



Leveraging multi-agent framework for root cause analysis

Feng Fu¹ · Hong Ding¹ · Yong Qin¹ · Jian Yu¹ · Dechao Xu¹

Received: 20 June 2025 / Accepted: 5 September 2025

© The Author(s) 2025

Abstract

RCA is critical for operational stability in complex integrated systems, such as cloud-native platforms and distributed power metering infrastructures. However, achieving automated RCA faces three fundamental challenges: (1) the complex mapping between anomalies and runtime data, (2) the semantic gap between unstructured logs and structured metrics, and (3) combinatorial explosion of causal relationships for inferring the root causes. While LLMs offer potential for automated RCA due to their superior reasoning and knowledge-linking capabilities, their susceptibility to hallucinations constrains practical deployment. Specifically, prevailing single-agent architectures exacerbate error propagation and context-switching failures during multi-step RCA reasoning, leading to incorrect root cause identification. To address these limitations, we propose MA-RCA (multi-agent root cause analysis), a collaborative framework deploying specialized agents for distinct subtasks. Each agent operates within a dedicated domain to minimize context-switching failures. Crucially, to counteract hallucinations and error propagation, MA-RCA introduces two agents: a Retrieval Agent that grounds hypotheses in external domain knowledge (e.g., historical documentation) using retrieval-augmented generation, and a Validation Agent that verifies hypotheses by executing dynamic tests against runtime data. Experimental evaluations on cloud-native platforms (Nezha, 95.2% F1) and distributed power metering infrastructures (82.8% F1) demonstrate the effectiveness of MA-RCA in automating multi-domain RCA, bridging intelligent computing with complex system resilience through agent collaboration.

Keywords Root cause analysis · Complex systems diagnostics · Multi-agent framework · Large language models

Abbreviations

RCA	Root cause analysis
LLMs	Large language models
SREs	Site reliability engineers
IEC	International Electrotechnical Commission
TTL	Time-to-live
LSH	Locality-sensitive hashing
KPIs	Key performance indicators
CoT	Chain-of-thought
RAG	Retrieval-augmented generation
NLP	Natural language processing

Introduction

RCA serves as a cornerstone of operational reliability in complex distributed systems, particularly in cloud-based

platforms [21, 35] and distributed power metering infrastructures [18, 23] where transient anomalies (e.g., microservice failures, malware or voltage sags) can trigger cascading outages within minutes [3, 8, 12]. Traditional RCA predominantly relies on manual expertise: SREs leverage domain knowledge and personal experience to deduce potential causes from anomalous reports and symptoms (e.g., logs, metrics, traces). However, cultivating experienced SREs entails significant time investments. Furthermore, as system complexity increases, the exponential growth of telemetry data across multi-modal logs, service dependency graphs, and resource metrics makes human-driven analysis increasingly impractical [10, 17, 32, 34]. These limitations underscore an urgent need for automated RCA solutions, which could reason over ephemeral failure modes with minimal human intervention.

The manual RCA process typically involves three sequential steps [24]: (1) determining necessary supplementary data based on anomalies (e.g., alerts), guided by domain expertise and personal experience; (2) gathering targeted evidence from logs, traces, and operational repositories; (3) corre-

✉ Feng Fu
fufeng6688@hotmail.com

¹ NARI-TECH Nanjing Control Systems Ltd, Nanjing 210003, China

lating collected data with anomalies to infer root causes through systematic examination. To automate this workflow faces three fundamental challenges: (1) the complex mapping between anomalies and runtime data; (2) the semantic gap between unstructured logs and structured metrics; (3) combinatorial explosion of causal relationships for inferring the root causes. Early approaches [9, 16, 36] employed rule-based and statistical heuristics (e.g., decision trees, correlation analysis) to directly map symptoms to root causes, bypassing dynamic data collection and analysis. This strategy prone to failure in complex environments due to oversimplified causality assumptions. Subsequent machine learning methods [15, 30] and deep learning methods [27] introduced multi-modal runtime data (e.g., logs, traces, topology) for causal inference. Yet, these techniques exhibited limited generalizability and relied heavily on high-quality training data, which is often scarce in production scenarios [5].

The emergence of LLMs has demonstrated exceptional reasoning capabilities and knowledge-linking capacities [13, 22, 25, 29]. These models can understand and parse both unstructured and structured data, uncovering causal relationships across heterogeneous sources, which naturally align with the requirements of automating RCA. Pioneering work [1] fine-tunes LLMs to generate root causes directly from anomaly titles and descriptions, demonstrating significant potential in RCA accuracy. To achieve end-to-end automation, RCACOPilot [4] leverages predefined handlers for data collection followed by LLM-based analysis. However, manually configured handlers lack flexibility to adapt to dynamic system topologies. Innovations like REACT [24] and RCAgent [28] propose LLM-based agents to autonomously select tools for data gathering and analysis. Notwithstanding these advances, the hallucinations problem of LLMs impede practical deployment, where agents generate plausible but ungrounded causal hypotheses. To mitigate hallucinations, mABC [33] draws inspiration from blockchain consensus mechanisms, employing multi-agent voting to reduce the reliance on single-agent decisions. However, this method requires each agent to independently execute the full RCA workflow, neglecting hallucinations induced by context-switching across disparate domain knowledge during multi-step reasoning. Furthermore, the absence of historical diagnostic experience leads agents to overlook critical clues or collect irrelevant data, exacerbating hallucination risks.

To address the aforementioned limitations, in this paper, we propose MA-RCA (multi-agent root cause analysis), a collaborative framework deploying specialized agents for distinct subtasks. Following the manual RCA pipeline, MA-RCA assigns dedicated agents to data analysis, runtime data collection, hypothesis generation and validation, and report generation. By coordinating task allocation with an RCA Agent, each agent operates within a dedicated

domain, minimizing context-switching error propagation during multi-step reasoning. Crucially, to counteract the scarcity of actionable information in anomaly alerts, MA-RCA introduces a Retrieval Agent. This agent grounds hypotheses in historical diagnostic experience and domain knowledge, thereby expanding the hypothesis space and uncovering latent indicators for validation. Subsequently, a Validation Agent employs dynamic testing tools to verify hypotheses against runtime data, effectively suppressing hallucinations and cascading errors. Comprehensive evaluations across cloud-native platforms (Nezha [31]) and power metering datasets validate the superiority of our proposed MA-RCA. Specifically, MA-RCA achieves 95.8% accuracy ($F1 = 0.952$) on Nezha and 84.3% accuracy ($F1 = 0.828$) on the Power monitoring dataset, outperforming single-agent baselines by a large margin.

The contributions of this paper are as follows:

- To counteract LLM hallucinations induced by context-switching during multi-step RCA reasoning, we propose a multi-agent framework deploying specialized agents for distinct subtasks. Each agent operates within a dedicated domain, minimizing context switching and reducing hallucination risks.
- To mitigate error propagation between agents, we introduce external historical diagnostic experience and domain knowledge. This grounds hypotheses in verified evidence, thereby expanding the hypothesis space and uncovering latent indicators for further validation.
- Extensive experiments on two datasets demonstrate that our proposed MA-RCA could achieve the best performance among all the baselines.

