JAISCR, 2026, Vol. 16, No. 4, pp. 343 – 366
10.2478/jaiscr-2026-0017

GRAFLOW: A MICROSERVICE ANOMALY DETECTION METHOD BASED ON CROSS-MODAL FEATURE FUSION AND MULTI-SCALE GRAPH ATTENTION NETWORKS

Shuangshi Zhao[1], Kunming Liu[1], Jianlin Lu[2], Zhejie Xu[2], Meifang Yan[3], Keyuan Qiu[4,*]

[1]*Graduate School of Information and Communication, Ajou University, 206 World cup-ro, Yeongtong-gu, Suwon, Gyeonggi-do, Republic of Korea, 16499*

[2]*College of Information Science and Technology, Shihezi University, No. 221 Beisi Road, Shihezi, Xinjiang Uygur Autonomous Region, China, 832003*

[3]*School of Computer Science and Technology, Xinjiang Normal University, 102 Xinyi Road, Shayibak District, Ürümqi, Xinjiang Uygur Autonomous Region, China, 830054*

[4] *College of Information Science and Technology, Shihezi University, No. 221 Beisi Road, Shihezi, Xinjiang Uygur Autonomous Region, China, 832003*

[*]*E-mail: nightwish2024@163.com*

Submitted: 11th December 2025; Accepted: 11th May 2026

Abstract

With the widespread adoption of microservice architectures, software systems have become increasingly complex and dynamic, leading to more diverse types of anomalies. Traditional single-modal anomaly detection methods are no longer sufficient for accurately identifying system anomalies and locating their root causes. To address these challenges, this paper proposes GRAFlow, an anomaly detection and root cause localization method based on cross-modal feature fusion and multi-scale graph attention networks. GRAFlow integrates three key modalities—logs, performance metrics, and traces—and employs a cross-modal attention mechanism to dynamically model the semantic interactions among these heterogeneous data sources, thereby enhancing detection accuracy and robustness. Additionally, a multi-scale graph attention network is introduced to capture both local dependencies and long-range propagation paths among microservices, enabling more comprehensive system state modeling. The model jointly optimizes both anomaly detection and root cause localization tasks. Experimental results on two real-world microservice datasets, TrainTicket and SocialNetwork, demonstrate that GRAFlow significantly outperforms existing state-of-the-art methods across multiple evaluation metrics, including accuracy, F1 score, HR@K, and NDCG@K, confirming its effectiveness and robustness in complex system environments.

Keywords: anomaly detection, graph neural networks, Microservice systems, cross-modal fusion, root cause localization

1 Introduction

Microservice architecture decomposes an application system into a set of fine-grained, single-responsibility, and independently deployable services. With its excellent capabilities in dynamic scalability, flexible development, and rapid deployment, it has gradually become the mainstream architectural paradigm for cloud-native enterprises [1, 2], and has been widely applied in key industries such as internet finance, e-commerce, and social networking. Furthermore, with the rapid evolution of artificial intelligence, these distributed architectures are increasingly extending into mobile edge computing and the Internet of Things (IoT) [3], where system stability becomes even more critical. Persistent system anomalies, if left undetected, can expose platforms to severe vulnerabilities and cybercrimes, particularly in sensitive domains like e-banking [4] and smart transportation networks [5]. In large-scale cloud computing environments, tens of thousands of microservice instances are deployed in containerized form across distributed server clusters, continuously generating massive volumes of monitoring data. While microservice architecture significantly enhances system flexibility and maintainability, it also introduces greater complexity [6].

On one hand, the structural heterogeneity and semantic gaps across different types of monitoring data make it difficult for traditional approaches to achieve both efficient and accurate anomaly detection.[7] In such multi-source settings, anomalous patterns are often buried within normal fluctuations, making them prone to false alarms and missed detections. On the other hand, the intricate invocation relationships between microservices greatly complicate anomaly localization and root cause analysis. A single anomaly can propagate along the invocation chain, leading to cascading failures that pose serious challenges to accurate root cause tracing. For instance, in large-scale e-commerce platforms, a minor latency spike in a non-critical backend service can propagate upward, eventually causing critical timeouts in the order processing system and leading to significant economic losses. Consequently, how to leverage multi-source monitoring data to achieve efficient anomaly detection and precise root cause localization has become a crucial issue in ensuring system stability and improving operational efficiency [8].

During system operation, microservice environments are prone to diverse types of anomalies, including performance degradation, invocation latency, and functional failures. For instance, under high concurrency, resource contention among services may lead to delayed responses or request backlog, often manifested as abnormal CPU or memory usage. In systems with stringent real-time requirements, network jitter may induce increased delays in service invocation chains. Functional anomalies—typically caused by code defects, data inconsistencies, or failures in external dependencies—often result in request failures or frequent error logs [9].

To diagnose such anomalies, system operators typically rely on three major types of observability data: metrics, logs, and traces. Metrics are structured time series that record performance states of services and systems, useful for detecting resource fluctuations. Logs are semi-structured textual records of runtime events, rich in contextual information and critical for diagnosing functional faults. Traces represent the end-to-end execution paths of user requests in microservice systems, often in tree structures. Their atomic unit, the span, characterizes the behavior of a service instance for a particular request and reveals inter-service dependencies and execution patterns—making trace data well-suited for modeling invocation latency and anomaly propagation paths [9].

With the advent of multi-modal fusion techniques, research combining diverse monitoring data modalities has demonstrated improvements in anomaly detection accuracy and robustness.[10] However, existing multi-modal approaches often treat each modality in isolation and rely on simple feature concatenation, neglecting the potential semantic correlations among modalities [11, 12, 13]. In addition, some studies have leveraged graph neural networks (GNNs) to model the structural propagation of anomalies with promising results [14, 15, 16]. Nonetheless, conventional GNNs typically rely on static graphs, which limit their ability to capture dynamic or multi-scale anomaly propagation behaviors. In real-world microservice environments, anomalies may not only affect neighboring services but can also propagate across long distances along invocation chains, resulting in remote service disruptions.

Moreover, many existing methods [11, 13, 14, 17] treat anomaly detection as a binary classification task using linear or graph-based classifiers to determine whether a system is anomalous. These methods face several practical limitations. They often struggle to accurately detect localized anomalies. They lack effective strategies for identifying true root causes. And they are unable to model how anomalies propagate through the microservice graph or estimate the full scope of their impact on the system.

To address these challenges, this paper proposes GRAFlow, an anomaly detection model based on cross-modal feature fusion and multi-scale graph attention networks. The main contributions of this work are as follows:

1. GRAFlow deeply fuses three types of operational data, namely logs, call chains and performance metrics, and designs a cross-modal attention module for dynamically modelling the semantic associations and complementary relationships among different modalities. This mechanism can adaptively adjust the fusion weights of each modality according to the difference in the degree of reaction to the anomaly when the anomaly occurs, thus improving the sensitivity and discrimination ability of the model to complex anomaly patterns.

2. In order to effectively model local and remote dependencies in microservice systems, GRAFlow constructs a multiscale graph attention structure containing local and global paths. The local path captures the first-order direct dependencies between services, and the global path models the higher-order interactions and anomaly propagation links across services. After detecting a system anomaly, the model performs probabilistic inference based on the embedded features of each node in the graph structure, combined with the anomaly propagation paths, to achieve global awareness of the anomaly impact range and accurate localisation of the root cause service nodes.

3. The experiments of the model on two microservice datasets, TrainTicket and SocialNetwork, show that GRAFlow outperforms existing state-of-the-art methods in several evaluation indexes, such as precision, recall, F1 value, HR@K, and NDCG@K, and demonstrates excellent anomaly detection and root cause localisation, which verifies its validity and robustness in real-world scenarios. and robustness in real scenarios.

2 Related Works

2.1 Call Chain Based System Anomaly Detection

In modern distributed systems, ensuring comprehensive reliability is a multi-faceted challenge. Beyond anomaly detection, recent advancements have heavily relied on deep learning and cryptographic strategies to secure system interactions, such as deploying octave convolutional neural networks for offline signature authentication [18], utilizing cryptographic hashing for secure mobile computing [19], and engineering advanced graphical CAPTCHAs to prevent malicious online access [20]. Building upon these foundational reliability concepts, application operators typically focus on tracing internal behaviors.

In distributed systems, call chains can help application operators identify the root causes of anomalies in the system. Some scholars have tried to directly perform statistical analyses on call chains to identify potential root causes of anomalies. For example, Yu et al. [21] proposed the MicroRank method, which combines PageRank and spectral analysis to locate latency problems in microservice systems, while TraceAnomaly [22] directly analyses the response time in the call chain by using Variational Auto-Encoder (VAE) and Bayesian models to detect anomalies and locate the root causes.

Another class of approaches constructs and analyses service topology graphs to locate the root cause of performance anomalies. For example, Liu et al [23] proposed MicroHECL to dynamically construct service call graphs and rank candidate root cause services by correlation analysis, while Kim et al [24] proposed MonitorRank, an unsupervised algorithm that combines call graphs with time-series metrics to predict anomalous root causes in service architectures by means of a personalised PageRank algorithm. Similarly, MicroRCA [25] constructs an attribute graph that cor-

relates microservice call chain data with infrastructure metrics, employing graph-based sorting algorithms to pinpoint root causes. Zhang et al [14] used graphs to describe the interactions within a microservice system and the log events occurring in that architecture to detect anomalies in microservice logs.Xie et al [16] combined graph structure and node semantics to perform system anomaly detection using graph Transformer.Chen et al [15] accomplished server anomaly detection by using graph convolutional networks for different node subsequences of the server. anomaly detection, similarly (Huang et al [26], Morris et al [27], Xu et al [28, 29]).

Although such methods are mostly centred on call chaining, most of them still incorporate system performance metrics to enhance root cause analysis. However, in general, they can usually only locate root causes at the service level, and it is difficult to refine them to the level of more specific metrics.

