Journal of Computer Technology and Software

ISSN:2998-2383

Vol. 3, No. 1, 2024

Structure-Aware Root Cause Localization for Microservice Backends via Joint Graph Representation Learning and Causal Inference

Lixing Wang[1*], Chia-Yuan Chang2
1Independent Researcher, Bellevue, USA
2University of Southern California, Los Angeles, USA
*Corresponding author: Lixing Wang; wanglixing244@gmail.com

Abstract-Service oriented architectures improve system flexibility and scalability through decoupling and independent deployment. At the same time, they introduce complex dependencies, hidden anomaly propagation paths, and highly heterogeneous observations. These factors make backend fault root cause localization particularly challenging. In microservice systems, anomalies often cascade along dependency structures, and surface symptoms are separated from true triggers in time and space. To address these characteristics, this paper proposes a backend fault root cause localization method based on microservice dependency graphs. Graph representation learning and causal inference are unified within a single modeling framework. Service invocations and resource dependencies are used to construct a system dependency graph. Logs, metrics, and traces are mapped into node state representations. System-level runtime states are learned under structural constraints. By modeling service interactions on the graph, the method enables system-level understanding of anomaly organization rather than decisions based on local signals alone. Causal modeling is further introduced to capture directional influences among service nodes. Anomaly analysis is thus elevated from correlation-based description to a mechanism-level explanation. This design allows a clear distinction between triggering sources and propagation effects. Comparative analysis under the same evaluation setting shows that the proposed method achieves more stable performance in overall discrimination, result consistency, and system-level anomaly characterization. This work provides a unified modeling approach that is structure-aware and causally interpretable. It improves the reliability of anomaly understanding and diagnosis in complex microservice backend systems.

Keywords: Microservice systems; root cause analysis; service dependency graph; causal modeling

1. Introduction

With the maturation of cloud computing and containerization technologies, backend system architectures are rapidly evolving from monolithic designs to highly decoupled and elastically scalable microservice systems. Fine-grained service decomposition and independent deployment improve scalability and iteration efficiency. At the same time, they introduce more complex dependencies and interaction paths at runtime. Services are coupled through call chains, shared resources, and asynchronous messaging. When a local component becomes abnormal, the impact often propagates along dependency structures. This propagation appears as cross-service performance degradation or functional failure. In this context, traditional operations methods based on single-point monitoring or static rules cannot accurately capture failure propagation mechanisms [1]. As a result, fault localization cycles are prolonged and recovery costs increase. These limitations no longer meet the requirements for high reliability and high availability in modern backend systems [2].

Root cause localization is a core task in system operations and reliability assurance. Its primary objective is to identify the initial trigger of anomalies from complex observations and to distinguish root causes from propagation effects. In microservice environments, this task faces several challenges. First, service scale and invocation relationships continuously evolve with business changes, which leads to highly dynamic dependency structures. Second, system states are described by logs, metrics, and traces. These data sources are heterogeneous and noisy. Third, anomalies rarely occur in isolation. They propagate along dependency paths, which causes temporal and spatial shifts between surface symptoms and true root causes. If service dependencies and state propagation patterns are ignored, localization methods may confuse correlation with causation. This confusion reduces diagnostic accuracy and interpretability [3].

Microservice dependency graphs provide a structured view for understanding overall system behavior. Services or components can be abstracted as nodes, while invocations and resource dependencies are modeled as edges. System states can then be represented as graph structures that evolve. This representation explicitly captures interactions among components. It also offers a natural carrier for modeling anomaly propagation paths [4]. Graph representation learning models node and subgraph embeddings under structural constraints. It helps to learn system-level state patterns from local observations. This capability alleviates the challenges caused by high dimensionality, sparsity, and noise. However, relying only on representation similarity or structural

association cannot answer why an anomaly occurs. It fails to distinguish causal drivers from incidental co-occurrence at the mechanism level.

Causal inference offers an important theoretical tool to address these limitations. Unlike correlation analysis, a causal perspective focuses on intervention effects and generative mechanisms. It models directional influence relationships among variables. In microservice systems, fault generation and propagation often follow explicit dependency constraints and temporal logic [5]. This property implies the existence of latent causal structures. Introducing causal inference into root cause localization helps infer potential causal chains from observed anomaly patterns. It reduces the risk of mistaking downstream symptoms for upstream causes. It also provides more interpretable evidence for decision-making. However, direct causal modeling on high-dimensional and dynamic service graphs is challenging. It suffers from high computational cost and unstable structures. Effective integration with representation learning is therefore required [6].

