International Journal of Cloud Applications and Computing
Volume 16 • Issue 1 • January-December 2026

A Graph Transformer-Based Framework for Multi-Modal Failure Diagnosis in Microservice Systems

Áron Kiss
https://orcid.org/0000-0003-4549-4107
Institute of Information Science, University of Miskolc, Miskolc-Egyetemváros, Hungary

Károly Nehéz
https://orcid.org/0000-0002-6953-3898
Institute of Information Science, University of Miskolc, Miskolc-Egyetemváros, Hungary
Received: October 6th, 2025 | **Accepted:** February 2nd, 2026

ABSTRACT

Failure diagnosis in microservice systems is difficult due to the complex, multimodal nature of telemetry data. Existing methods use metrics, logs, and traces, but rely on message-passing graph neural networks with limited ability to model global context. This study introduces TransTVDiag, which replaces TVDiag's GraphSAGE encoder with a Graph Transformer enhanced with structural encodings for microservice correlation graphs. The study provides four main contributions: (1) adapting Graphormer to multimodal alert graphs with degree centrality and shortest-path encodings, (2) analyzing these encodings in microservice diagnostics, (3) quantifying the individual and joint impact of metrics, logs, and traces, and (4) demonstrating robustness to missing or noisy alerts. TransTVDiag improves root cause localization in hit ratio by 5.3%, in ranking quality by 3.5%, while reducing inference time by 83.8% over TVDiag. The study also outlines how model outputs can be made more actionable for operators, showing that Graph Transformers offer an accurate and efficient alternative for multimodal failure diagnosis.

KEYWORDS

Failure Diagnosis, Root Cause Localization, Anomaly Detection, Microservice, Multi-Modal Analysis, Graph Encoding, Graph Neural Network, Transformer, Graphormer

INTRODUCTION

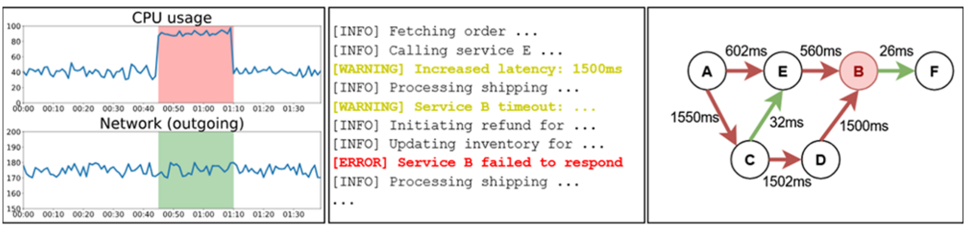
Over the past decade, there has been increasing interest in microservice-based systems within industry and academia (Colanzi et al., 2021). These systems are more reliable and scalable compared to applications built using classical monolithic architecture (Luo et al., 2021). However, their complexity and dynamism pose novel challenges in system maintenance (Guo et al., 2020). An anomaly in a specific microservice can propagate to other microservices, affecting a larger part or even the entire system. This makes identifying the type of failure and localizing the culprit instance particularly challenging while also potentially causing financial losses due to service outages or performance degradation, leading to reduced user satisfaction. Therefore, quickly investigating errors in such systems and initiating mitigation actions as soon as possible is critical.

DOI: 10.4018/IJCAC.402208

System administrators often monitor the health of such systems by collecting and analyzing status information such as logs, traces, and metrics. Software like OpenTelemetry (https://opentelemetry.io/), Jaeger (https://www.jaegertracing.io/), Prometheus (https://prometheus.io/), or Grafana Loki (https://grafana.com/oss/loki/), which handle and generate standardized data formats, can be used for this purpose. These data types represent different communication intents and therefore different modalities, as they provide insight into distinct aspects of the system's operation.

Figure 1 illustrates several modalities. Metrics (a) are quantifiable, time-varying data points (e.g. CPU, network, or memory usage) that characterize the aggregated performance of the system or one of its components (Dai et al., 2021). Logs (b) are sequences of semi-structured messages generated during system runtime. They provide information about the internal state of the system and its temporal changes (Kovács et al., 2024). Traces (c) are used to track transactions in distributed systems, showing how a request is handled across various system components and capturing attributes such as timestamps, response times, and status codes of individual calls (Zhou et al., 2021).

Figure 1. Metric, Log, and Trace Modalities of a Microservice System



(a) Metrics **(b) Logs** **(c) Traces**

Industrial fault localization in large-scale systems remains a predominantly manual process, resulting in significant time costs for engineers (Li et al., 2022). Automating fault detection could significantly decrease diagnosis times, especially for complex systems involving multiple teams. The automatic failure diagnosis of microservice systems has been of longstanding interest to researchers and industry professionals. Most traditional diagnostic methods rely on *single-modal* data, namely metrics (Kim et al, 2013; Pan et al., 2021), logs (Bansal et al., 2020; Zhang et al., 2021), or traces (Li et al., 2021; Zhou et al., 2019) to detect faults. However, this is often insufficient for precise analysis, as focusing on a single aspect of the system's state may not provide enough information to determine the type of failure or identify the culprit instance.

For example, suppose users of a video streaming service observe that videos are loading slowly or freezing at certain times. Diagnosing this problem can rely on multiple modalities. Metrics might reveal that one of the servers exhibits unusually high network latency, indicating potential network congestion but providing no further details on the root cause. The streaming server's logs may contain error messages indicating specific issues with video file access, such as corrupted data or improper permissions. Traces, however, could show that the video loading process consistently fails at a particular content delivery network (CDN) server, pinpointing the fault within the infrastructure. Analysing only metrics could result in misinterpreting the issue as a general network slowdown, leading to inaccurate conclusions. Logs can highlight specific events but fail to provide an overarching view of the process. By integrating traces, however, it becomes clear that the problem lies with one overloaded CDN server that cannot serve specific video files. This integration enables rapid localization and targeted mitigation of the fault.

Existing *multi-modal* failure diagnosis models generally perform better due to their ability to process, fuse, and learn patterns from multiple data types simultaneously (Lee et al., 2023; Tao et al.,