2.1.1 System anomaly detection based on analysing system metrics

Some anomaly methods focus on using performance indicator data (e.g., CPU usage, memory occupation, network traffic, etc.) collected during system operation to locate the root cause of anomalies in a microservice system by analysing the time-series changes of key indicators. In parallel with these performance-centric diagnostic methods, advanced machine learning paradigms, such as federated learning, are increasingly being adopted to analyze network traffic for intrusion detection in distributed environments.[30] The advantages of this type of approach are fine granularity, ease of quantification, and the ability to capture the numerical manifestation of system anomalies at an early stage. For example, G-RCA [31] models the microservice system as a causal graph network and adopts graph traversal and edge scoring methods to identify metric variables that may trigger the propagation of anomalies, which is one of the earlier graph-structured metrics analysis methods.MicroDiag [32] constructs metrics causal propagation graphs, infers the root causes based on propagation paths between metrics, and ranks the metrics by using a random wandering approach. It breaks through the shallow connection method of traditional correlation analysis and can more effectively locate those root cause indicators located at the beginning of the propagation chain.ALERTRANK [33] scores and sorts the suspicious indicators by constructing the anomaly propagation topology graph, combining with the a priori correlation between indicators, and adopting PageRank or weighted centrality algorithm. This method is particularly effective when alarms are flooded and helps to quickly identify high-impact root causes from multiple simultaneous anomalies.

However, there is still an unavoidable problem with these performance metrics-based approaches, i.e., modern microservice systems often collect hundreds of metrics, and there are complex correlations and redundancies among different metrics, so how to quickly filter root cause-related metrics from them is still a difficult problem. Secondly, performance indicators are sometimes jittery in a short period of time which can produce false alarms, leading to a decrease in root cause localisation accuracy. Moreover, most of the methods still rely on statistical correlation as the basis for causal inference between indicators, but correlation is not equal to causality, and in practical applications, anomalous propagation chains often lead to correlation generalisation, which leads to misjudging the wrong cause.

2.1.2 System anomaly detection based on log analysis

In modern complex software systems, logs play an irreplaceable role in anomaly detection and fault diagnosis as the native record information during system operation. Compared with metrics and traces, logs have richer contextual semantics, event granularity and system behavioural descriptive capabilities. It not only records system state changes, but also often contains specific exception information, error reports and execution paths, which is especially crucial for understanding the internal behaviour of the system. Therefore, many scholars analyse logs to determine whether the system is normal or not.HU et al [34] proposed a log anomaly detection method LogADSBERT with high accuracy and strong generalization ability, which adopts the Sentence-BERT model to extract the semantic behavioural features of the log events and achieves the anomaly detection through a bi-directional recurrent neural network Bi-LSTM.Le et al [35] pro-

posed a method that does not require the Sentence-BERT model to detect the anomalies. [35] proposed NeuralLog, an anomaly detection method without log parsing, which extracts the semantic meanings of raw log messages and represents them as semantic vectors for anomaly detection via a classification model that captures contextual information from log sequences. In addition, with the wide application of Transformer model in the field of NLP, some scholars use Transformer model for anomaly detection, e.g., Xiao et al [36] proposed the Loader model based on Transformer model combined with variable fusion, which effectively fuses template and variable information in log data. And Bert model has more powerful feature extraction ability and language representation ability in the field of NLP compared to Transformer model, and achieved the highest level of results in 11 natural language processing tasks [37]. Therefore, many scholars use Bert model with other neural networks to achieve anomaly discrimination of logs, for example, Guo et al [38] proposed LogBert, a log anomaly detection model based on Bert, and Yu et al [39] proposed MFLAD, an anomaly detection method that adopts Bert in combination with Bi-LSTM, in addition to other approaches for log anomaly detection. Qiu et al. [40] generated pseudo anomaly data by learning known anomalies using a variant model based on Electra, which not only improves the identification of known anomalies, but also enhances the discovery of unknown anomalies, and effectively solves the challenges of data imbalance and limited labelling. Furthermore, extending anomaly detection to distributed IoT environments, literature [41] proposed Fedaware. This method utilizes a fractal shrinking autoencoder for intrusion detection, offering robust security monitoring for decentralized networks.

2.1.3 Multimodal based system anomaly detection

In recent years, anomaly detection methods based on multimodal data have emerged, aiming to enhance system state modeling by integrating heterogeneous data such as logs, metrics, and traces. SCWarn [42] employs LSTM to model the temporal dependencies between logs and metrics, but it does not explicitly account for semantic interactions across modalities. Hades [17] enhances system modeling by aligning embeddings of logs and metrics, yet its fusion strategy is fixed in structure and lacks adaptability. Eadro [13] concatenates multimodal features to construct service graph node embeddings. However, this static fusion strategy lacks adaptability. For instance, when an anomaly manifests exclusively in logs (e.g., a logic error) while performance metrics remain stable, Eadro's fixed concatenation may dilute the critical log signals with normal metric features, leading to false negatives. Similarly, DiagFusion [43] transforms different modalities into a unified textual representation to build a graph classification model. While unifying formats simplifies processing, it risks losing modality-specific semantics. Specifically, converting continuous high-frequency time-series metrics (e.g., CPU usage trends) into discrete text tokens discards precise numerical gradients and temporal patterns, making it difficult to detect subtle resource exhaustion anomalies. HeMiRCA [44] performs root cause localization by combining traces and metrics, using correlation mining for metric ranking, but its fusion strategy is static and does not consider coordinated modality shifts. AnoFusion [45] models multimodal data as temporal graph streams and detects anomalies based on prediction errors, yet lacks fine-grained modeling of cross-modal dependencies. Recently, Li et al. [46] proposed a self-supervised framework for learning spatio-temporal representations, enabling models to capture the dynamic evolution of microservices without relying on large amounts of labelled data.

In summary, there are two key limitations in the existing multimodal approaches: first, most of the modal fusion methods are based on static splicing or shallow alignment, which fail to capture the dynamic synergistic changes and semantic interactions between modes in anomalous states; second, the structural modelling generally relies on the shallow graph structure, which is not able to sense the remote propagation of the anomalies along the service dependency chain, which restricts the ability of the model to accurately depict the state of the complex system and locate the root cause in depth. The ability of the model to accurately portray the state of complex systems and locate the root causes in depth is limited.

Table 1. Summary of Key Notations

Symbol	Description
q_i, k_j, v_j	Query, Key, and Value vectors in cross-modal attention mechanism
α_{ij}	Attention weight between modality i and modality j in feature fusion
β_i	Attention coefficient for node i in global attention pooling
α	Hyperparameter balancing anomaly detection and root cause localization tasks
μ	Base intensity in the Hawkes process
α_{hawkes}	Self-excitation strength in the Hawkes process (denoted as α in Eq. 1)
β_{hawkes}	Temporal decay parameter in the Hawkes process (denoted as β in Eq. 1)
H^{fuse}	Fused representation of multi-modal features
h_{graph}	Graph-level representation for the final classification

3 Proposed Methodology

To enhance the readability of the mathematical formulations used in GRAFlow, we summarize the key symbols and their descriptions in Table 1.

3.1 Data preprocessing

In order to achieve efficient modelling of multi-source operational data in microservice systems, in this paper, four types of modal data in the original dataset - fault annotations, key performance indicators (KPIs), call chaining (Trace) and system logs (Log) - are unified preprocessing, as shown in Figure 1.

Failure annotations: standardise the time period of the failure, the service number, and the type of the failure, and unify the time-zone format and standardise the naming of the service, and finally convert it into a structured JSON format;

KPIs data: record the service performance indexes in a time series of per-second samples, and standardise it by filtering the dimensions that are highly correlated with the failure and save it as a CSV ;

Trace data: extract the start time, duration and service name of the span based on Jaeger, aggregate them into a service-level call delay sequence by timestamp, and save them in JSON format;

Log data: use Drain3 tool to perform structured template mining, convert the log text into a standardised sequence of events, extract the event timestamps and service identifiers, and save them in structured CSV format. Log data: using Drain3 tool for structured template mining, the log text is converted into standardised event sequences, event timestamps and service identifiers are extracted and saved in structured CSV format.

In addition, in order to unify the modelling inputs, this paper divides the pre-processed multi-source data into time chunks by 10 seconds and aligns the KPI, Trace and Log modal data. By comparing the overlapping relationship between each chunk and the time interval in the fault annotations, the chunks are labelled with whether they contain anomalies and the corresponding fault service nodes.

3.1.1 GRAFlow model

The GRAFlow model fuses three types of heterogeneous modal data, namely logs, performance metrics and call chains, to jointly model the semantic information and structural dependencies among microservices, so as to achieve in-depth perception of the system state and accurate identification of abnormal behaviours. As shown in Figure 2, GRAFlow consists of three main phases.

Multimodal Feature Learning: structured features are extracted for three types of modal data, such as event intensity vectors for logs, multivariate time series (MTS) for metrics, and service invocation delays for call chains, etc., and features are fused through the cross-modal attention mechanism to generate a unified representation of the services;

Multi-scale Dependency Sensing Modelling: a local and global two-channel graph attention network (GAT) is introduced to model the direct impact and cross-service dependencies of first-order and neighbouring services, respectively. Multi-scale dependency awareness modelling: introducing local and global dual-channel graph attention network (GAT), modelling the direct impact of

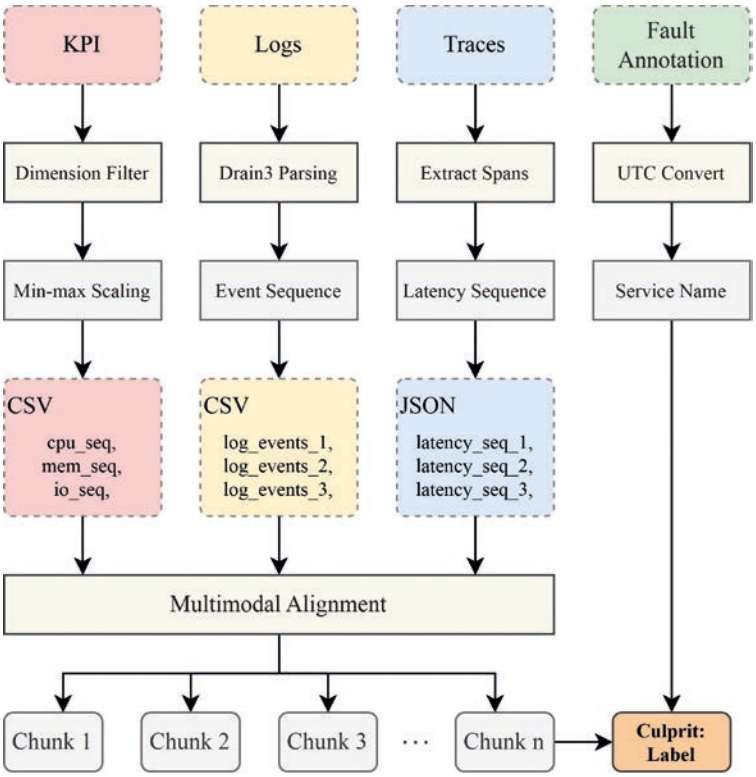


Figure 1. Schematic diagram of data preprocessing

neighbouring services and cross-service remote dependency respectively, and portraying the propagation characteristics of anomalies in the service dependency graph;

Anomaly Detection and Root Cause Localization: based on the fused representation of service nodes, optimizing the two tasks of anomaly detection and root cause localization simultaneously through a joint learning strategy, to achieve the identification of anomalous states and precise localization of the root cause service nodes. Root Cause Localisation.