Against this background, jointly modeling graph representation learning and causal inference on microservice dependency graphs has significant research value. Graph representation learning can extract robust and structure-aware system state representations from complex dependencies. These representations provide compact and informative inputs for causal analysis [7]. Causal inference can then introduce directionality and mechanism constraints in the representation space. Root cause localization is no longer limited to surface associations. It can instead target true triggering sources. Such joint frameworks have the potential to unify structural modeling and causal explanation at the theoretical level. In practice, they can enhance observability, diagnosability, and autonomous operations. This direction offers more reliable technical support for the stable operation of large-scale microservice systems.

Recent backend anomaly studies also show that reliable root cause localization requires robust representation under sparse labels, historical context, and structured event relations. Prior-enhanced representation learning under weakly labeled and few-shot conditions demonstrates how auxiliary priors can stabilize backend anomaly recognition when labeled failures are limited [8]. Memory-enhanced transformer modeling further preserves long-range microservice state evolution for anomaly detection [9], while log semantic graph construction with convolutional neural networks captures semantic dependencies among system events [10]. Temporal context and entity-relationship anomaly modeling in audit risk identification and counterfactual time-series strategy simulation in cash-flow forecasting provide additional methodological support for modeling evolving entities, intervention scenarios, and risk propagation under incomplete observations [11], [12]. These directions strengthen the motivation for combining dependency graphs, history-aware representations, and causal reasoning to distinguish true triggering services from propagated symptoms.

2. Method

This method uses the dependency graph of a microservice system as the core modeling object, uniformly representing the system's operational state within a given time window as a directed graph that evolves. Nodes in the graph correspond to specific services or service instances, and edges depict the call and resource dependencies between services. For each node, an initial state vector is first constructed from multi-source observations such as logs, metrics, and traces, and the initial feature representation of the node is obtained through time alignment and normalization. Let the system dependency graph at time t be represented as $G_t = (V, E)$, where V is the set of service nodes, E is the set of dependency edges, and the initial state feature representation of the node $v_i \in E$ is:

$$x_i^{(t)} \in R^d$$

This approach compresses heterogeneous observations into a unified vector space, enabling the overall system state to be jointly modeled under structural constraints. In this way, the method avoids dependence on single modalities or local signals from the outset, instead characterizing operational behavior from a holistic system perspective, laying the foundation for subsequent structured inference. This paper also presents the overall model architecture, as shown in Figure 1.

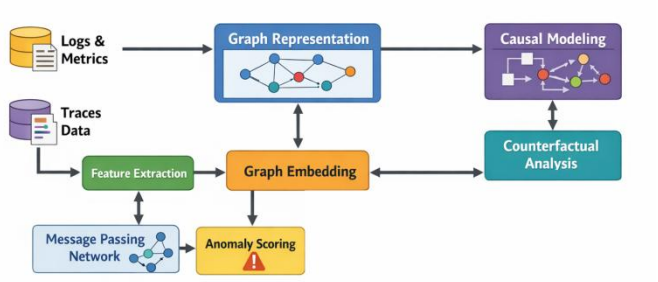

Figure 1. Overall model architecture

In the graph structure modeling phase, a message-passing-based graph representation learning mechanism is introduced to explicitly model service dependencies. During each layer update, the node representation is determined by its own state and the states of its directly dependent nodes, thus reflecting the characteristic that anomalies may propagate along dependency edges. Specifically, the node representation at layer $l+1$ is obtained through the following update rules:

$$h_i^{(l+1)} = \sigma\left(W_1 h_i^{(l)} + \sum_{j \in N(i)} W_2 h_j^{(l)}\right)$$

Where $h_i^{(0)} = x_i^{(t)}$, $N(i)$ represents the set of dependent neighbors of node i, W_1 and W_2 are learnable parameters, and $\sigma(\bullet)$ is a nonlinear mapping function. After multiple layers of propagation, the node representation can simultaneously encode local runtime state and dependency structure information, thus forming a structure-aware system state embedding.

Building upon the structured representation, causal modeling is introduced to characterize the directional influence relationships between services. The method assumes that the potential anomalous state of each service node is jointly determined by the states of its upstream dependent nodes, and follows a structured causal model. For node i, its potential anomalous variable z_i is defined as:

$$z_i = \sum_{j \in Pa(i)} a_{ji} z_j + \epsilon_i$$

Where $Pa(i)$ represents the set of parent nodes pointing to node i in the dependency graph, a_{ji} is the causal influence strength, and ϵ_i is the independent noise term. This modeling approach explicitly distinguishes between the causes of anomalies and their propagation results, allowing the dependency structure to not only exist as a topological constraint but also directly participate in characterizing the anomaly generation mechanism, thereby providing an interpretable causal path for root cause inference.