The rest of this paper is organized as follows: “[Related work](#)” introduces the related work on traditional and LLM-based RCA methods; “[Method](#)” describes the four main agents (RCA Agent, Retrieval Agent, Validation Agent, and Report Agent) of the proposed MA-RCA framework; “[Evaluation](#)” reports the extensive experimental results on two datasets, including ablation studies and performance comparisons; “[System illustration](#)” describes the intelligent diagnosis system for real-world power metering infrastructures deploying MA-RCA; and “[Conclusions](#)” provides the conclusions, summarizing the contributions and future research directions.

Related work

Traditional RCA

RCA is critical for maintaining operational stability in complex integrated systems, yet it remains a time-intensive

process in system maintenance workflows. Traditional RCA depends heavily on manual expertise from SREs. However, training experienced SREs requires several years, creating scalability challenges as systems grow more complex and deployments expand [1]. This reliance on human experts slows response times and leads to inconsistent diagnoses, especially when facing unprecedented failures or staff changes. To address these limitations, early automated approaches [9, 16, 36] employed rule-based systems and statistical heuristics to directly map symptom descriptions to root causes. While effective in static homogeneous environments, these methods fail in complex systems where a single anomaly manifestation may stem from diverse underlying factors (e.g., concurrent hardware faults and configuration errors). Subsequently, machine learning [11, 15, 19, 30] and deep learning [2, 20, 27] methods were developed to incorporate multi-modal runtime data (e.g., logs, metrics, traces), establishing causal mappings between anomalies and root causes through data-driven inference. However, these techniques exhibit limited generalization capability due to their dependence on high-quality labeled datasets, which are challenging to acquire in production environments (e.g., sparse fault labels in power grids).

LLM-based RCA

Automating RCA necessitates addressing three core challenges derived from manual SREs workflows: (1) the complex mapping between anomaly descriptions and runtime data, (2) the semantic gap between unstructured logs and structured metrics, and (3) the combinatorial explosion of causal relationships when inferring root causes. LLMs offer potential solutions due to their exceptional capabilities in processing heterogeneous data, logical reasoning, and knowledge linking. Pioneering method [1] proposed fine-tunes LLMs on domain-specific datasets to generate root causes directly from incident titles and descriptions. While demonstrating initial promise, these approaches face scalability limitations: collecting high-quality training data for complex systems is challenging, and continuous model retraining becomes infeasible for state-of-the-art LLMs as systems evolve. To circumvent fine-tuning, Chen et al. [4] proposed predefined handlers for data collection, followed by LLM-based analysis. However, static handlers lack adaptability to dynamic system topologies. Subsequent frameworks like REACT [24] introduced tool-augmented agents for autonomous data gathering, while RCAgent [28] deployed LLM-based agents for end-to-end RCA task planning. Despite achieving workflow automation, these methods suffer from LLM hallucinations, where agents generate plausible but ungrounded causal hypotheses. To mitigate this, inspired by blockchain consensus, mABC [33] adopted multi-agent voting to reduce reliance on single-agent deci-

sions. Crucially, mABC requires each agent to independently execute the full RCA workflow, thereby neglecting hallucinations induced by context-switching during multi-step reasoning (e.g., transitioning from log parsing to dependency analysis). Furthermore, the absence of historical diagnostic experience leads agents to collect irrelevant data, overwhelming LLM context windows and diluting critical evidence. This data noise amplifies hallucination risks and impedes practical deployment in mission-critical systems like cloud platforms or power grids.

To counteract hallucinations in LLMs during RCA, we propose MA-RCA, a framework leveraging a multi-agent architecture that decomposes the RCA workflow into specialized subtasks. Each agent operates within a dedicated domain, thereby minimizing context switching and reducing error propagation across reasoning steps. Concurrently, historical diagnostic experience is integrated to guide agents to expanding the hypothesis space and uncovering latent indicators for further validation. This dual mechanism, the agent specialization and the knowledge grounding, could effectively suppresses hallucinations.

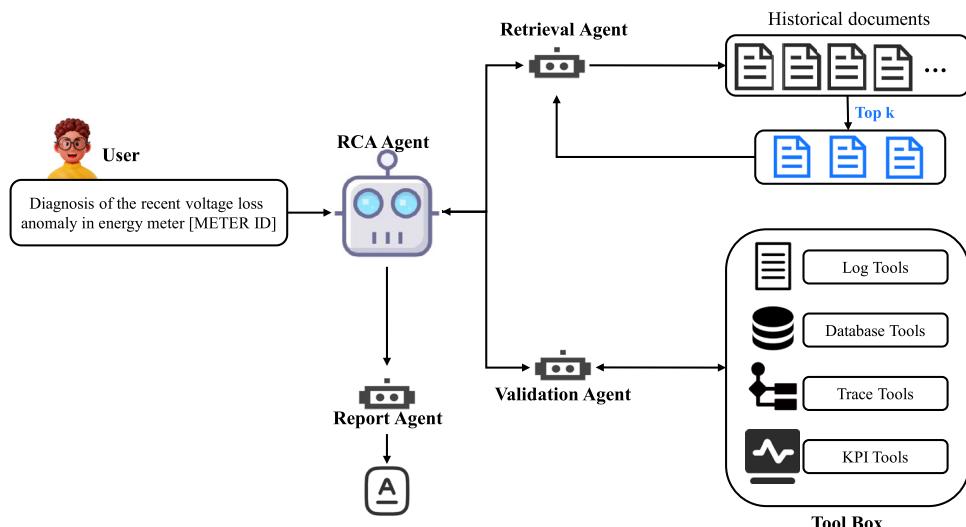
Method

As illustrated in Fig 1, the proposed MA-RCA framework supports industrial anomaly diagnosis via a modular and collaborative architecture. It integrates four core intelligent agents: the RCA Agent, Retrieval Agent, Validation Agent, and Report Agent. These agents operate in conjunction with a historical anomaly case repository and a tool repository, enabling automated and context-aware fault resolution in dynamic environments. Unlike single agent-based approaches, MA-RCA decomposes RCA workflows into specialized tasks. Each task is assigned to a dedicated agent equipped with domain-specific tools and tailored LLM capabilities. This structure reduces error propagation, thereby overcoming hallucinations of single-agent systems. The following sections elaborate on the design principles and operational mechanisms of each component.

RCA Agent

As the central orchestrator of the multi-agent diagnostic framework, the RCA Agent executes three critical functions: (1) semantic parsing of unstructured anomaly reports, (2) coordination of distributed diagnostic workflows, and (3) probabilistic synthesis of root cause hypotheses. The agent processes user queries through a prompt-engineered pipeline that first extracts structured diagnostic parameters (device ID, anomaly type, temporal context), then initiates evidence-driven investigation workflows, and finally applies causal reasoning over multi-source findings. Its decision-making

Fig. 1 The overview of our proposed MA-RCA, a collaborative framework deploying specialized agents for distinct subtasks. It integrates four core intelligent agents: the RCA Agent, Retrieval Agent, Validation Agent, and Report Agent. These agents operate in conjunction with a historical anomaly case repository and a tool repository, enabling automated and context-aware fault resolution in dynamic environments



architecture combines deterministic pattern matching with probabilistic association mining, enabling systematic navigation through historical precedents and real-time telemetry data.