3.1.2 Multimodal Feature Learning

In order to achieve deep fusion and modelling of multi-source operational data, this study designs corresponding feature extraction methods for three key modes, namely Log, Metrics and Trace, and uniformly projects them into a shared representation space for subsequent anomaly detection tasks.

Log events have significant dependence on the time dimension, and different types of events may show the causal relationship of 'event triggering event'. In order to deeply analyse the self-excitation behaviour of log events in the time dimension, this paper treats each type of log event as a point process, and introduces the HawkesADM4 algorithm based on maximum likelihood estimation to model it. From there, the base occurrence intensity of each type of event is obtained, indicating its activity level in the current time period. Although the approach assumes that inter-service events do not interfere with each other, i.e. it does not model co-excitation effects across services, subsequent multi-scale graph networks will compensate for this limitation by introducing structure propagation.

To ensure numerical stability, a small perturbation (on the order of 10^{-5}) is added to overlapping timestamps when constructing the event time sequence. A unified initial baseline of 0.2 is set for optimization initialization. The conditional intensity function $\lambda(t)$ of the model is defined as follows:

$$\lambda(t) = \mu + \sum_{t_i<t} \alpha \cdot e^{-\beta(t-t_i)} \tag{1}$$

Specifically, μ denotes the base intensity, α represents the self-excitation strength, β is the temporal decay parameter, and t_i indicates the timestamp of a historical event. The fitted intensities of all event types are concatenated to form a feature

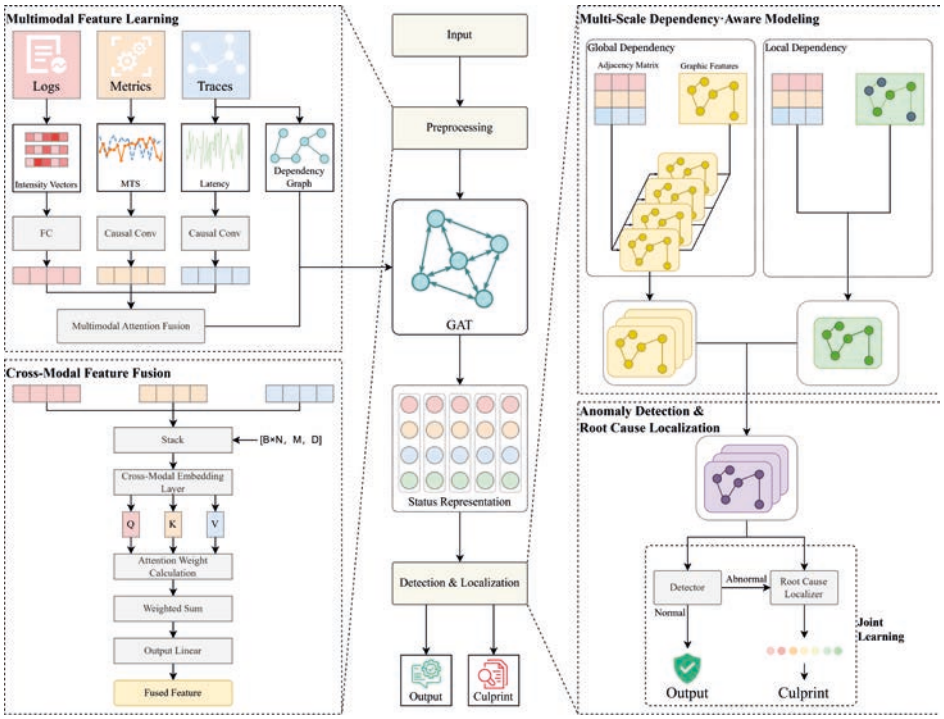


Figure 2. Schematic diagram of GRAFlow structure

vector, which is then projected into a shared latent space through a linear transformation, denoted as $\mathbf{V}_L$.

In addition, the *Metrics* data is treated as a high-frequency multivariate time series, including service-level indicators such as CPU utilization and memory usage. We first apply a Dilated Causal Convolution (DCC) network to extract temporal dependencies across different dimensions, followed by a self-attention mechanism to capture important cross-time dependencies. This process yields the context-aware representation $\mathbf{V}_K$.

Trace data records the invocation latency between services, reflecting explicit upstream and downstream logical dependencies. We construct time series from the invocation delay of each service, and process them through the same DCC network used for *Metrics*, further enhanced by a self-attention module. This results in the trace-based representation $\mathbf{V}_T$. Finally, the representations from all modalities are aligned through linear transformation and fused via a cross-modal attention module.

3.1.3 Multi-scale Dependency Sensing Modelling

In the anomaly detection task with multi-modal time-series data, accurately capturing the dependencies among modalities is crucial for identifying system status transitions. To effectively represent the structural interactions among microservices, we leverage GNNs as the core component for modeling microservice states, enabling dynamic dependency modeling across service nodes and providing a structural perspective of the system's status.

First, we take the previously extracted three modality-specific representations—log modality $\mathbf{V}_L$, trace modality $\mathbf{V}_T$, and metrics modality $\mathbf{V}_K$—and linearly project them into a shared latent space with dimension D, denoted as:

$$\mathbf{H}_L = W_{\log} \cdot \mathbf{V}_L, \quad \mathbf{H}_T = W_{\text{trace}} \cdot \mathbf{V}_T, \quad \mathbf{H}_K = W_{\text{metric}} \cdot \mathbf{V}_K \tag{2}$$

where $W_{\log}, W_{\text{trace}}, W_{\text{metric}}$ are learnable projection matrices corresponding to each modality, used to align them into a unified embedding dimension D.

Then, the three transformed modality features are fused using a cross-modal attention mechanism.

We stack them along the modality axis to obtain a 3D tensor:

$$\mathbf{X} = [\mathbf{H}_L; \mathbf{H}_T; \mathbf{H}_K] \in \mathbb{R}^{B \times M \times D} \tag{3}$$

where B denotes the batch size, $M = 3$ is the number of modalities, and D is the embedding dimension.

Next, based on each modality representation in the stacked tensor $\mathbf{X} \in \mathbb{R}^{B \times M \times D}$, we compute cross-modal attention. For the i-th sample, we extract the j-th modality feature vector $\mathbf{x}_j \in \mathbb{R}^D$ from $\mathbf{X}$, and use three learnable linear projections to compute the query, key, and value vectors:

$$\mathbf{q}_i = W^Q \cdot \mathbf{x}_i, \quad \mathbf{k}_j = W^K \cdot \mathbf{x}_j, \quad \mathbf{v}_j = W^V \cdot \mathbf{x}_j \tag{4}$$

where W^Q, W^K, and $W^V \in \mathbb{R}^{d \times D}$ are trainable projection matrices mapping input features into a shared attention space of dimension d.

The attention weight α_{ij} between the query vector of modality i and the key vector of modality j is computed using the scaled dot-product attention mechanism:

$$\alpha_{ij} = \frac{\exp\left(\mathbf{q}_i^\top \mathbf{k}_j / \sqrt{d}\right)}{\sum_{l=1}^{M} \exp\left(\mathbf{q}_i^\top \mathbf{k}_l / \sqrt{d}\right)} \tag{5}$$

Finally, the fused representation for the i-th sample is obtained by taking a weighted sum over all modality-specific value vectors and projecting the result via a fully connected layer:

$$\mathbf{H}^{\text{fuse}} = \text{FC}\left(\sum_{j=1}^{M} \alpha_{ij} \cdot \mathbf{v}_j\right) \tag{6}$$

where $\text{FC}(\cdot)$ denotes a linear transformation layer that outputs the final cross-modal representation for each sample.

Based on the constructed service dependency graph, we adopt a dual-path Graph Attention Network (GAT) structure to model multi-scale dependency relationships between service nodes. The fused feature $\mathbf{H}^{\text{fuse}}$ obtained from the previous module is separately fed into two attention graphs: a local graph $\mathcal{G}_{\text{local}}$ and a global graph $\mathcal{G}_{\text{global}}$, which are used to capture first-order structural dependencies and long-range indirect dependencies, respectively. Specifically, we establish the topological distance (i.e., hop count) as the classification criterion for these two scopes. The Local Graph focuses on direct, 1-hop neighbor interactions to identify immediate service faults, whereas the Global Graph utilizes multi-layer stacking to expand the receptive field, covering multi-hop indirect dependencies to detect cascading anomalies across the entire system.

For the local path, a shallow single-layer GAT is applied to perform one-hop message passing over the local graph:

$$\mathbf{H}_{\text{local}} = \text{GAT}_{\text{local}}(\mathcal{G}, \mathbf{H}^{\text{fuse}}) \tag{7}$$

For the global path, a deeper multi-layer GAT is employed to propagate information over several hops. The output of the l-th layer is defined as:

$$\mathbf{H}_{\text{global}}^{(l)} = \text{GAT}^{(l)}(\mathcal{G}, \mathbf{H}_{\text{global}}^{(l-1)}), \quad l = 1, 2, \ldots, L \tag{8}$$

The global propagation starts with the initial input:

$$\mathbf{H}_{\text{global}}^{(0)} = \mathbf{H}^{\text{fuse}} \tag{9}$$

The final output of the global GAT layers is:

$$\mathbf{H}_{\text{global}} = \mathbf{H}_{\text{global}}^{(L)} \tag{10}$$

To fuse local and global features, we concatenate the corresponding local and global embeddings for each node, and apply a shared linear projection to obtain the final node representation:

$$\mathbf{h}_i = \text{FC}_{\text{fuse}}\left(\left[\mathbf{H}_{\text{local},i} \| \mathbf{H}_{\text{global},i}\right]\right) \tag{11}$$

where $\|$ denotes the concatenation operator, and $\text{FC}_{\text{fuse}}(\cdot)$ is a fully connected layer.