Ultimately, root cause localization is achieved through joint structural representation and causal effect inference. First, a node anomaly scoring function is constructed based on node embeddings and potential outlier variables:

$$s_i = \| h_i - \mu_i \|_2$$

Where μ_i represents the reference representation of the node in its normal state. Subsequently, the anomaly score is weighted by the causal influence strength to obtain the comprehensive root cause score:

$$r_i = s_i \cdot \sum_{k \in Ch(i)} |a_{ik}|$$

Here, $Ch(i)$ represents the set of downstream dependent nodes of node i. This score reflects both the degree of anomaly of the node itself and its potential impact on other components of the system, making the final localization result more inclined towards the real trigger source rather than passively affected downstream services, thereby achieving a unified approach to backend fault root cause localization that combines structural awareness with causal explanation.

3. Experimental Analysis

3.1 Dataset

This study uses the open-source TrainTicket microservice anomaly monitoring dataset as the data source. The dataset is released on Zenodo. It is designed for observable data collection and anomaly scenario recording in microservice architectures. It contains multi-source monitoring data from the TrainTicket benchmark microservice system. The data cover typical observation channels, including logs, Jaeger distributed traces, and Prometheus KPIs and metrics. The dataset is organized into multiple independent sub-packages. This design supports unified reading and alignment under different anomaly settings or system changes. It also provides descriptive information about anomaly phenomena. These properties satisfy the structural and semantic requirements for dependency graph modeling and root cause inference.

From the perspective of the research topic, the dataset is highly aligned with backend fault root cause localization on microservice dependency graphs. Trace data can be used to reconstruct service invocation topology and interaction direction. This process enables the construction of microservice dependency graphs, where nodes represent services or components and edges represent invocation dependencies or upstream-to-downstream request relations. Metrics and logs provide complementary information. Metrics describe continuous numerical states. Logs provide discrete event semantics. Together, they serve as multimodal node state inputs and support graph representation learning for modeling the global system state. In the causal inference stage, annotated or documented anomaly segments help identify candidate intervention variables and affected scopes. This setting allows root cause localization to move beyond correlation ranking toward directional effect explanation. It better reflects real fault propagation and cascading degradation mechanisms in backend systems.

In data preparation, three main processing steps are applied to support the joint framework. First, span relationships in traces are aggregated to the service level. This step derives a stable set of directed dependency edges and provides prior constraints for causal direction. Second, Prometheus metrics are aligned to a unified sampling interval. Missing values are handled, and features are normalized. Each service is then represented by a window-based state sequence. Third, logs are processed through templating or event counting. High-dimensional text events are compressed into representations that can be fused with metrics and traces. After these steps, the data can be uniformly mapped into a standard input form. This form consists of a graph structure, multiple source node states, and anomaly time slices. It directly supports the joint modeling of graph representation learning and causal inference.

3.2 Experimental Results

This article first presents the results of the comparative experiments, as shown in Table 1.

Table 1. Comparative experimental results

Method	Acc	Precision	F1-Score	AUC
Deeptralog [13]	0.846	0.832	0.818	0.871
TraceGra [14]	0.868	0.857	0.841	0.892
MADMM [15]	0.883	0.874	0.862	0.907
Uac-ad [16]	0.897	0.886	0.874	0.921
TraceDAE [17]	0.914	0.903	0.897	0.938
Ours	0.953	0.947	0.942	0.975

The comparative results show a clear progression from traditional representations to more structured modeling strategies. This trend indicates that, in microservice environments, relying only on local logs or a single observation signal is often insufficient to stably characterize fault evolution. As tracing information and cross-service

associations are introduced, the separability between abnormal and normal states improves significantly. This improvement demonstrates that dependency structures carried by cross-component interactions are essential cues for backend fault identification and root cause localization. This observation is consistent with the dependency graph assumption, where faults rarely appear as isolated point events but instead spread along invocation paths and manifest as coherent structural deviations across multiple observation sources.