Input parsing mechanism

In RCA systems, robust input parsing forms the foundational layer for ensuring diagnostic validity. The anomaly input submitted by SREs, typically unstructured (e.g., “Meter [METER ID] experienced sudden voltage drop during the afternoon”), presents three key processing challenges: semantic ambiguity, informational redundancy, and terminological heterogeneity. Direct processing of such raw inputs risks propagating error accumulation through subsequent retrieval and verification phases.

To mitigate error propagation in downstream diagnostic workflows, our framework implements strict entity extraction across four mandatory parameters, including device type (e.g., energy meter, sensor), symptom (e.g., voltage loss, communication failure), device ID (exact hardware identifier), and timeframe (temporal context). This methodology not only resolves linguistic ambiguities in natural language descriptions (e.g., distinguishing between transient voltage sags and sustained under-voltage conditions) but also enables implicit association recognition (such as identifying known defect patterns specific to particular device models), thereby achieving dual optimization of recall rate in historical case retrieval and operational precision in verification task scheduling.

As illustrated in Fig 2, the parsing module enforces completeness checks prior to Retrieval Agent invocation. For incomplete inputs, a closed-loop validation protocol triggers user query with explicit missing parameter notifications (e.g., “Missing required fields: [device id, timeframe]”).

This protocol ensures information fidelity and semantic consistency during subsequent knowledge retrieval processes.

Agent scheduling

Effective anomaly diagnosis necessitates the orchestration of heterogeneous tasks including historical experience retrieval, real-time data validation, and professional report generation, which exceeds the capability of single-function modules. As the core scheduling hub of the multi-agent collaborative framework, the RCA Agent implements a dynamic decision engine to coordinate the temporal execution logic and information exchange among the Retrieval Agent, Validation Agent, and Report Agent through the following workflow:

- **Historical pattern matching:** Leveraging parsed entities (device identifiers, anomaly signatures, spatiotemporal contexts) to dispatch the Retrieval Agent for querying historical case repositories and identifying analogous/relevant incidents;
 - **Real-time evidence verification:** Activating the Validation Agent to perform cross-validation between retrieved patterns and current system telemetry;
 - **Causal synthesis & reporting:** Coordinating the Report Agent to generate diagnostic conclusions through probabilistic fusion of validation results and historical root cause associations.

Retrieval Agent

Conventional RCA methodologies relying on manual expertise recollection or fragmented case repositories exhibit inherent limitations of knowledge fragmentation and limited dynamic adaptability. When confronting complex system failures, singular manifestations (e.g., “grid voltage

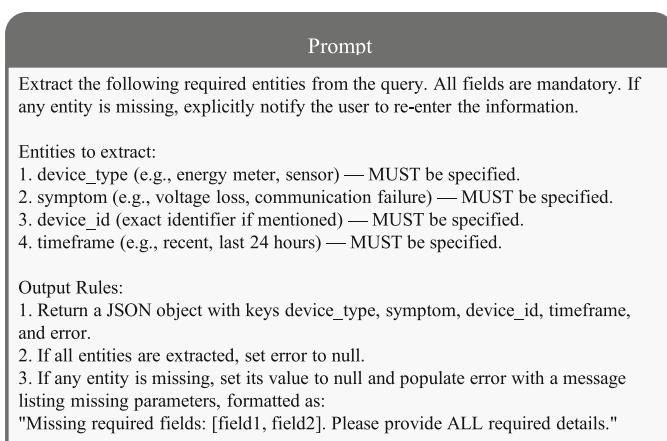
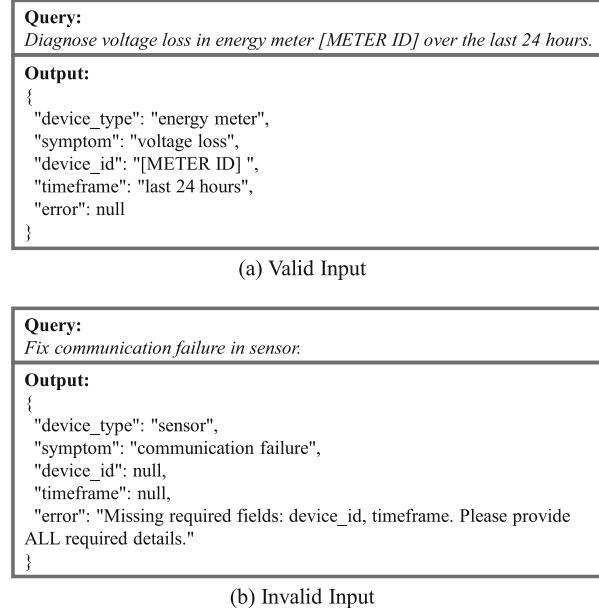


Fig. 2 The input parsing mechanism of RCA Agent, which enforces structured extraction of critical entities from user queries. Valid inputs generate complete outputs with null error fields, while missing entities trigger explicit error messages listing omitted parameters. Examples

sag") often correlate with multidimensional potential root causes (capacitor aging, harmonic distortion, or topological anomalies). Manual analysis requiring exhaustive historical experience traversal coupled with subjective hypothesis dependency frequently leads to diagnostic misinterpretations or critical causal chain omissions. The proposed Retrieval Agent addresses these limitations through a dual mechanism of structured experience reuse and multi-modal causal inference, providing intelligent augmentation to the RCA Agent's hypothesis generation process. The whole workflow of our Retrieval Agent is shown in Algorithm 1.

Case retrieval

The case retrieval module, designed with the core objective of reusing historical experience, achieves precise cross-domain and cross-temporal case matching through a hierarchical indexing architecture. During the data preprocessing phase, it implements multi-granular segmentation of historical case repositories – partitioning technical documents according to IEC standards and dividing operational records into time windows – while constructing a hybrid vector space that integrates textual semantic embeddings, temporal feature encodings, and domain knowledge graph triples to establish symptom-root cause mappings. The retrieval process employs a hybrid recall strategy combining keyword matching (for exact retrieval of canonical cases like “voltage sag”) with semantic expansion (covering variant descriptions such



demonstrate both scenarios: **a** a valid query with all entities detected, and **b** an invalid query prompting user re-entry due to incomplete device ID and timeframe

Algorithm 1: The workflow of Retrieval Agent

```

input : The parsed parameters: symptom s, timeframe t, device type d; The number N of required historical cases
output: The mapping M of root causes and symptoms
1 Initialize an empty set of historical cases H and an empty mapping M ;
/* Case Retrieval */ 
2 Keyword matching: Retrieve historical cases with s and save to H;
3 Semantic retrieval: Retrieve historical cases with the s and d, and save to H ;
4 Re-rank the cases in H by the semantic similarity and time decay ;
5 Reserve the top-N cases from the ranked H ;
/* Root Cause Inversion */ 
6 for each case c in H do
7   Use LLM to parse the root cause r and symptoms S ;
8   Use LLM to abstract concrete symptoms into generalized features and update S ;
9   Use LLM to perform logical verification and conflict resolution and update S ;
10  Put r and S to the mapping M ;
11 end for

```

as “potential drop” through synonym libraries), followed by cross-encoder reranking of Top-K candidate cases that prioritizes temporal relevance weighting and domain-specific rule filtering (e.g., excluding external interference factors like lightning events).