After obtaining the final node embedding $\mathbf{h}_i$, we introduce a global attention pooling mechanism to compute a graph-level representation for downstream tasks.

To aggregate node embeddings into a graph-level representation, we apply a global attention pooling mechanism. For each node i, we compute an attention coefficient β_i as:

$$\beta_i = \frac{\exp\left(\mathbf{w}^\top \tanh(\mathbf{h}_i)\right)}{\sum_{j=1}^{N} \exp\left(\mathbf{w}^\top \tanh(\mathbf{h}_j)\right)} \tag{12}$$

where $\mathbf{h}_i \in \mathbb{R}^d$ denotes the embedding of the i-th node, $\mathbf{w} \in \mathbb{R}^d$ is a learnable context vector, and N is the total number of nodes in the graph.

The final graph-level representation is computed as a weighted sum over all node embeddings:

$$\mathbf{h}_{\text{graph}} = \sum_{i=1}^{N} \beta_i \cdot \mathbf{h}_i \tag{13}$$

This aggregated representation $\mathbf{h}_{\text{graph}}$ serves as the graph-level embedding and is subsequently fed into the downstream modules for anomaly detection and root cause localization.

3.1.4 Anomaly Detection and Root Cause Localization

Anomaly detection and root cause localization are the core tasks of our model. The model first determines whether the system is in an abnormal state. If an anomaly is detected, the model proceeds to locate the root cause, identifying the specific microservice responsible for the anomaly.

During the anomaly detection phase, we perform a binary classification based on the graph-level representation obtained from the GNN. A sigmoid-activated linear classifier outputs the predicted anomaly probability p_d for each graph. The prediction is compared against the ground truth label $y_d \in \{0,1\}$, where $y_d = 1$ indicates an abnormal system state, and $y_d = 0$ indicates normal.

The predicted probability is computed as:

$$p_d = \sigma\left(\mathbf{W}_d \cdot \mathbf{h}_{\text{graph}} + b_d\right) \tag{14}$$

The anomaly detection loss is calculated using the binary cross-entropy loss function:

$$\mathcal{L}_d = -\left(y_d \cdot \log(p_d) + (1 - y_d) \cdot \log(1 - p_d)\right) \tag{15}$$

Here, p_d denotes the predicted anomaly probability, y_d is the ground truth label, $\sigma(\cdot)$ is the sigmoid function, and $\mathbf{W}_d$, b_d are the classifier's weight and bias parameters.

The goal of root cause localization is to identify the microservice(s) responsible for the detected anomaly. Suppose the model outputs a probability distribution over all microservices, where $p_l[i]$ denotes the probability that the i-th microservice is the root cause of the anomaly.

Let y_l denote the index of the true root cause (i.e., the actual faulty microservice). If there is no root cause in a sample (i.e., the system is normal), we set $y_l = -1$ and exclude the sample from the localization loss computation.

The loss for root cause localization is computed using the cross-entropy function:

$$\mathcal{L}_l = -\sum_{i=1}^{N} \mathbb{I}(y_l = i) \cdot \log(p_l[i]) \tag{16}$$

where N is the total number of microservices, and $\mathbb{I}(y_l = i)$ is an indicator function that returns 1 if i is the ground truth root cause, and 0 otherwise.

Since anomaly detection and root cause localization are two closely related tasks, we adopt a joint optimization strategy by combining their respective loss functions. The overall loss is defined as:

$$\mathcal{L}_{\text{total}} = \alpha \cdot \mathcal{L}_d + (1 - \alpha) \cdot \mathcal{L}_l \tag{17}$$

Here, $\alpha \in [0,1]$ is a hyperparameter that balances the contributions of the two tasks. It can be tuned depending on the relative importance or difficulty of the tasks. The joint loss allows the model to simultaneously optimize for both anomaly detection and root cause localization, thereby improving the overall performance.

4 Experiments

4.1 Experimental Datasets

To evaluate the performance of the proposed model, we conduct experiments on two widely-used open-source microservice system datasets: TrainTicket and SocialNetwork. The basic statistics of the datasets are summarized in Table 2.

TrainTicket (TT)[47] is collected from the TrainTicket benchmark system, where users can search, book, and pay for train tickets. This dataset

involves interactions among 41 microservice instances, and collects multi-modal monitoring data for 27 business-related services. Three types of typical anomalies (CPU exhaustion, network latency, and network packet loss) are injected into the system. The average fault duration for each microservice node is 10 minutes. TT is released under the Apache License 2.0.

SocialNetwork (SN)[48] is collected from the SocialNetwork benchmark system, where users can create, read, like, and forward posts. It contains interactions among 21 microservice instances, and includes multi-modal data for 14 business-related services. Similar to TrainTicket, three types of anomalies are injected per service node, with an average fault duration of 2 minutes. SN is distributed under the GNU General Public License v2.0 (GPLv2).

All data within the dataset has been anonymised, with sensitive information removed, which aligns with the strict contemporary requirements for personal privacy evaluation in smart applications [49] and the growing imperative for preserving data confidentiality in high-security environments [50].

4.2 Environment Settings

The hardware devices and virtual environment used in this paper are as shown in the Table 3 below.

4.3 Evaluation Indicators

To comprehensively evaluate the performance of GRAFlow in anomaly detection tasks, this paper introduces a variety of metrics, including Precision, Recall, Area Under Curve (AUC), F1 Score, HR@K, and NDCG@K.

$$\text{Precision} = \frac{TP}{TP+FP} \tag{18}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{19}$$

$$\text{AUC} = \int_0^1 \frac{TP}{TP+FN}\, d\left(\frac{FP}{FP+TN}\right) \tag{20}$$

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{21}$$

$$\text{HR@K} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(s_{t_i} \in S_{p_i}[1:k]) \tag{22}$$

$$\text{NDCG@K} = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=1}^{M} \frac{p_j}{\log_2(j+1)}\right) \tag{23}$$

Here, TP(True Positive) denotes the number of samples that are actually positive and predicted as positive by the model. TN (True Negative) represents the number of samples that are actually negative and also predicted as negative. FP (False Positive) refers to the number of samples that are actually negative but incorrectly predicted as positive. FN (False Negative) denotes the number of samples that are actually positive but predicted as negative.

s_{t_i} is the ground truth root cause for the i-th test instance, and $S_{p_i}[1:k]$ is the top-k predicted candidate set for that instance. $\mathbb{I}(\cdot)$ is the indicator function, which equals 1 if the condition is true and 0 otherwise. p_j denotes the predicted probability score for the j-th microservice, M is the total number of microservices, and N is the number of test instances.

4.4 Results and Analyses

The results of the comparison experiments are shown in Table 4, and the experimental data are obtained from the results of the literature [9, 13] as well as the experimental reproduction results of the researcher. Eadro achieves F1 values of 0.989 and 0.986 on the two datasets respectively, showing strong detection ability, while GRAFlow achieves the best performance on SN with an F1 score of 0.992, showing higher detection robustness. In terms of the balance between Precision and Recall, the overall performance of GRAFlow is stable and better than Eadro in some scenarios, while the performance of GRAFlow is slightly worse than Eadro in the TT dataset, which is mainly due to the fact that the complex modal fusion and graph modelling structure of GRAFlow is more sensitive to trace signals, which may lead to feature perturbation in unimodal dominated scenarios. In systems such as SN, where anomalies propagate widely and modal synergy is more significant, GRAFlow shows stronger ability to capture anomalies and differentiate root causes by relying on multi-scale

Table 2. Basic statistics of the datasets

Dataset	Microservice	Modality	Record	Abnormal Ratio
TT	27	KPI	1,570,266	9.30%
		Log	108,637	
		Trace	148,296	
SN	14	KPI	101,646	7.80%
		Log	340,797	
		Trace	126,384	

Table 3. Experimental Environment Configuration

Experimental Environment	Configuration
System	Linux
GPU	RTX 2080×2
CPU	15 vCPU Xeon Platinum 8362
Pytorch	1.11.0
Python	3.8
Cuda	11.3

Table 4. Model performance comparison on the SN and TT datasets

Model	TT			SN		
	Precision	Recall	F1	Precision	Recall	F1
TraceAnomaly	0.589	0.414	0.486	0.636	0.468	0.539
MultimodalTrace	0.644	0.576	0.608	0.726	0.632	0.676
MS-RF-AD	0.971	0.705	0.817	0.700	0.866	0.773
MS-SVM-AD	0.938	0.678	0.787	0.808	0.770	0.789
MS-LSTM	0.940	0.887	0.967	0.937	0.959	0.948
MS-DCC	0.938	0.993	0.965	0.934	0.962	0.948
LogAnomaly	0.374	0.827	0.515	0.393	0.682	0.499
SCWarn	0.467	0.737	0.572	0.688	0.835	0.754
AnoFusion	0.702	0.859	0.773	0.749	0.890	0.813
STEAD	0.774	0.897	0.831	0.827	0.879	0.852
DySAT	0.925	0.942	0.933	0.918	0.935	0.926
EvolveGCN	0.910	0.925	0.917	0.895	0.912	0.903
Eadro	**0.984**	**0.995**	**0.989**	0.977	**0.996**	0.986
Ours	0.970	0.978	0.974	**0.990**	0.995	**0.993**

Table 5. Performance comparison of different models on the SN and TT datasets in terms of HR@K.