From the relationships among evaluation metrics, the simultaneous improvement in accuracy and AUC suggests that the model becomes more stable in overall discrimination while maintaining strong ranking capability under varying thresholds. This property is critical for root cause localization. Such tasks rely on risk ranking and prioritization of candidate services rather than binary decisions at a fixed threshold. Enhanced ranking ability allows the model to concentrate anomaly signals on services closer to the true trigger within complex dependency chains. It reduces misjudgments caused by downstream symptoms. As a result, localization outcomes better align with the true direction of dependency propagation.

The joint improvement in precision and F1 indicates that both false positives and false negatives are effectively reduced. This pattern reflects a clearer characterization of anomaly boundaries. In microservice systems, many anomalies represent surface-level fluctuations after cascading propagation. These effects are often misidentified as root causes, which increases false alarms and reduces localization efficiency. Higher precision implies that the model can filter secondary anomalies induced by propagation and retain key evidence related to upstream triggers. The increase in F1 further shows that true anomalies are still captured while maintaining low false alarm rates. This balance prevents the loss of early signals related to root causes and aligns with the objectives of structural constraints and causal direction inference on dependency graphs.

Overall, the best performing method achieves consistent advantages across all four metrics. This consistency indicates that the learned representations reflect system-level coherence rather than accidental fitting to a specific modality or local pattern. For root cause localization on microservice dependency graphs, such consistency implies effective alignment of multiple source observations in a shared representation space. It also indicates that dependency structures act as channels for information propagation and evidence aggregation. As a result, the anomaly detection stage preserves interpretable structural cues and provides stable inputs for subsequent causal inference. These findings support the necessity of a joint framework. Structure-aware representation learning and directional reasoning on dependency graphs are both required to achieve reliable fault identification and root cause attribution in complex backend environments.

This sensitivity experiment was used to examine whether the model's ability to handle microservice dependency structure information remains stable when the graph embedding dimension changes. By repeating the same process under different dimension settings, the impact of the balance between representation capacity and structural inductive bias on discriminative ability can be observed. The experimental results are shown in Figure 2.

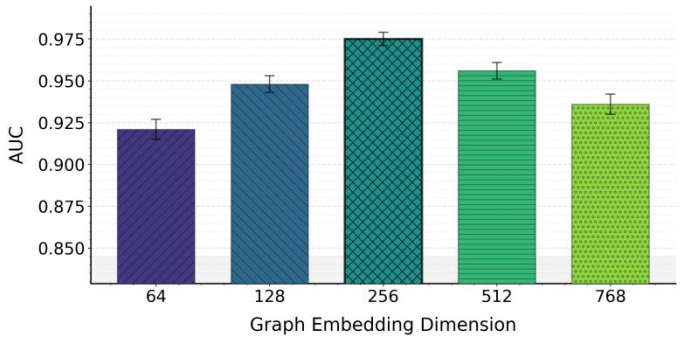

Figure 2. Sensitivity experiment of graph embedding dimension to AUC

The overall trend shows that the graph embedding dimension has a significant impact on model discriminative capability. This pattern reflects a clear trade-off between representation capacity and microservice dependency structure modeling. When the dimension is low, the embedding space cannot sufficiently encode cross-service invocation relations and state propagation information. System-level structural features are compressed. As a result, anomaly evolution patterns are not fully captured. As the embedding dimension increases, the model gradually gains stronger structural expression ability. State differences and propagation paths among services in the dependency graph become more distinguishable in the representation space. This improvement enhances the stability of overall backend anomaly discrimination.

When the embedding dimension is further enlarged, performance no longer improves and instead begins to decline. This behavior indicates that simply increasing the representation dimension cannot continuously strengthen model capability. Excessively high dimensions introduce redundant features. The representation space then contains more noise that is unrelated to fault propagation. This effect weakens the role of structural inductive bias in dependency graph modeling. In microservice environments, such redundancy can amplify local fluctuations or downstream symptoms. It reduces the model's focus when distinguishing root causes from propagation effects and degrades discriminative effectiveness.

From the perspective of dependency graph modeling, these results suggest that a moderate embedding dimension is more effective for capturing causal relations and state transmission patterns among services. An appropriate representation scale can encode key structural information required for cross-service interaction. At the same time, it naturally constrains noise and non-critical variations. This constraint allows the model to focus on anomaly diffusion patterns along dependency paths. Such structure awareness is essential for backend fault root cause localization. Root causes are usually reflected in a small number of critical nodes and their outward influence, rather than in weakly related features scattered across a high-dimensional space.