Root cause inversion

The proposed root cause inversion mechanism addresses two fundamental bottlenecks in conventional case retrieval: symptom-root cause association gaps and insufficient coverage of long-tail root causes, particularly critical in dynamically evolving industrial ecosystems (e.g., power grid equipment upgrades, healthcare process optimization). Historical case repositories exhibit fragmented causal mappings where single root causes (e.g., “capacitor aging” in web reference) manifest through diverse symptom patterns (12 distinct voltage waveforms in power grid cases). Leveraging LLM-powered semantic-driven reverse index construction and multi-level causal generalization mechanisms, this module transforms fragmented case-specific cause-symptom pairs into interpretable, extensible causal knowledge units, generating structured inputs for efficient retrieval and validation. The core technical implementation is as follows:

- **Reverse index construction:** Employing LLMs for semantic deconstruction of historical cases to extract root cause entities (e.g., “capacitor aging”) and associated symptom sets (voltage sag, harmonic distortion rate $>5\%$), and construct millisecond-response “cause \rightarrow symptoms” inverted indices;
- **Symptom generalization mapping:** Utilizing LLM contextual reasoning to abstract concrete symptoms (e.g., “API latency $>500\text{ ms}$ ”) into generalized features (network I/O anomalies, computational resource contention), establishing cross-scenario causal association rules;
- **Dynamic causal knowledge integration:** Performing logical verification and conflict resolution through domain knowledge graphs (e.g., IEC 62053 equipment failure taxonomy), eliminating pseudo-associations violating physical principles (e.g., invalid “lightning disaster \rightarrow firmware logic error” mappings).

Validation Agent

The Validation Agent systematically verifies root cause hypotheses through automated evidence integration and cross-domain data correlation. Its core innovation lies in converting manual diagnostic assumptions into quantifiable validation tasks, addressing the inefficiency and subjectivity of traditional verification methods. The workflow of this module is shown as Algorithm 2.

Hypothesis Validation Protocol

The Hypothesis Validation Protocol transforms unstructured diagnostic conjectures into systematic verification workflows through three core technical innovations: automated task deduplication, prioritized evidence collection, and dynamic

Algorithm 2: The workflow of Validation Agent

```

input : The mapping  $M$  of root causes and symptoms
output: 1. The matching score of each root cause hypothesis; 2.
          Differences between historical cases and real-time
          symptoms

1 Initialize an empty mapping  $M_r$  and a empty set of indicators  $I$  ;
2 for each root causes  $r$  in  $M$  do
3   | Read the symptoms  $S$  of  $r$  in  $M$  ;
4   | for each symptom  $s$  in  $S$  do
5     |   | Use LLM to parse the indicator  $i$  of  $s$  ;
6     |   |  $i \rightarrow I$  ;
7   | end for
8 end for
9 Remove duplicate indicators in  $I$  ;
10 for each indicator  $i$  in  $I$  do
11   | Choose the corresponding tool to read the real-time data of  $i$  ;
12   | Return the abnormal result of  $i$  ;
13 end for
14 Refer to  $M$  and construct the  $M_r$  with real-time results ;
15 for each root cause hypothesis  $r$  in  $M_r$  do
16   | Read the corresponding real-time results  $R$  of  $r$  in  $M_r$  ;
17   | Read the corresponding symptoms  $S$  of  $r$  in  $M$  ;
18   | Use LLM to generate the descriptions of  $R$  and  $S$  ;
19   | Use semantic similarity to compute the matching scores of
      these two descriptions ;
20   | Use LLM to generate the differences between these two
      descriptions ;
21 end for

```

cache invalidation. This protocol addresses the computational overhead of validating concurrent hypotheses in large-scale systems while ensuring verification integrity.

Upon receiving a set of candidate root causes (e.g., “capacitor aging” or “firmware defects”) from the RCA Agent, the protocol initiates a three-phase verification cycle. First, it decomposes hypotheses into quantifiable validation metrics, transforming “capacitor aging” into temperature thresholds ($>85^\circ\text{C}$), harmonic distortion levels ($>5\%$), and capacitance deviation ($>15\%$ from nominal values). These metrics undergo similarity clustering using LSH, grouping overlapping verification requirements. For instance, both capacitor aging and thermal runaway hypotheses might share temperature validation needs, triggering a single sensor data retrieval operation.

The protocol then orchestrates toolchain execution through priority-aware scheduling. Critical real-time metrics like CPU load or network latency receive immediate attention through direct API calls to monitoring systems (Prometheus for cloud infrastructure, Modbus/TCP for industrial controllers), while background validations such as log analysis or configuration audits queue for batch processing. A distributed cache pool with TTL rules stores and shares results across hypothesis threads—volatile metrics like ambient temperature refresh every 30s, whereas stable parameters like firmware versions persist for up to an hour. This dual-layer optimization reduces redundant tool invocations in

cloud infrastructure trials while maintaining result consistency compared to isolated validation processes.

Multi-source data convergence and confidence scoring

To compute confidence scores, the agent leverages semantic alignment between historical symptom patterns and real-time observations. LLMs transform structured data—such as numerical metrics, log entries, or trace spans—into contextualized natural language descriptions. These descriptions enable semantic similarity scoring using embedding models (e.g., BERT), which quantify the alignment between real-time anomalies and historical failure signatures. For example, a hypothesis linking “capacitor aging” to voltage sags is evaluated by comparing real-time voltage waveforms (sampled at 10 Hz from PMUs) against historical degradation patterns, while harmonic distortion metrics are validated against thresholds derived from equipment specifications.

Context-aware data-to-text conversion with LLMs bridges gaps between structured metrics and unstructured domain knowledge. The validation cache pool optimizes performance by reusing shared data across hypotheses, while semantic similarity scoring replaces rigid threshold-based rules, enhancing resilience to data variability. This functionality not only automates hypothesis verification but also provides auditable evidence chains—critical for compliance-driven industries.

Report Agent

The Report Agent serves as the unified output layer of the multi-agent RCA framework, transforming raw diagnostic data into structured reports for SREs and downstream systems. By integrating historical case matches, real-time validation metrics, and causal reasoning results, it generates compliance-ready documentation with human interpretability. The agent employs adaptive templates to organize findings into standardized sections, such as Basic Situation (summarizing anomaly context and system configurations), Evidence Chain, and Mitigation Recommendations. For instance, in a voltage anomaly scenario, the report highlights discrepancies between rated and measured voltages (e.g., “threshold ratio < 0.9”) and suggests manual terminal inspections or data revalidation.

To ensure interoperability, reports are exported in PDF formats for audit trails and automated parsing. By automating these workflows, the Report Agent reduces manual report generation time while adhering to industry standards like IEC and NERC, ensuring both efficiency and regulatory compliance.

Table 1 The statistics of Nezha dataset

TrainTicket	Resource	CPU contention	7
		Network delay	14
		Code	11
	Code	Error return	13
		Code defects	13
		OnlineBoutique	16
OnlineBoutique	Resource	CPU contention	16
		Network delay	16
		CPU consumed	10
	Code	Error return	7
		Code defects	7
		Code	7

Evaluation

Datasets

To validate the effectiveness of our proposed MA-RCA, we conduct comprehensive experiments on two datasets, which are collected from real-world scenarios.