Model	TT			SN		
	HR@1	HR@3	HR@5	HR@1	HR@3	HR@5
TBAC	0.037	0.111	0.185	0.001	0.085	0.181
NetMedic	0.094	0.257	0.425	0.069	0.187	0.373
MonitorRank	0.086	0.199	0.331	0.068	0.118	0.221
CloudRanger	0.101	0.306	0.509	0.122	0.382	0.629
DyCause	0.231	0.615	0.808	0.273	0.636	0.727
MS-RF-RCL	0.637	0.922	0.970	0.704	0.908	0.970
MS-SVM-RCL	0.541	0.908	0.944	0.614	0.838	0.955
MS-LSTM	0.756	0.930	0.969	0.757	0.884	0.907
MS-DCC	0.767	0.938	0.972	0.789	0.968	0.985
HeMiRCA (Service Level)	0.827	0.889	0.901	0.713	0.784	0.874
HeMiRCA (Metric Level)	0.740	0.864	0.926	0.689	0.832	0.916
DySAT	0.815	0.920	0.965	0.795	0.915	0.950
EvolveGCN	0.798	0.905	0.955	0.776	0.890	0.935
Eadro	0.857	0.857	0.857	0.826	0.829	0.829
Ours	**0.986**	**0.996**	**0.996**	**0.970**	**0.988**	**0.990**

graph modelling and cross-modal fusion. In contrast, Eadro's lightweight gated fusion is more robust in this scenario.

Table 5 demonstrates the root cause localisation (HR@K) results of various models on the TT and SN datasets, where the data are taken from the literature [13, 44]. As can be seen from Table 5, deep learning models such as MS-DCC and MS-LSTM perform well overall in the root cause localisation task, especially in the HR@1 metric, where they clearly lead. In contrast, traditional methods such as TBAC and NetMedic are less effective, with HR@1 of 0.037 and 0.094 on the TT dataset, while the HR@1 of GRAFlow is 0.986 and 0.970 on the TT and SN datasets, which is significantly better than that of other methods, and shows its excellent performance in the root cause localisation task.

Table 6 demonstrates the NDCG and NGDCG metrics of different models on the TT and SN datasets, where the data are taken from the literature [13], and the combined evaluation metrics show that GRAFlow performs the best in the root cause localisation task. In particular, the performance on the TT dataset and SN dataset far exceeds that of the other models. In contrast, traditional models such as TBAC and NetMedic scored lower on all metrics.

The results in Table 7 show that GRAFlow outperforms Eadro on NDCG@1/3/5 for both TT and SN datasets. Although GRAFlow's overall F1 on the TT dataset is slightly lower than that of the best model, it performs better in root cause localisation thanks to the model's explicit modelling of the anomalous propagation paths and the precision of the two-channel graph attention structure in capturing the local and global dependencies. Eadro, on the other hand, is robust in detection but relatively limited in localisation accuracy in complex structures.

Figure 3 shows the PR and ROC curves of GRAFlow on the TT and SN datasets. The model performs well on both, with PR-AUC of 0.997 and ROC-AUC of 0.983 on TT, showing extremely low false alarm and omission rates; and PR-AUC and ROC-AUC of 0.995 and 0.982 on SN, respectively, which also reflect robust anomaly detection capability and good classification performance.

The confusion matrix of the GRAFlow model in Figure 4 shows that GRAFlow has good ability to identify abnormalities in both TT and SN datasets.

Table 6. Model performance comparison on the SN and TT datasets in terms of NDCG@3, NDCG@5, and NGDCG@5.

Model	TT		SN	
	NDCG@3	NDCG@5	NDCG@3	NGDCG@5
TBAC	0.079	0.109	0.048	0.087
NetMedic	0.195	0.209	0.146	0.218
MonitorRank	0.142	0.196	0.095	0.137
CloudRanger	0.218	0.301	0.269	0.370
DyCause	0.448	0.607	0.301	0.353
MS-RF-RCL	0.807	0.827	0.825	0.851
MS-SVM-RCL	0.814	0.820	0.741	0.790
MS-LSTM	0.859	0.877	0.834	0.844
DySAT	0.915	0.928	0.892	0.905
EvolveGCN	0.895	0.908	0.875	0.886
Ours	**0.992**	**0.992**	**0.984**	**0.984**

Table 7. Performance comparison of models on the SN and TT datasets in terms of NDCG@K

Model	TT			SN		
	NDCG@1	NDCG@3	NDCG@5	NDCG@1	NDCG@3	NDCG@5
Eadro	0.8571	0.8573	0.8575	0.8241	0.8249	0.8249
Ours	**0.9855**	**0.9918**	**0.9920**	**0.9776**	**0.9844**	**0.9844**

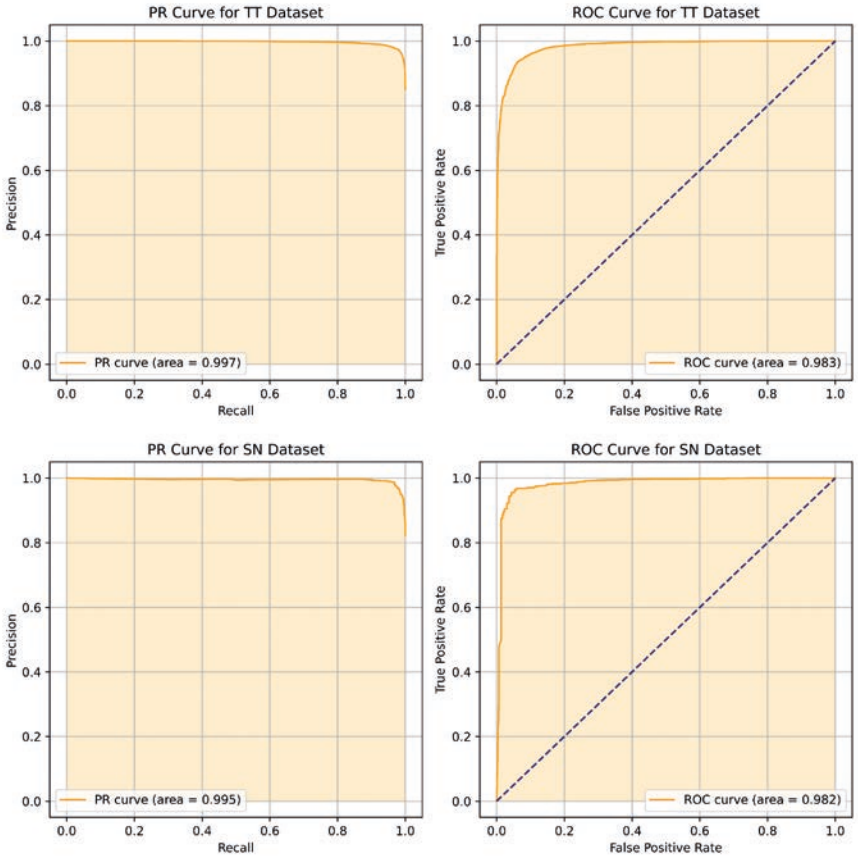


Figure 3. GRAFlow model PR curve and ROC curve changes

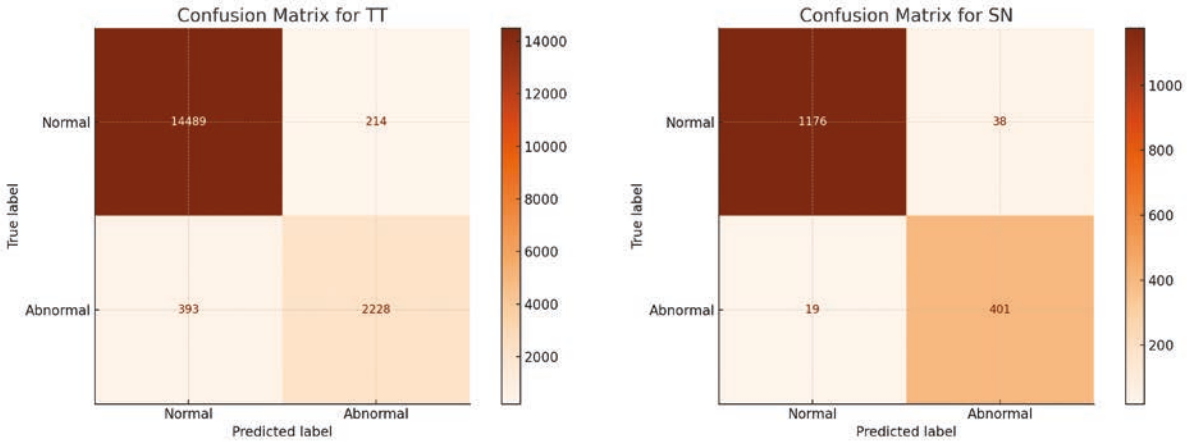


Figure 4. Confusion matrix of GRAFlow models on the dataset

In the TT dataset, the model accurately identifies most of the normal and abnormal samples despite some misclassification, while in the SN dataset, the model also demonstrates the accuracy and robustness of the identification. Confusion matrix on the dataset.

Figure 5 illustrates the T-SNE downscaling visualisation of the spatial features of the microservices in the test set. The graph on the left side shows the initial feature distribution, and the right side shows the final feature distribution of the test set. In the initial feature space, the distribution of microservice categories is rather cluttered, and many different categories of microservices overlap, and after training, the categories of microservices show obvious clustering and separation effects. Different categories of microservices are assigned to different regions, and there is a good spatial separation between the categories, indicating that GRAFlow has a good effect of differentiating and detecting different microservices.

In this paper, a set of normal microservices and an anomalous microservice are selected from each dataset respectively to extract normal and anomalous fragments for comparative analysis. Figure 6 demonstrates the model's ability to portray anomalous microservices on both datasets with respect to three dimensions: call chain, performance metrics, and anomaly score. Although there is a certain degree of response lag, the overall results show that the model is able to keenly perceive abnormal states and accurately identify abnormal service nodes.

4.5 Ablation Experiments

4.5.1 Module Validity Analysis

In order to verify the effectiveness of the Cross-modality Attention Module and the Multi-scale Graph Attention Network in the anomaly detection task, this paper designs multiple sets of ablation experiments. Specifically, ablation can be analysed in the following ways:

GRAFlow w/o Cross-modality: replace the Cross-modality fusion module with a linear layer, and retain only the Multi-scale GAT network.

GRAFlow w/o Multi-scale GAT: replaces the Multi-scale GAT network with a single GAT network, keeping only the Cross-modality module.