Overall, the graph embedding dimension is not better when it is larger. Its optimal range reflects a balance between representation power and structural constraints. This phenomenon further confirms the importance of structure awareness and inductive bias in modeling microservice dependency graphs. By controlling the embedding dimension, graph representation learning can better support causal inference objectives. This design provides more stable and interpretable support for root cause identification of complex faults in backend systems.

This sensitivity experiment was used to evaluate whether the model's characterization of the microservice's operational state remained stable when log observations were subjected to noise disturbances of varying intensities. By gradually increasing the noise injection ratio, the robustness boundary of the model to log uncertainty and event interference could be tested. The experimental results are shown in Figure 3.

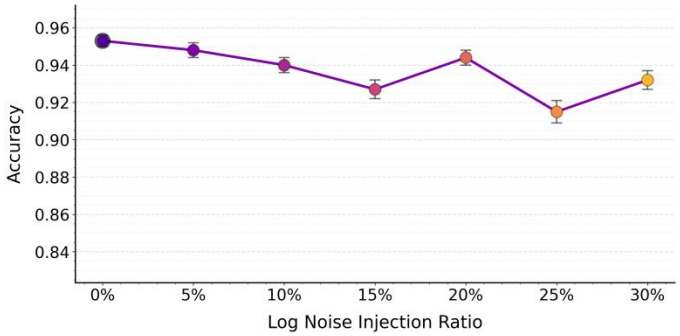

Figure 3. Sensitivity experiment of log noise injection ratio to Acc

As the proportion of injected log noise increases, the model discrimination of runtime states shifts from stable to more fluctuating behavior. This pattern reflects the sensitivity of logs as a key observation channel in the root cause localization pipeline. In microservice environments, log events carry rich local semantics and anomaly cues. Increased noise directly disrupts the identifiability of event patterns. This disruption makes it harder for the model to clearly separate anomaly sources from propagation effects on the dependency graph. Performance instability, therefore, emerges.

Within the low noise range, the curve remains relatively smooth. This behavior indicates that the model has a certain tolerance to mild log disturbances. It is consistent with the robustness introduced by structured modeling. Dependency graph representation learning aggregates and constrains local log perturbations through cross-service invocation structures. Node state representations do not rely on individual log events alone. Instead, neighborhood propagation and multi-source consistency help mitigate random noise. As a result, stable discrimination can be maintained under minor interference.

When noise intensity increases further, the curve shows more pronounced drops and rebounds. This change indicates that noise begins to dominate and damage critical event sequences. Early triggering signals related to root causes may be diluted or misaligned in logs. The model then becomes more likely to treat propagation symptoms as primary anomaly sources during node state construction. This effect influences directional inference on the dependency graph. The non-monotonic curve pattern also suggests that noise does not cause a simple performance decline. It alters the relative importance between observable semantics and structural propagation cues. The model thus exhibits different decision biases under different noise levels.

Overall, this sensitivity analysis shows that root cause localization on microservice dependency graphs requires more than structural representation learning. Stronger suppression and correction of uncertainty introduced by log noise are also needed. The curve fluctuations at medium and high noise levels indicate that degraded log quality significantly reduces the purity of root cause evidence. System-level reasoning then relies more heavily on structural priors and multimodal consistency constraints. Enhancing robustness to log perturbations within a unified framework can therefore help maintain stability and interpretability of localization results in complex backend environments.

4. Conclusion

This work addresses key challenges in backend fault localization for microservice architectures. These challenges include difficult localization, complex propagation paths, and heterogeneous multi-source observations. A joint modeling approach on microservice dependency graphs is proposed. The method starts from the overall system runtime structure to characterize fault generation and diffusion. Service invocations and dependencies are explicitly modeled. Structure-aware system state representations are then learned. This design moves beyond analyses that rely only on local signals or a single modality. Fault identification is no longer limited to surface symptoms. Abnormal behavior can be understood from a system-level perspective. This shift provides a more consistent and robust foundation for anomaly understanding in complex backend systems.

Causal inference is further introduced to move anomaly analysis from correlation description to a mechanism-level explanation. This addition improves the interpretability and credibility of root cause localization results. Directional influence relations among services are modeled under dependency graph constraints. The model can distinguish triggering sources from propagation effects. It avoids being misled by downstream anomaly signals in cascading failure scenarios. Comparative results show clear advantages in discriminative stability and overall consistency. These findings indicate that coordinated design of structural representation learning and causal modeling better reflects real system logic and anomaly evolution in microservice environments.