Nezha dataset [31] serves as a multi-modal observability benchmark constructed from two open-source microservice architectures: the OnlineBoutique e-commerce platform and the TrainTicket railway reservation system. These representative implementations model distinct cloud-native operational patterns: OnlineBoutique encapsulates core e-commerce transactions, whereas TrainTicket handles complex ticketing workflows involving real-time inventory synchronization and payment gateways. To mimic production-level faults, two principal issues are injected across both systems: resource issues (e.g., CPU contention, network latency) and code defects (e.g., error return, exception code defects). The fault distribution comprises 56 instances in OnlineBoutique (42 resource, 14 code) and 45 instances in TrainTicket (21 resource, 24 code), as detailed in Table 1.

To adapt this dataset for our proposed MA-RCA evaluation, we implement a stratified partitioning strategy that preserves fault-type distributions, allocating 50% of instances per category for historical case construction and 50% for evaluation. The Validation Agent operates on pre-processed telemetry data (logs, metrics, traces) rather than live system instrumentation, enabling deterministic reproducibility. We specifically isolate key service-level indicators such as success rates and CPU utilization metrics to simulate real-world SREs diagnostic scenarios, where engineers typically initiate RCA investigations based on singular observable metric anomalies. This methodology establishes controlled experimental conditions while maintaining operational relevance to production debugging practices.

The Power monitoring dataset originates from power metering infrastructure, provides validated root cause-symptom correlations across two critical failure domains, as shown in Table 2: (1) ICT Infrastructure Anomalies exhibit patterns of

computational resource exhaustion (CPU throttling, memory leakage), storage subsystem failures (disk I/O contention, filesystem corruption), database connectivity breakdowns (connection pool exhaustion, query deadlocks), and network performance degradation (latency spikes, packet loss anomalies); (2) Electrical System Faults encompass power circuit abnormalities including short-circuit faults, open circuit faults, grounding faults, and electrical equipment faults. The pre-processed method is similar to Nezha. Following the above pre-process methodology in the Nezha dataset, we implement identical preprocessing pipelines for temporal alignment and feature extraction across heterogeneous data sources. This includes stratified partitioning of incident records preserving fault-type distributions, synthetic generation of monitoring telemetry streams, and abstraction of KPIs such as harmonic distortion levels and voltage sag characteristics.

Baselines

We compare the proposed MA-RCA with three LLM-based approaches:

- CoT implements structured reasoning decomposition through explicit intermediate step generation [29]. Our implementation combines the prompt with few-shot exemplars from historical cases.
- RAG establishes knowledge grounding through semantic retrieval from the vector database [6]. We leverage MA-RCA’s historical case repository as the database and employ cosine similarity-based retrieval of the top 5 most contextually relevant cases for each input symptom profile.
- RCACOPILOT represents workflow-driven diagnosis integrating multi-source telemetry analysis [4]. We adapt its predefined diagnostic pipelines to our experimental context, maintaining original design principles for log-trace-metric correlation and automated evidence collection while aligning root cause taxonomies with our evaluation framework.
- RCAgent [28] uses LLM-based agent to recursively extract and summarize key details from logs and codes, helping to identify the root causes of system failures.
- mABC [33] draws inspiration from blockchain consensus mechanisms, employing multi-agent voting to reduce the reliance on single-agent decisions.

Given the classification nature of our experimental setup, we employ four principal evaluation metrics: Accuracy measures overall prediction correctness across fault categories; Precision quantifies class-specific prediction validity by comparing correct identifications against total predictions; Recall assesses detection completeness through the percent-

age of successfully identified true positives per fault class; and F1-score provides balanced performance assessment via harmonic mean integration of precision and recall.

Overall performance

As shown in Table 3, the experimental results demonstrate the superiority of the proposed MA-RCA framework across both evaluated domains: the cloud-native Nezha platform and the power monitoring infrastructure. MA-RCA achieves state-of-the-art performance, significantly outperforming all baselines in accuracy, precision, recall, and F1-score. The performance disparities highlight limitations in existing approaches and validate the design choices underpinning our multi-agent system.

Conventional methods exhibit critical shortcomings. CoT yields the lowest performance ($F1 = 0.412$ on Nezha; $Acc = 0.117$ on Power), underscoring the insufficiency of closed-loop reasoning without external grounding. Its low precision, particularly on the power dataset ($Pre = 0.100$), indicates a propensity for generating unsubstantiated hypotheses. RAG shows moderate improvement (Nezha $F1 = 0.704$) by incorporating historical cases, yet its performance remains constrained by semantic retrieval limitations. The reliance on cosine similarity often returns textually similar but causally irrelevant cases, failing to ensure diagnostic precision.

Workflow-driven and agent-based methods form a competitive middle group. RCACOPILOT achieves respectable results (Nezha $F1 = 0.748$) through predefined data correlation pipelines, but its inflexibility hinders adaptation to novel or dynamic failure modes. RCAgent and mABC, employing single and multi-agent paradigms respectively, demonstrate comparable performance (Nezha $F1 = 0.725$ and $F1 = 0.726$). However, their architectures inherently propagate errors. RCAgent suffers from context-switching hallucinations within a single agent performing all tasks. While mABC mitigates this via multi-agent voting, it requires each agent to execute the entire workflow independently, thus amplifying computational overhead without fundamentally addressing grounding issues. All three methods share a critical weakness: they operate on a limited hypothesis space derived solely from the immediate anomaly context, lacking mechanisms to incorporate broader historical knowledge for hypothesis expansion or validation.

The proposed MA-RCA framework addresses these limitations through a specialized multi-agent architecture, achieving superior performance (Nezha $Acc = 0.958$, $F1 = 0.952$; Power $Acc = 0.843$, $F1 = 0.828$). Its superiority stems from three pivotal design innovations. First, task decomposition assigns specialized agents to discrete subtasks—retrieval, analysis, validation, and reporting. This division of labor minimizes context-switching and error propagation between individuals. Second, the dedicated Retrieval Agent proac-

Table 2 The statistics of Power monitoring dataset

ICT infrastructure	Computational resource		CPU throttling	10
			Memory leakage	20
	Storage system		Disk I/O contention	15
			Filesystem corruption	6
	Database		Connection pool exhaustion	30
			Query deadlocks	11
	Network		Latency spikes	20
			Packet loss anomalies	19
	Electrical system		Short-circuit faults	36
			Open circuit faults	26
Grounding faults				22
		Electrical equipment faults		16

Table 3 RCA results on Nezha and Power monitoring dataset

Method	Nezha				Power monitoring			
	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
CoT	0.308	0.350	0.500	0.412	0.218	0.100	0.140	0.117
RAG	0.740	0.628	0.800	0.704	0.571	0.505	0.580	0.540
RCACOPilot	0.745	0.721	0.777	0.748	0.643	0.681	0.623	0.651
RCAgent	0.740	0.650	0.820	0.725	0.580	0.550	0.640	0.591
mABC	0.742	0.680	0.780	0.726	0.600	0.580	0.650	0.613
MA-RCA (Ours)	0.958	0.960	0.944	0.952	0.843	0.827	0.830	0.828

tively queries a historical case repository. This critical step grounds the subsequent reasoning process in verified evidence and, more importantly, expands the initial hypothesis space with latent indicators derived from past diagnoses, countering the scarcity of information in raw alerts. Third, the Validation Agent employs dynamic tools to test generated hypotheses against runtime system data. This verification actively suppresses hallucinations and cascading errors, ensuring only evidence-supported conclusions are finalized.