As can be seen from Table 8, the F1 values of the full model are 0.974 and 0.993 on the TT and SN datasets, respectively, which are both better than the two ablation versions. When the Cross-modality module is removed, the model performance decreases most significantly, with the F1 on SN dropping to 0.964, indicating that synergy between multimodal features plays a key role in identifying complex anomalies. Logs, metrics and call chains provide semantic, performance and structural perspectives, respectively, and it is difficult to depict the cross-modality performance and propagation paths of anomalies without any of the modalities. After removing the multi-scale GAT, the model F1 value also decreases significantly (TT: 0.965, SN: 0.987), suggesting that the joint modelling of local and global dependencies plays an important role in capturing anomaly propagation features. The local path focuses on modelling the first-order service dependency, while the global path senses the remote influence across modules, and both of them complement each other to improve the system structure characterization ability.

In order to assess the impact of each module on the root cause localisation effect, HR@K (K=1, 3, 5) was used as an indicator to carry out ablation experiments in this paper. As shown in Table 9, the complete model achieved the best performance on both TT and SN datasets. With the removal of the Cross-modality module, HR@1 decreased from 0.986 to 0.841 for TT and from 0.841 to 0.776 for SN, suggesting that modelling inter-modality interactions is crucial for improving root cause identification accuracy.

In contrast, the effect of removing the multi-scale graph attention module is relatively small but still statistically significant. Since anomalies often propagate along dependency chains and root cause nodes often do not have the strongest anomaly manifestations, it is difficult to effectively trace back the true source of anomalies using single structure modelling. Multi-scale graph modelling enhances the perception of anomaly propagation paths by jointly capturing local and remote dependencies, verifying its key role in improving the localisation accuracy under complex structures.

In addition, this paper further tests the NDCG@K (K=1,3,5) metrics of each model. As shown in Table 10, after removing the Cross-

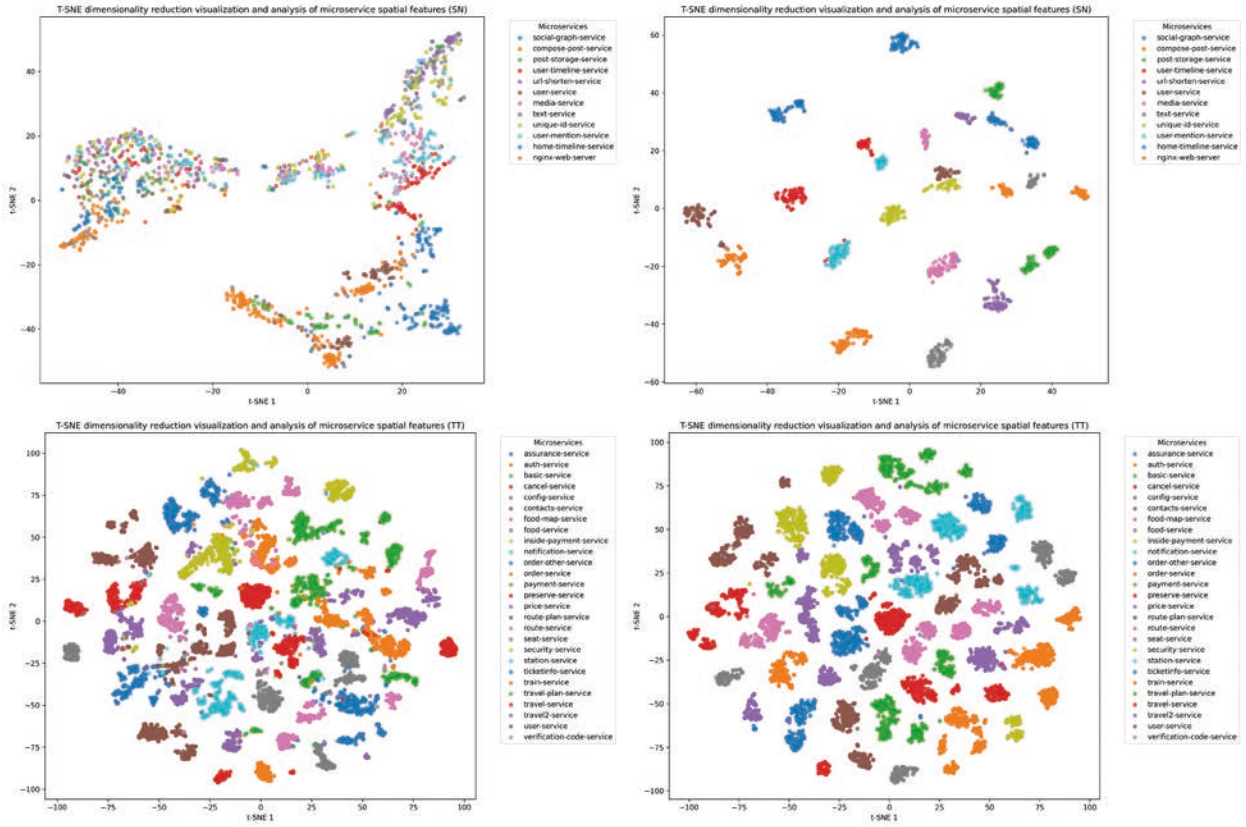


Figure 5. Spatial distribution of T-SNE features of the model on TT dataset and SN dataset

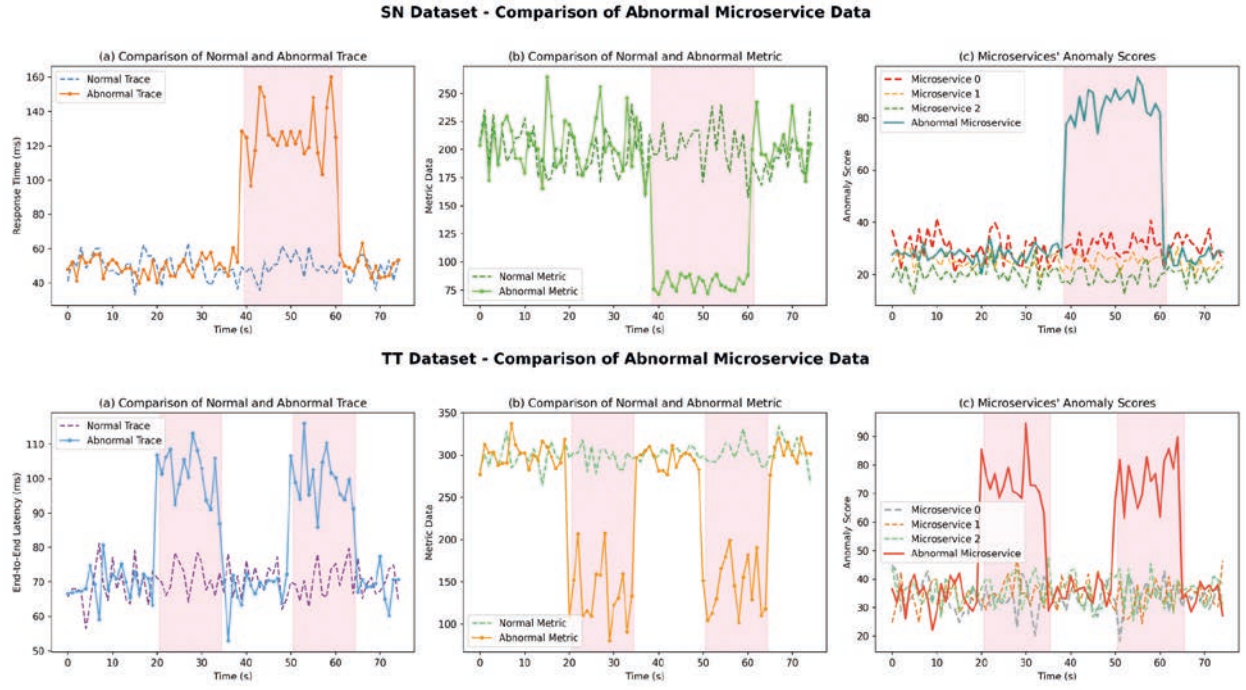


Figure 6. Comparison of Normal and Abnormal Microservices on Trace, Metric and Abnormal Score Dimensions

Table 8. Ablation results of the GRAFlow model on the anomaly detection task.

Model Variant	TT			SN		
	Precision	Recall	F1	Precision	Recall	F1
Ours w/o Cross-modality	0.952	0.967	0.959	0.968	0.960	0.964
Ours w/o Multi-scale GAT	0.963	0.967	0.965	0.983	0.993	0.987
Ours	**0.970**	**0.978**	**0.974**	**0.990**	**0.995**	**0.993**

Table 9. Ablation results of the GRAFlow model on the root cause localization task in terms of HR@K.

Model Variant	TT			SN		
	HR@1	HR@3	HR@5	HR@1	HR@3	HR@5
Ours w/o Cross-modality	0.841	0.844	0.844	0.776	0.816	0.816
Ours w/o Multi-scale GAT	0.867	0.867	0.868	0.772	0.835	0.840
Ours	**0.986**	**0.996**	**0.996**	**0.970**	**0.988**	**0.990**

Table 10. Ablation results of the GRAFlow model on the root cause localization task in terms of NDCG@K.

Model Variant	TT			SN		
	NDCG@1	NDCG@3	NDCG@5	NDCG@1	NDCG@3	NDCG@5
Ours w/o Cross-modality	0.841	0.843	0.843	0.876	0.929	0.931
Ours w/o Multi-scale GAT	0.867	0.867	0.868	0.904	0.916	0.917
Ours	**0.986**	**0.992**	**0.992**	**0.970**	**0.981**	**0.982**

modality fusion module, the NDCG@1 of the model on TT and SN decreases to 0.841 and 0.776, respectively, indicating that the lack of inter-modality interactions will lead to the difficulty of the model to fully comprehend the associations between anomalous semantics, temporal features, and the calling structure, which affects the model's ranking judgement on the importance of root cause nodes. And after removing the multi-scale graph modelling module, the model's ability to model dependencies in the full graph structure is weakened, and it cannot effectively distinguish between source nodes and affected nodes on the anomaly propagation path, resulting in the root-cause service being ranked lower in the ranking, and the NDCG@K scores are significantly reduced.

