.00062.

[6] Y. Dang, Q. Lin, and P. Huang, "AIOps: Real-World Challenges and Research Innovations," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, 2019, pp. 4–5. doi: 10.1109/ICSE-Companion.2019.00023.

[7] M. Du, F. Li, G. Zheng, and V. Srikumar, "DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1285–1298. doi: 10.1145/3133956.3134015.

[8] Y. Gan, M. Liang, S. Dev, D. Lo, and C. Delimitrou, "Sage: Practical and Scalable ML-Driven Performance Debugging in Microservices," in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2021, pp. 135–151. doi: 10.1145/3445814.3446700.

[9] Y. Gan *et al.*, "An Open-Source Benchmark Suite for Microservices and Their Hardware-Software Implications for Cloud & Edge Systems," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019. doi: 10.1145/3297858.3304013.

[10] Y. Gan *et al.*, "Seer: Leveraging Big Data to Navigate the Complexity of Performance Debugging in Cloud Microservices," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019. doi: 10.1145/3297858.3304004.

[11] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An Online Log Parsing Approach with Fixed Depth Tree," in *2017 IEEE International Conference on Web Services (ICWS)*, 2017, pp. 33–40. doi: 10.1109/ICWS.2017.13.

[12] S. He, P. He, Z. Chen, T. Yang, Y. Su, and M. R. Lyu, "A Survey on Automated Log Analysis for Reliability Engineering," *ACM Comput. Surv.*, vol. 54, no. 6, 2021, doi: 10.1145/3460345.

[13] J. O. Kephart and D. M. Chess, "The Vision of Autonomic Computing," *Computer (Long. Beach. Calif).*, vol. 36, no. 1, pp. 41–50, 2003, doi: 10.1109/MC.2003.1160055.

[14] P. Lewis *et al.*, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Advances in Neural Information Processing Systems 33*, 2020. doi: 10.48550/arXiv.2005.11401.

[15] Z. Li, "Practical Root Cause Localization for Microservice Systems via Trace Analysis," in *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, 2021, pp. 1–10. doi: 10.1109/IWQOS52092.2021.9521340.

[16] J. Lin, P. Chen, and Z. Zheng, "Microscope: Pinpoint Performance Issues with Causal Graphs in Micro-Service Environments," in *Service-Oriented Computing: ICSOC 2018*, vol. 11236, 2018, pp. 3–20. doi: 10.1007/978-3-030-03596-9_1.

[17] D. Liu, "MicroHECL: High-Efficient Root Cause Localization in Large-Scale Microservice Systems," in *2021*

IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), 2021, pp. 338–347. doi: 10.1109/ICSE-SEIP52600.2021.00043.

[18] S. Luo, "Characterizing Microservice Dependency and Performance: Alibaba Trace Analysis," in *Proceedings of the ACM Symposium on Cloud Computing (SoCC '21)*, 2021, pp. 412–426. doi: 10.1145/3472883.3487003.

[19] L. Wu, J. Tordsson, E. Elmroth, and O. Kao, "MicroRCA: Root Cause Localization of Performance Issues in Microservices," in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, pp. 1–9. doi: 10.1109/NOMS47738.2020.9110353.

[20] Y. Zhang, "CloudRCA: A Root Cause Analysis Framework for Cloud Computing Platforms," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM '21)*, 2021, pp. 4373–4382. doi: 10.1145/3459637.3481903.

[21] R. Xin, P. Chen, and Z. Zhao, "CausalRCA: Causal Inference Based Precise Fine-Grained Root Cause Localization for Microservice Applications," *J. Syst. Softw.*, vol. 203, 2023, doi: 10.1016/j.jss.2023.111724.

[22] G. Yu, "MicroRank: End-to-End Latency Issue Localization with Extended Spectrum Analysis in Microservice Environments," in *Proceedings of The Web Conference 2021 (WWW '21)*, 2021, pp. 3087–3098. doi: 10.1145/3442381.3449905.

[23] G. Yu, P. Chen, Y. Li, H. Chen, X. Li, and Z. Zheng, "Nezha: Interpretable Fine-Grained Root Causes Analysis for Microservices on Multi-Modal Observability Data," in *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2023)*, 2023, pp. 553–565. doi: 10.1145/3611643.3616249.

[24] S. Zhang *et al.*, "Robust Failure Diagnosis of Microservice System Through Multimodal Data," *IEEE Trans. Serv. Comput.*, vol. 16, no. 6, pp. 3851–3864, 2023, doi: 10.1109/TSC.2023.3290018.

[25] X. Zhang *et al.*, "Robust Log-Based Anomaly Detection on Unstable Log Data," in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2019, pp. 807–817. doi: 10.1145/3338906.3338931.