The observed performance gap between the two domains further validates our approach. All methods perform better on the Nezha dataset, likely due to more structured logging and clearer causal pathways in cloud systems. The power monitoring environment presents greater challenges, including heterogeneous data sources and complex physical interdependencies. MA-RCA still achieves large margin of improvement on the power dataset underscores its ability to handle domain-specific complexities through its knowledge-guided and validation-driven architecture.

Ablation study

To systematically evaluate the contributions of MA-RCA's modular components, we conduct an ablation study on the Nezha (cloud-native) and Power Monitoring (cyber-physical) datasets, designing four functionally degraded variants (Table 4):

- **w/o Multi-agent:** A monolithic LLM architecture consolidates all RCA tasks – symptom parsing, historical case retrieval, tool validation, and diagnosis – without domain-specialized agents, exposing context-switching overhead and error propagation risks;
- **w/o Retrieval Agent:** Deactivates the hybrid (keyword+semantic) case retrieval and LLM-based causal inversion, restricting diagnostic scope exclusively to real-time telemetry and symptom descriptions while halting historical knowledge integration;
- **w/o Semantic retrieval:** Restricts case retrieval to keyword matching, eliminating LLM-driven semantic expansion (e.g., synonym resolution, symptom abstraction) to quantify terminology generalization loss;
- **w/o Validation Agent:** Removes tool-mediated hypothesis verification (e.g., metric queries, physics-aware checks), reducing the workflow to a RAG-based method prone to unvalidated hallucinations.

On the Nezha dataset, the full MA-RCA framework achieves near-perfect performance (Acc: 0.958, F1: 0.952), demonstrating its robustness in handling service dependency chains and real-time telemetry correlation. However, ablating core components exposes severe degradation. The **w/o Multi-agent** variant, which collapses the multi-agent workflow into a monolithic LLM, suffers catastrophic performance collapse (F1: 0.716 on Nezha, 0.680 on Power),

Table 4 Ablation study on Nezha and Power monitoring dataset

Method	Nezha				Power monitoring			
	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
w/o multi-agent	0.729	0.774	0.666	0.716	0.735	0.742	0.627	0.680
w/o Retrieval Agent	0.275	0.361	0.250	0.295	0.112	0.124	0.173	0.144
w/o semantic retrieval	0.902	0.892	0.856	0.874	0.716	0.685	0.643	0.663
w/o Validation Agent	0.749	0.650	0.715	0.681	0.562	0.581	0.510	0.543
MA-RCA (Ours)	0.958	0.960	0.944	0.952	0.843	0.827	0.830	0.828

underscoring the indispensability of specialized agents for task decomposition. Without the Retrieval Agent’s semantic grounding (**w/o Retrieval Agent**), performance plummets further (F1: 0.295 on Nezha, 0.144 on Power), as the model loses access to historical causal patterns, particularly in Power’s hybrid ICT-electrical failures where cross-domain symptom-cause mappings are sparse. Replacing hybrid retrieval with keyword matching (**w/o Semantic retrieval**) halves the F1 gap on Nezha (0.874 vs. 0.952) but cripples Power (0.663 vs. 0.828), reflecting the latter’s reliance on contextual nuance—electrical terms like “harmonic distortion” resist keyword reduction, whereas cloud-native issues like “API latency” tolerate partial lexical matches. Disabling the Validation Agent (**w/o Validation Agent**) mimics classical RAG’s limitations, inflating false positives (Precision: 0.650 on Nezha, 0.581 on Power) due to unchecked LLM hallucinations, especially in Power’s physics-driven scenarios where hypotheses like “short-circuit” require tool-verified metrics (e.g., voltage sag profiles) to avoid spurious correlations.

The Power dataset’s stark sensitivity to component removal—its F1 drops 72% without the Retrieval Agent versus Nezha’s 69%—highlights the compounded challenges of diagnosing hybrid faults (e.g., memory leaks masquerading as voltage sags). Here, semantic retrieval and multi-agent collaboration are non-negotiable: the Validation Agent’s tool-in-the-loop checks (e.g., querying harmonic distortion levels) suppress misattributed root causes, while the Retrieval Agent’s causal abstraction bridges ICT metrics (CPU throttling) and electrical KPIs (capacitor aging). In contrast, Nezha’s cloud-centric faults, while complex, benefit from more self-contained telemetry (traces, logs) that partially compensate for missing agents—explaining why **w/o Semantic retrieval** retains 87.4% F1 versus Power’s 66.3%. Notably, both datasets suffer equally from monolithic LLM designs (**w/o Multi-agent**), where fused task handling amplifies error propagation: Precision plunges to 0.774 on Nezha and 0.742 on Power, as undivided LLMs conflate symptom interpretation (e.g., misclassifying network latency as code defects) without agent-driven validation or retrieval-augmented context.

These results cement MA-RCA’s architectural superiority. Its multi-agent synergy – semantic retrieval for knowledge grounding, validation for hypothesis pruning, and tool inte-

Table 5 Performance of different base LLMs on Power monitoring dataset

LLM base	Acc	Pre	Rec	F1
Qwen-QwQ	0.824	0.808	0.810	0.809
LLAMA3.3-70B	0.830	0.814	0.816	0.815
DeepSeek	0.835	0.819	0.821	0.820
QwenMax	0.843	0.827	0.830	0.828

gration for domain-specific verification – proves vital in both cloud and cyber-physical domains. The framework’s resilience to single-symptom inputs (e.g., “API latency >500 ms” or “voltage sag detected”) stems from this layered rigor: agents collaboratively disambiguate sparse signals, whereas ablated variants regress to brittle heuristics.

Influence of base LLMs

Table 5 demonstrates that MA-RCA’s performance is consistently robust across different base LLMs, yet subject to model capability variations. QwenMax [26] achieves the highest scores (Acc = 0.843, F1 = 0.828), reflecting its strong reasoning and knowledge integration capacity. In contrast, Qwen-QwQ [26], with lower results (Acc = 0.824, F1 = 0.809), highlights limitations in contextual understanding or logical inference. Intermediate models like LLAMA3.3-70B [7] and DeepSeek [14] exhibit scalable performance, correlating with model scale and training data quality. These results affirm that MA-RCA’s multi-agent structure benefits effectively from enhanced base LLMs, while its design mitigates inherent model weaknesses through structured collaboration and validation.