Figure 7 shows the t-SNE visualisation results of the distribution of service node features for different models on the TT and SN datasets. The first column is the GRAFlow model, the second column is GRAFlow w/o Cross-modality, and the third column is GRAFlow w/o Multi-scale GAT. it can be clearly seen that the node embeddings generated by the GRAFlow model present a clear clustering structure in the feature space with distinct category boundaries. In contrast, removing the cross-modal attention mechanism results in a severe mixing of embeddings from different service categories and blurring of inter-category boundaries, leading to a decrease in the model's discriminative ability. In contrast, removing the multiscale graph attention network has some discriminative ability but the cluster structure is loose and the intra-class tightness decreases, suggesting that the lack of structure-awareness ability weakens the model's ability to model service dependencies.

4.5.2 Computational Efficiency Analysis

To assess the practical deployability of GRAFlow in real-world microservice environments, we evaluated its time efficiency on both the TT and SN datasets.

Table 11 presents the total training time and testing time for both datasets. The training phase involves learning the parameters for the multi-scale graph attention networks and cross-modal fusion modules. The testing phase measures the time required for the model to perform anomaly detection and root cause localization on the entire test set.

As shown in Table 11, the training process for the TT dataset requires approximately 93 minutes, while the SN dataset requires 127 minutes. Although the model incorporates complex components such as GAT and attention mechanisms, the training time remains within a reasonable range (approx. 1.5 to 2 hours). Considering that model training is typically performed **offline** (e.g., daily or weekly updates), this computational cost is acceptable for industrial maintenance cycles.

More importantly, the **testing time** (inference phase) demonstrates the model's efficiency. GRAFlow completes the detection and localization tasks for the entire test set in just 1 minute for TT and 2 minutes for SN. This low latency indicates that the model can process monitoring data streams in near real-time, ensuring timely alerts when anomalies occur. This balance between detection accuracy and computational efficiency confirms that GRAFlow is well-suited for online microservice monitoring systems.

4.5.3 Hyperparametric Validity Analysis

To further investigate the impact of the hyperparameter α, which controls the trade-off between the anomaly detection and root cause localization tasks in the joint loss function, we conduct experiments on the TT and SN datasets with $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. A larger α indicates a stronger emphasis on the anomaly detection task, while a smaller value focuses more on root cause localization.

In this section, we use the F1 score to evaluate the performance of anomaly detection, and HR@K and NDCG@K to assess the effectiveness of root cause localization.

Figure 8 illustrates the performance trends on the TT and SN datasets as the hyperparameter α increases. Overall, the anomaly detection-related metrics remain consistently high across different values of α, indicating that the model maintains a strong capability to perceive anomalies.

However, the performance of root cause localization shows a noticeable difference: when α increases from 0.5 to 0.7 and 0.9, the HR and NDCG scores on the TT dataset drop significantly, while the SN dataset remains relatively stable.

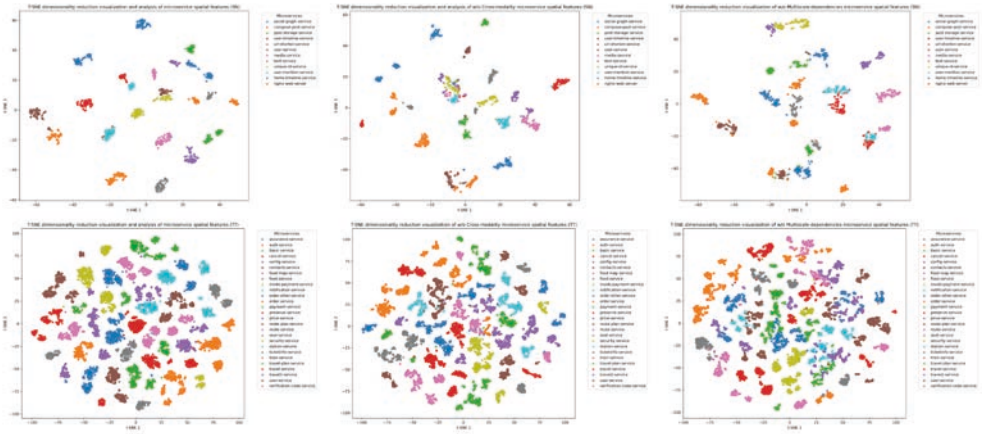


Figure 7. The GRAFlow model removes the distribution of T-SNE features from different modules separately

Table 11. Computation Time Analysis on TT and SN Datasets

Dataset	Training Time (min)	Testing Time (min)
TT	93	1
SN	127	2

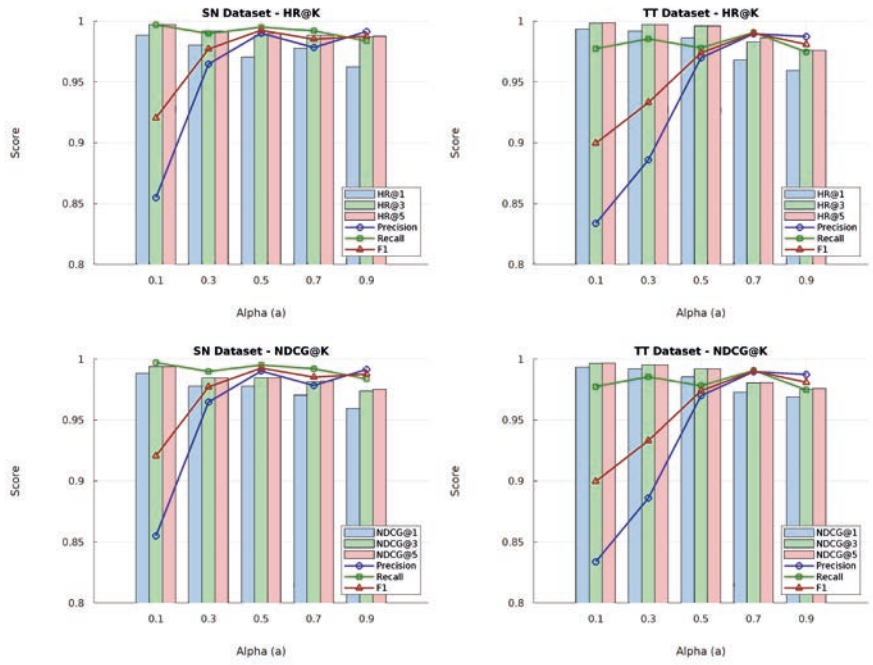


Figure 8. Analysis of the effect of hyperparameter a on the performance of anomaly detection and root cause localisation

This is attributed to the structural complexity of the TT dataset, which involves 27 microservices with longer anomaly durations, making anomaly propagation paths harder to trace. When α is set too high, the model training overly focuses on the anomaly detection task, which weakens the optimization of the root cause localization objective, and consequently impairs the ability to construct deeper propagation dependencies, leading to inaccurate root cause localization.

5 Conclusion

In this paper, we propose GRAFlow, a multimodal anomaly detection and root cause localization framework for microservice systems. By fusing logs, call chains, and performance metrics through a cross-modal attention mechanism and a multi-scale graph attention network, GRAFlow effectively captures the semantic synergies and structural dependencies inherent in system operations.

Extensive experiments validate that GRAFlow bridges the semantic gap between heterogeneous monitoring data. By integrating multi-scale graph topology, it overcomes the limitations of local-only analysis common in existing methods, providing a robust solution for diagnosing cascading failures in complex distributed systems. The superior performance across diverse datasets confirms the universal value of jointly modeling cross-modal interactions and structural dependencies for ensuring microservice reliability.

Future work will focus on three key directions. First, we plan to introduce dynamic graph modeling to capture time-varying service dependencies, addressing the challenges of frequent scaling and routing changes in real-world environments. Second, we aim to enhance interpretability by combining fine-grained indicator-level data with log events to provide deeper insights into root causes. Finally, we intend to migrate the model to a real-time detection framework and integrate it with industrial AIOps platforms to achieve efficient online deployment.

References

[1] Yu, G.; Chen, P.; Zheng, Z. Microscaler: Cost-effective Scaling for Microservice Applications in the Cloud with an Online Learning Approach. IEEE Transactions on Cloud Computing, 2020, 10(2), 1100–1116.

[2] Zhou, X.; Peng, X.; Xie, T.; et al. Fault Analysis and Debugging of Microservice Systems: Industrial Survey, Benchmark System, and Empirical Study. IEEE Transactions on Software Engineering, 2018, 47(2), 243–260.

[3] Singh, A.; Satapathy, S. C.; Roy, A.; et al. Ai-based mobile edge computing for iot: Applications, challenges, and future scope. Arabian Journal for Science and Engineering, 2022, 47(8): 9801–9831.

[4] Al-Shaarani, F.; Basakran, N.; Gutub, A. Sensing e-banking cybercrimes vulnerabilities via smart information sciences strategies. RAS Eng Technol, 2020, 1(1): 1–9.

[5] Alsaidi, A.; Gutub, A.; Alkhodaidi, T. Cybercrime on transportation airline. J. Forensic Res, 2019, 10(4): 449.

[6] Dragoni, N.; Giallorenzo, S.; Lluch Lafuente, A.; et al. Microservices: Yesterday, Today, and Tomorrow. Present and Ulterior Software Engineering, 2017, 195–216.

[7] Jacob, S. L., & Sultana Habibullah, P. (2024). A Systematic Analysis and Review on Intrusion Detection Systems Using Machine Learning and Deep Learning Algorithms. Journal of Computational and Cognitive Engineering, 4(2), 108-120. https://doi.org/10.47852/bonviewJCCE42023249

[8] Soldani, J.; Brogi, A. Anomaly Detection and Failure Root Cause Analysis in (Micro) Service-Based Cloud Applications: A Survey. ACM Computing Surveys (CSUR), 2022, 55(3), 1–39.

[9] Ma, Y.; Ding, K.; Li, R. Multimodal Spatiotemporal Enhanced Model for Anomaly Detection in Microservice Systems. Journal of Frontiers of Computer Science & Technology, 2025, 1–15.(Chinese)

[10] S, R., & Battineni, G. (2025). A Survey on Recent Advancements in Auto-Machine Learning with a Focus on Feature Engineering. Journal of Computational and Cognitive Engineering, 4(1), 56-63. https://doi.org/10.47852/bonviewJCCE3202720

[11] Zhang, S.; Jin, P.; Lin, Z.; et al. Robust Failure Diagnosis of Microservice System Through Multimodal Data. IEEE Transactions on Services Computing, 2023, 16(6), 3851–3864.