From an application perspective, the proposed method provides a scalable technical path for intelligent operations in large-scale distributed systems. In cloud platforms, financial backends, online services, and industrial internet systems, availability requirements are strict. System scale continues to grow, and service dependencies evolve rapidly. Manual inspection and rule-based approaches become insufficient. The proposed framework enables system-level understanding of anomalies under complex dependencies. It offers more targeted diagnostic cues for operators. This capability helps

reduce fault localization time and lowers business risks caused by service disruption. It has direct practical value for improving stable backend operation.

Looking ahead, this study lays a foundation for more autonomous and intelligent system operations. The combination of dependency graphs and causal modeling provides a structured basis for online learning and adaptive updates. Models can continuously adjust their understanding of normal and abnormal behavior as systems evolve. The framework also supports generalization across systems and scenarios. It encourages a shift from isolated tools toward system-level decision support for anomaly detection and root cause localization. As microservice architectures expand into more critical domains, this structure and mechanism-oriented joint modeling approach is expected to play an increasingly important role in software system reliability assurance.

References

[1] C. M. Lin, C. Chang, W. Y. Wang, et al., "Root cause analysis in microservice using neural Granger causal discovery," in Proc. AAAI Conf. Artificial Intelligence, vol. 38, no. 1, pp. 206-213, 2024.

[2] R. Xin, P. Chen, and Z. Zhao, "CausalRCA: Causal inference based precise fine-grained root cause localization for microservice applications," Journal of Systems and Software, vol. 203, Art. no. 111724, 2023.

[3] Y. Zhu, J. Wang, B. Li, et al., "MicroIRC: Instance-level root cause localization for microservice systems," Journal of Systems and Software, vol. 216, Art. no. 112145, 2024.

[4] C. Zhang, Z. Dong, X. Peng, et al., "Trace-based multi-dimensional root cause localization of performance issues in microservice systems," in Proc. IEEE/ACM 46th Int. Conf. Software Engineering, 2024, pp. 1-12.

[5] Z. Yao, C. Pei, W. Chen, et al., "Chain-of-event: Interpretable root cause analysis for microservices through automatically learning weighted event causal graph," in Companion Proc. 32nd ACM Int. Conf. Foundations of Software Engineering, 2024, pp. 50-61.

[6] L. Zheng, Z. Chen, J. He, et al., "MULAN: Multi-modal causal structure learning and root cause analysis for microservice systems," in Proc. ACM Web Conf. 2024, pp. 4107-4116.

[7] L. Zheng, Z. Chen, H. Chen, et al., "Online multi-modal root cause analysis," arXiv preprint arXiv:2410.10021, 2024.

[8] Y. Sun, "Prior Enhanced Representation Learning and Adaptive Recognition Method for Backend Anomaly Detection under Weakly Labeled and Few-Shot Conditions," 2024.

[9] Y. Yang, "Memory-Enhanced Transformer for Backend Microservice Anomaly Detection," 2024.

[10] Y. Yang and A. Xu, "Log Semantic Graph Construction and System Event Anomaly Detection via Convolutional Neural Networks," 2024.

[11] R. Yan and M. Guo, "Enterprise Audit Risk Identification Through Temporal Context and Entity Relationship Anomaly Modeling," 2024.

[12] Y. Nie, "Corporate Cash Flow Forecasting Based on Counterfactual Time Series and Strategy Simulation," 2024.

[13] C. Zhang, X. Peng, C. Sha, et al., "DeepTraLog: Trace-log combined microservice anomaly detection through graph-based deep learning," in Proc. 44th Int. Conf. Software Engineering, 2022, pp. 623-634.

[14] J. Chen, F. Liu, J. Jiang, et al., "TraceGra: A trace-based anomaly detection for microservice using graph deep learning," Computer Communications, vol. 204, pp. 109-117, 2023.

[15] P. Wang, X. Zhang, Z. Cao, et al., "MADMM: Microservice system anomaly detection via multi-modal data and multi-feature extraction," Neural Computing and Applications, vol. 36, no. 25, pp. 15739-15757, 2024.

[16] H. Liu, X. Huang, M. Jia, et al., "UAC-AD: Unsupervised adversarial contrastive learning for anomaly detection on multi-modal data in microservice systems," IEEE Transactions on Services Computing, vol. 17, no. 6, pp. 3887-3900, 2024.

[17] P. Liu, H. Xu, Q. Ouyang, et al., "Unsupervised Detection of Microservice Trace Anomalies through Service-Level Deep Bayesian Networks," in Proc. IEEE 31st International Symposium on Software Reliability Engineering (ISSRE), 2020, pp. 48 – 58.