Influence of the number of historical cases

Table 6 demonstrates that the number of retrieved historical cases significantly influences MA-RCA’s performance. Accuracy and F1-score improve as cases increase from 1 to 5 (Acc: 0.798 → 0.843, F1: 0.794 → 0.828), indicating enhanced hypothesis grounding through richer contextual knowledge. However, further increasing to 10 or 20 cases results in performance degradation (Acc: 0.835, 0.812; F1:

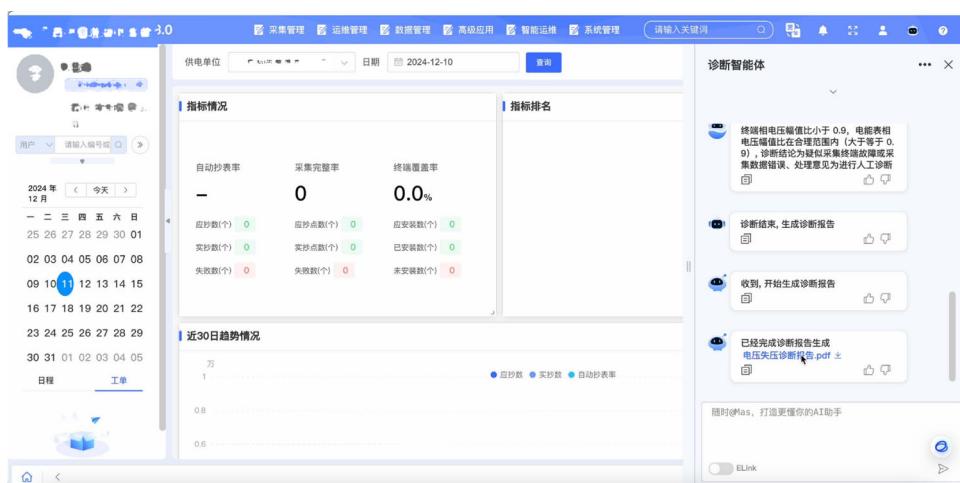
Table 6 Performance of different number of historical cases on Power monitoring dataset

Retrieval cases	Acc	Pre	Rec	F1
1	0.798	0.783	0.805	0.794
3	0.826	0.812	0.831	0.821
5	0.843	0.827	0.830	0.828
10	0.835	0.818	0.822	0.820
20	0.812	0.796	0.804	0.800

Fig. 3 User interface of the power metering system. The proposed MA-RCA method is integrated as a plug-in, referred to as the “Diagnosis Agent” (On the right side)



Fig. 4 Diagnosis Agent utilizes historical cases and current KPI data to analyze the root cause of the user-described issue and generate a corresponding report



0.820, 0.800), suggesting the introduction of irrelevant or noisy instances that dilute diagnostic relevance. This non-monotonic behavior underscores the importance of optimal retrieval setting: too few cases limit knowledge utilization, while too many introduce distraction, affirming that precise retrieval quantity is critical for balancing informativeness and focus in evidence-based RCA.

System illustration

To empirically validate the operational efficacy of MA-RCA in industrial scenarios, we developed an intelligent diagno-

sis system for real-world power metering infrastructures via lightweight plugin integration. As shown in Fig. 3, the system incorporates a “Diagnosis Agent” (on the right side) that implements the multi-agent coordination framework introduced in Sect. 3. This agent serves as a cognitive aid for SREs, allowing them to initiate diagnostic workflows by submitting alert descriptions through a natural language interface.

Upon receiving an unstructured alert from the user (Fig. 2a), the agent performs syntactic parsing and validates the input against a predefined incident schema, as detailed in Sect. 3.1.1. This ensures all essential diagnostic elements, such as device type, device ID, timeframe, and symptom descriptions, are correctly specified.

The analytical process is illustrated in Fig. 4. Once the input is validated, the RCA Agent orchestrates the Retrieval Agent and Validation Agent to identify root causes. The Retrieval Agent queries relevant historical cases using the structured input. Multiple root-cause hypotheses are then constructed. The Validation Agent correlates these with real-time operational data using tool-enabled verification to identify the most probable cause. Finally, the Report Agent generates a comprehensive diagnostic report.

Conclusions

In this paper, we presented MA-RCA, a novel multi-agent collaborative framework designed to automate RCA in complex distributed systems. Confronting the persistent challenges of LLM hallucination and error propagation in multi-step reasoning, MA-RCA decomposes the manual RCA workflow into distinct subtasks managed by specialized agents. Its core innovations include a dedicated Retrieval Agent that grounds the process in historical diagnostic knowledge to expand the hypothesis space, and a Validation Agent that actively tests hypotheses against runtime data to suppress hallucinations. Comprehensive evaluations on cloud-native and power metering datasets demonstrate that MA-RCA substantially outperforms strong baselines including RAG, RCACOPilot, and agent-based models, achieving superior accuracy and robustness across diverse operational environments.

Although MA-RCA demonstrates strong performance, it still has certain limitations: (1) its effectiveness depends heavily on the quality and comprehensiveness of the historical case repository, as sparse or outdated data may constrain the Retrieval Agent's performance; (2) the data collection and validation processes, while essential for accuracy, introduce additional latency compared to direct inference methods, which could affect usability in highly time-sensitive scenarios. These limitations pave the way for concrete future research directions. First, we will investigate adaptive retrieval mechanisms that can dynamically determine the optimal number and relevance of historical cases based on the anomaly's context, moving beyond a static top-k strategy to improve efficiency and accuracy. Second, to address latency, we plan to explore speculative execution and prefetching strategies for the agent team, where potential data collection steps are anticipated and initiated in parallel to streamline the diagnostic pipeline.

Author Contributions All authors contributed to the conception and design of the study. Specifically: Feng Fu: Conceptualization, Methodology, Software development, Original draft preparation, Funding acquisition, Supervision. Hong Ding: Data curation, Project administration. Yong Qin: Validation, Formal analysis. Jian Yu: Writing-review and editing. Dechao Xu: Supervision, Project administration. All

authors critically reviewed and approved the final version of the manuscript.

Funding This work was supported by the State Grid Electric Power Research Institute Co., Ltd. under its 2024 Frontline Production-Oriented Technology Project Package (Electric Power Consumption Branch Company)—Research and Application of Key Technologies for Enhancing Intelligent Capabilities in Metering Operations and Maintenance (Project No. WBS 524609240180).

Data availability Nezha dataset [31] is open-sourced and available at <https://github.com/IntelligentDDS/Nezha>. The “Power monitoring” dataset is collected with real-world power system and contains sensitive information, which will be provided as request.

Declarations

Conflict of interest No Conflict of interest exists in the submission of this manuscript.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

References

1. Ahmed T, Ghosh S, Bansal C, Zimmermann T, Zhang X, Rajmohan S (2023) Recommending root-cause and mitigation steps for cloud incidents using large language models. In: 2023 IEEE/ACM 45th international conference on software engineering (ICSE). IEEE, pp 1737–1749
2. Bin Y, Yang Y, Shen F, Xie N, Shen HT, Li X (2018) Describing video with attention-based bidirectional lstm. IEEE Trans Cybern 49(7):2631–2641
3. Chen H, Chen P, Yu G, Li X, He Z (2024) Microfi: non-intrusive and prioritized request-level fault injection for microservice applications. IEEE Trans Depend Secure Comput 21(5):4921–4938
4. Chen Y, Xie H, Ma M, Kang Y, Gao X, Shi L, Cao Y, Gao X, Fan H, Wen M et al (2023) Empowering practical root cause analysis by large language models for cloud incidents. arXiv preprint [arXiv:2305.15778](https://arxiv.org/abs/2305.15778)
5. Chen Z, Luo Y, Wang S, Li J, Huang Z (2022) GSMFlow: generation shifts mitigating flow for generalized zero-shot learning. IEEE Trans Multimed 25:5374–5385
6. Gao Y, Xiong Y, Gao X, Jia K, Pan J, Bi Y, Dai Y, Sun J, Wang H (2023) Retrieval-augmented generation for large language models: a survey. arXiv preprint [arXiv:2312.10997](https://arxiv.org/abs/2312.10997)
7. Grattafiori A, Dubey A, Jauhri A, Pandey A, Kadian A, Al-Dahle A, Letman A, Mathur A, Schelten A, Vaughan A et al (2024) The Llama 3 herd of models. arXiv preprint [arXiv:2407.21783](https://arxiv.org/abs/2407.21783)