[12] Zhao, N.; Chen, J.; Yu, Z.; et al. Identifying Bad Software Changes via Multimodal Anomaly Detection for Online Service Systems. In Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2021, 527–539.

[13] Lee, C.; Yang, T.; Chen, Z.; et al. Eadro: An End-to-End Troubleshooting Framework for Microservices on Multi-Source Data. In Proceedings of the 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), 2023, 1750–1762.

[14] Zhang, C.; Peng, X.; Sha, C.; et al. Deeptralog: Trace-Log Combined Microservice Anomaly Detection Through Graph-Based Deep Learning. In Proceedings of the 44th International Conference on Software Engineering, 2022, 623–634.

[15] Chen, L.; Song, C.; Wang, X.; et al. CSCLog: A Component Subsequence Correlation-Aware Log Anomaly Detection Method. IEEE Transactions on Dependable and Secure Computing, 2025.

[16] Xie, Y.; Zhang, H.; Babar, M. A. Loggd: Detecting Anomalies from System Logs with Graph Neural Networks. In Proceedings of the 2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS), 2022, 299–310.

[17] Lee, C.; Yang, T.; Chen, Z.; et al. Heterogeneous Anomaly Detection for Software Systems via Semi-Supervised Cross-Modal Attention. In Proceedings of the 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), 2023, 1724–1736.

[18] Gutub, A.; Altalhi, S.; Ghazwani, B. Offline efficient signature authentication using octave convolution neural network. Arabian Journal for Science and Engineering, 2025, 50(19): 15673–15688.

[19] Alotaibi, M.; Al-hendi, D.; Alroithy, B.; et al. Secure mobile computing authentication utilizing hash, cryptography and steganography combination. Journal of Information Security and Cybercrimes Research, 2019, 2(1).

[20] Kheshaifaty, N.; Gutub, A. Engineering graphical captcha and AES crypto hash functions for secure online authentication. Journal of Engineering Research, 2023, 11(3): 69–80.

[21] Yu, G.; Chen, P.; Chen, H.; et al. Microrank: End-to-End Latency Issue Localization with Extended Spectrum Analysis in Microservice Environments. In Proceedings of the Web Conference 2021, 2021, 3087–3098.

[22] Liu, P.; Xu, H.; Ouyang, Q.; et al. Unsupervised Detection of Microservice Trace Anomalies Through Service-Level Deep Bayesian Networks. In Proceedings of the 2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE), 2020, 48–58.

[23] Liu, D.; He, C.; Peng, X.; et al. Microhecl: High-Efficient Root Cause Localization in Large-Scale Microservice Systems. In Proceedings of the 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), 2021, 338–347.

[24] Kim, M.; Sumbaly, R.; Shah, S. Root Cause Detection in a Service-Oriented Architecture. ACM SIGMETRICS Performance Evaluation Review, 2013, 41(1), 93–104.

[25] Wu, L.; Tordsson, J.; Elmroth, E.; et al. MicroRCA: Root Cause Localization of Performance Issues in Microservices. IEEE/IFIP Network Operations and Management Symposium (NOMS), 2020, 1–7.

[26] Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K. Q. Densely Connected Convolutional Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, 4700–4708.

[27] Morris, C.; Ritzert, M.; Fey, M.; et al. Weisfeiler and Leman Go Neural: Higher-Order Graph Neural Networks. In Proceedings of the AAAI Conference on Artificial Intelligence, 2019, 33(1), 4602–4609.

[28] Xu, K.; Li, C.; Tian, Y.; et al. Representation Learning on Graphs with Jumping Knowledge Networks. In Proceedings of the International Conference on Machine Learning, 2018, 5453–5462.

[29] Xu, C.; Cui, Z.; Hong, X.; et al. Graph Inference Learning for Semi-Supervised Classification. arXiv preprint arXiv:2001.06137, 2020.

[30] Laddi, M., Allagi, S. ., Rachh, R., Sambrekar, K. ., & Athanikar, S. (2025). An Advanced Cyber Security Model Using Federated Machine Learning Approach for Intrusion Detection in Networks. Journal of Computational and Cognitive Engineering, 4(2), 223-235. https://doi.org/10.47852/bonviewJCCE42023751

[31] Yan, H.; Breslau, L.; Ge, Z.; et al. G-RCA: A Generic Root Cause Analysis Platform for Service Quality Management in Large IP Networks. In Proceedings of the 6th International Conference, 2010, 1–12.

[32] Wu, L.; Tordsson, J.; Bogatinovski, J.; et al. Microdiag: Fine-Grained Performance Diagnosis for Microservice Systems. In 2021 IEEE/ACM International Workshop on Cloud Intelligence (CloudIntelligence), 2021, 31–36.

[33] Zhao, N.; Jin, P.; Wang, L.; et al. Automatically and Adaptively Identifying Severe Alerts for Online Service Systems. In IEEE INFOCOM 2020—IEEE Conference on Computer Communications, 2020, 2420–2429.

[34] Hu, C.; Sun, X.; Dai, H.; et al. Research on Log Anomaly Detection Based on Sentence-BERT. Electronics, 2023, 12(17), 3580.

[35] Le, V.-H.; Zhang, H. Log-Based Anomaly Detection Without Log Parsing. In 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE), 2021, 492–504.

[36] Xiao, T.; Quan, Z.; Wang, Z.-J.; et al. Loader: A Log Anomaly Detector Based on Transformer. IEEE Transactions on Services Computing, 2023, 16(5), 3479–3492.

[37] Luo, L. Network Text Sentiment Analysis Method Combining LDA Text Representation and GRU-CNN. Personal and Ubiquitous Computing, 2019, 23(3), 405–412.

[38] Guo, H.; Yuan, S.; Wu, X. LogBERT: Log Anomaly Detection via BERT. In 2021 International Joint Conference on Neural Networks (IJCNN), 2021, 1–8.

[39] Yu, J.-N.; Hu, Z.-X.; Jiang, C.-F. Multi-Feature-Based Log Event Anomaly Detection. Computer Engineering & Science, 2024, 46(9).

[40] Qiu, K.; Zhang, Y.; Feng, Y.; Chen, F. LogAnomEX: An Unsupervised Log Anomaly Detection Method Based on Electra-DP and Gated Bilinear Neural Networks. Journal of Network and Systems Management, 2025, 33(2), 1–29.

[41] Qiu, K.; Yan, M.; Luo, T.; et al. Fedaware: A Distributed IoT Intrusion Detection Method Based on Fractal Shrinking Autoencoder. Journal of King Saud University Computer and Information Sciences, 2025, 37(7), 202.

[42] Zhao, N.; Chen, J.; Yu, Z.; et al. Identifying Bad Software Changes via Multimodal Anomaly Detection for Online Service Systems. In Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2021, 527–539.

[43] Zhang, S.; Jin, P.; Lin, Z.; et al. Robust Failure Diagnosis of Microservice System Through Multimodal Data. IEEE Transactions on Services Computing, 2023, 16(6), 3851–3864.

[44] Zhu, Z.; Lee, C.; Tang, X.; He, P. HeMiRCA: Fine-Grained Root Cause Analysis for Microservices with Heterogeneous Data Sources. ACM Transactions on Software Engineering and Methodology, 2024, 33(8), 1–25.

[45] Zhao, C.; Ma, M.; Zhong, Z.; et al. Robust Multimodal Failure Detection for Microservice Systems. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2023, 5639–5649.

[46] Li, Y.; Guo, Y.; Chen, Y.; et al. Self-Supervised Spatio-Temporal Representation Learning for Microservice Anomaly Detection. 2025 8th World Conference on Computing and Communication Technologies (WCCCT), 2025, 278–284.

[47] Zhou, X.; Peng, X.; Xie, T.; et al. Benchmarking Microservice Systems for Software Engineering Research. In Proceedings of the 40th International Conference on Software Engineering: Companion Proceedings, 2018, 323–324.

[48] Gan, Y.; Zhang, Y.; Cheng, D.; et al. An Open-Source Benchmark Suite for Microservices and Their Hardware-Software Implications for Cloud & Edge Systems. In Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, 2019, 3–18.

[49] Shambour, M. K.; Gutub, A. Personal privacy evaluation of smart devices applications serving Hajj and Umrah rituals. Journal of Engineering Research, 2023, 11(2): 1–14.

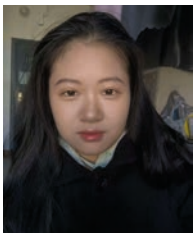
[50] Gutub, A. Dynamic smart random preference for higher medical image confidentiality. Journal of Engineering Research, 2023, 11(3): 113–123.

Zhao Shuangshi earned a master's degree in Computer Science and Technology from Ajou University in South Korea in 2026. He currently works for a Chinese internet company, focusing on system development, artificial intelligence, and cybersecurity.
https://orcid.org/0009-0003-2671-6998

Kunming Liu received his master's degree in Big Data from Ajou University, Korea, under the supervision of Professor Gaoyang Shan.
https://orcid.org/0009-0008-3592-6694

Jianlin Lu is currently pursuing a degree in Software Engineering at Shihezi University. His research interests primarily include software engineering and deep learning.
https://orcid.org/0009-0004-3276-6729

Zhejie Xu is currently pursuing a degree in Software Engineering at Shihezi University. His research interests primarily include cybersecurity and deep learning.
https://orcid.org/0009-0001-7851-8032



Meifang Yan received the B.S. degree in Software Engineering from Pingdingshan University and is currently pursuing the M.S. degree in Computer Technology at Xinjiang Normal University. Her research interests include bioinformatics and medical informatics.
https://orcid.org/0009-0007-1192-4216

Keyuan Qiu received the B.S. degree in Software Engineering from Southwest Minzu University in 2023 and is currently pursuing the M.S. degree in Electronic Information at Shihezi University. His research interests include network intrusion detection and system anomaly detection.
https://orcid.org/0009-0005-6613-5711