8. Guo J, Guo S, Ma S, Sun Y, Xu Y (2021) Conservative novelty synthesizing network for malware recognition in an open-set scenario. *IEEE Trans Neural Netw Learn Syst* 34(2):662–676
9. Guo X, Peng X, Wang H, Li W, Jiang H, Ding D, Xie T, Su L (2020) Graph-based trace analysis for microservice architecture understanding and problem diagnosis. In: Proceedings of the 28th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering, pp 1387–1397
10. Inam MA, Chen Y, Goyal A, Liu J, Mink J, Michael N, Gaur S, Bates A, Hassan WU (2023) SoK: history is a vast early warning system: auditing the provenance of system intrusions. In: 2023 IEEE symposium on security and privacy (SP). IEEE, pp 2620–2638
11. Jia T, Yang L, Chen P, Li Y, Meng F, Xu J (2017) Logsed: anomaly diagnosis through mining time-weighted control flow graph in logs. In: 2017 IEEE 10th international conference on cloud computing (CLOUD). IEEE, pp 447–455
12. Larsson JE, DeBor J (2007) Real-time root cause analysis for complex technical systems. In: 2007 IEEE 8th human factors and power plants and HPRCT 13th annual meeting. IEEE, pp 156–163
13. Li H, Yang Z, Ma Y, Bin Y, Yang Y, Chua TS (2024) MM-Forecast: a multimodal approach to temporal event forecasting with large language models. In: Proceedings of the 32nd ACM international conference on multimedia, pp 2776–2785
14. Liu A, Feng B, Xue B, Wang B, Wu B, Lu C, Zhao C, Deng C, Zhang C, Ruan C et al (2024) Deepseek-v3 technical report. arXiv preprint [arXiv:2412.19437](https://arxiv.org/abs/2412.19437)
15. Liu D, He C, Peng X, Lin F, Zhang C, Gong S, Li Z, Ou J, Wu Z (2021) 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). IEEE, pp 338–347
16. Liu P, Xu H, Ouyang Q, Jiao R, Chen Z, Zhang S, Yang J, Mo L, Zeng J, Xue W et al (2020) Unsupervised detection of microservice trace anomalies through service-level deep Bayesian networks. In: 2020 IEEE 31st international symposium on software reliability engineering (ISSRE). IEEE, pp 48–58
17. Ma M, Zhang S, Pei D, Huang X, Dai H (2018) Robust and rapid adaption for concept drift in software system anomaly detection. In: 2018 IEEE 29th international symposium on software reliability engineering (ISSRE). IEEE, pp 13–24
18. Ma Y, Xiao X, Wang Y (2020) Identifying the root cause of power system disturbances based on waveform templates. *Electr Power Syst Res* 180:106107
19. Nandi A, Mandal A, Atreja S, Dasgupta GB, Bhattacharya S (2016) Anomaly detection using program control flow graph mining from execution logs. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 215–224
20. Nedelkoski S, Cardoso J, Kao O (2019) Anomaly detection and classification using distributed tracing and deep learning. In: 2019 19th IEEE/ACM international symposium on cluster, cloud and grid computing (CCGRID). IEEE, pp 241–250
21. Nguyen H, Shen Z, Tan Y, Gu X (2013) FChain: toward black-box online fault localization for cloud systems. In: 2013 IEEE 33rd international conference on distributed computing systems. IEEE, pp 21–30
22. Ning L, Liang Z, Jiang Z, Qu H, Ding Y, Fan W, Wei XY, Lin S, Liu H, Yu PS et al (2025) A survey of WebAgents: towards next-generation AI agents for web automation with large foundation models. arXiv preprint [arXiv:2503.23350](https://arxiv.org/abs/2503.23350)
23. Pourbeik P, Kundur PS, Taylor CW (2006) The anatomy of a power grid blackout—root causes and dynamics of recent major blackouts. *IEEE Power Energy Mag* 4(5):22–29
24. Roy D, Zhang X, Bhave R, Bansal C, Las-Casas P, Fonseca R, Rajmohan S (2024) Exploring LLM-based agents for root cause analysis. In: Companion proceedings of the 32nd ACM international conference on the foundations of software engineering, pp 208–219
25. Shi W, Hu Z, Bin Y, Liu J, Yang Y, Ng SK, Bing L, Lee RKW (2024) Math-LLaVA: bootstrapping mathematical reasoning for multimodal large language models. In: Findings of the conference on empirical methods in natural language processing, pp 4663–4680
26. Team Q (2024) Qwen2.5 technical report. arXiv preprint [arXiv:2412.15115](https://arxiv.org/abs/2412.15115)
27. Wang L, Zhao N, Chen J, Li P, Zhang W, Sui K (2020) Root-cause metric location for microservice systems via log anomaly detection. In: 2020 IEEE international conference on web services (ICWS). IEEE, pp 142–150
28. Wang Z, Liu Z, Zhang Y, Zhong A, Wang J, Yin F, Fan L, Wu L, Wen Q (2024) RC-Agent: cloud root cause analysis by autonomous agents with tool-augmented large language models. In: Proceedings of the ACM international conference on information and knowledge management, pp 4966–4974
29. Wei J, Wang X, Schuurmans D, Bosma M, Xia F, Chi E, Le QV, Zhou D et al (2022) Chain-of-thought prompting elicits reasoning in large language models. *Adv Neural Inf Process Syst* 35:24824–24837
30. Wu L, Tordsson J, Elmroth E, Kao O (2020) MicroRCA: root cause localization of performance issues in microservices. In: NOMS 2020–2020 IEEE/IFIP network operations and management symposium. IEEE, pp 1–9
31. Yu G, Chen P, Li Y, Chen H, Li X, Zheng Z (2023) 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, pp 553–565
32. Zhang S, Jin P, Lin Z, Sun Y, Zhang B, Xia S, Li Z, Zhong Z, Ma M, Jin W et al (2023) Robust failure diagnosis of microservice system through multimodal data. *IEEE Trans Serv Comput* 16(6):3851–3864
33. Zhang W, Guo H, Yang J, Tian Z, Zhang Y, Yan C, Li Z, Li T, Shi X, Zheng L, Zhang B (2024) MABC: multi-agent blockchain-inspired collaboration for root cause analysis in micro-services architecture. In: Findings of the conference on empirical methods in natural language processing, pp 4017–4033
34. Zhang X, Xu Y, Qin S, He S, Qiao B, Li Z, Zhang H, Li X, Dang Y, Lin Q et al (2021) Onion: identifying incident-indicating logs for cloud systems. In: Proceedings of the 29th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering, pp 1253–1263
35. Zhang Y, Guan Z, Qian H, Xu L, Liu H, Wen Q, Sun L, Jiang J, Fan L, Ke M (2021) CloudRCA: a root cause analysis framework for cloud computing platforms. In: Proceedings of the 30th ACM international conference on information & knowledge management, pp 4373–4382
36. Zhou X, Peng X, Xie T, Sun J, Ji C, Li W, Ding D (2018) Fault analysis and debugging of microservice systems: industrial survey, benchmark system, and empirical study. *IEEE Trans Softw Eng* 47(2):243–260