EAI Endorsed Transactions
on Scalable Information Systems
Research Article EAI.EU

A Security Monitoring and Multi-source Anomaly Detection Method for New Distributed Systems Oriented to Economic Data Asset Circulation

Chenfei Zhuang[1,*], Jiajia Cao[1], Xingchao Ni[2]

[1]Business School, Nantong Institute of Technology, Nantong, Jiangsu, China
[2]Financial Department, Suzhou University of Technology, Suzhou, Jiangsu, China

Abstract

To address the difficulty of unified security risk monitoring caused by the coupling of on-chain auditing, microservice invocation, distributed databases, and network communication states during economic data asset circulation, this paper proposes a multi-source spatio-temporal graph anomaly detection method for new distributed systems, namely MSTG-AD. The method aligns five monitoring sources—transaction behavior, on-chain auditing, microservice invocation, database metrics, and network traffic—with a six-entity relation schema covering users, data assets, contracts, services, databases, and network nodes. Rather than claiming novelty for the individual temporal, Transformer, graph-attention, or spectral operators, the proposed contribution lies in their cross-layer semantic coupling: multi-window temporal encoding captures periodic changes and abrupt fluctuations, call-chain/database encoding characterizes service and node states, an asset-context-aware gate prioritizes anomaly evidence according to value and exposure context, and a relation-constrained decoder traces temporally consistent root-cause paths along the economic-data-asset chain. On a simulated economic data asset trading platform and a multi-source fusion test set, MSTG-AD achieves a Precision of 95.1%, a Recall of 94.0%, an F1-score of 94.5%, and an AUC of 0.973, with an average detection latency of 38.6 ms. In the root cause localization task, AC@1, AC@3, and AC@5 reach 0.91, 0.96, and 0.98, respectively. The results demonstrate the effectiveness of full-link multi-source modeling for anomaly identification, risk monitoring, and explainable root-cause localization.

Keywords: economic data assets, new distributed systems, security monitoring, multi-source anomaly detection, dynamic heterogeneous graph, root cause localization

Received on 30 June 2026, accepted on 08 September 2026, published on 18 November 2026

Copyright © 2026 Chenfei Zhuang *et al.*, licensed to EAI. This is an open access article distributed under the terms of the CC BY-NC-SA 4.0, which permits copying, redistributing, remixing, transformation, and building upon the material in any medium so long as the original work is properly cited.

doi: 10.4108/eetsis.13812

1. Introduction

1.1. Research Background and Problem Statement

Economic data assets are evolving from static information resources into tradable production factors whose circulation spans registration, authorization, transaction matching, contract execution, audit tracing, privacy protection, and risk supervision. New distributed infrastructures—including consortium/side chains, microservices, distributed databases, edge computing, and federated learning—support this process, but they also couple business behavior with contract, service, storage,

*Corresponding author. Email: zhuangcf0215@163.com

and network states. Consequently, an incident may propagate across layers, for example from abnormal access to contract failure, service degradation, database pressure, and suspicious network communication. This cross-layer coupling motivates continuous full-link security monitoring rather than independent protection at a single technical layer.

Existing studies have explored data asset protection and security management in distributed environments from different perspectives. Zhang and Yan proposed a cross-chain data asset rights-confirmation framework based on hybrid post-quantum zero-knowledge proofs to improve cross-chain interoperability and verifiability [1]. Tian developed a generative AI-based privacy protection and security management mechanism for distributed financial systems that integrates privacy-preserving data generation, fine-grained access control, and anomaly attack detection [2]. Guo et al. proposed a privacy-preserving secure aggregation scheme for federated learning with input verification and dropout resistance [3]. These studies provide relevant foundations for rights confirmation, distributed financial security, and privacy-preserving collaboration, but they do not jointly model transaction, service, database, and network anomalies along the full economic-data-asset circulation chain.

Recent English-language studies provide direct references for trustworthy data trading. Fang et al. proposed a blockchain-cloud architecture for distributed industrial data trading that combines privacy-enhanced verifiable credentials, smart-contract-based fair trading, and blockchain/multisignature-based data-integrity protection [4]. Xue et al. developed a blockchain-based fair and fine-grained data-trading scheme that preserves seller identity privacy and supports selective disclosure of required data attributes [5]. These studies indicate that the security of economic data assets depends not only on cryptographic protection and transaction fairness, but also on continuous monitoring of transaction execution, system operation, and network communication states. However, existing work focuses more on secure trading mechanisms or privacy protection, while unified modeling for full-process security monitoring and multi-source anomaly detection of economic data assets in new distributed systems remains insufficient.

1.2. Research Challenges

Economic data asset security monitoring is essentially an anomaly detection and root cause explanation problem based on multi-source heterogeneous monitoring data. Compared with a traditional single business system, a new distributed data asset platform includes multiple layers such as on-chain transactions, off-chain computing, microservice interfaces, database storage, cross-chain communication, and network access. Anomalies may be jointly triggered by business behavior, contract execution, service invocation, database nodes, or network traffic. Recent KPI anomaly-detection research uses self-supervised spatio-temporal graph attention to jointly model temporal dynamics and cross-KPI dependencies, demonstrating the value of relational KPI modeling for operational anomaly detection [6]. However, the monitoring data of economic data asset platforms include not only traditional KPIs such as CPU, memory, throughput, and response time, but also security-related indicators such as transaction frequency, asset access count, authorization failure rate, contract invocation results, cross-chain query latency, database transaction state, and network connection behavior, making the data sources more complex.

First, multi-source monitoring data show significant heterogeneity. Transaction logs, on-chain audit records, microservice call chains, database operation metrics, and network traffic differ greatly in data structure, temporal granularity, and anomaly expression, making them difficult to directly input into a unified model. Second, the monitoring data exhibit strong temporal characteristics and multi-scale patterns. Recent multi-scale time-series anomaly detection models extract features at different temporal scales to improve sensitivity to diverse anomaly patterns [7]. In economic data asset platforms, transaction volume, access volume, invocation latency, and database load may have different periods and burst patterns, so adaptive temporal modeling mechanisms should be designed for multidimensional indicators. Third, microservice call chains affect anomaly manifestation. Trace-based anomaly-detection studies model structural relationships among service calls and trace-level features, providing call-path context for abnormal invocation detection [8]. Fourth, distributed database anomalies often emerge from interactions among workloads, queries, and system metrics. Production-scale database troubleshooting shows that noisy, non-stationary, and quasi-periodic workloads motivate combining anomaly detection with root-cause analysis and bottleneck localization [9]. Fifth, an economic data asset platform can be further abstracted as a dynamic heterogeneous graph consisting of users, data assets, contracts, services, database nodes, and network traffic. Recent masked-autoencoder work on dynamic incomplete graphs addresses changing structures and incomplete graph observations through self-supervised graph reconstruction [10]. Meanwhile, graph-learning approaches for data-link anomaly detection show that inter-link structural information is important for identifying abnormal transmission states [11].

Therefore, security monitoring for new distributed systems oriented to economic data assets needs to simultaneously address multi-source data fusion, multi-scale temporal modeling, call-chain dependency modeling, dynamic heterogeneous graph anomaly identification, and explainable root cause localization. If only a single KPI threshold, on-chain audit rule, or isolated anomaly detection model is used, it is difficult to discover potential correlations among transaction behavior anomalies, system performance degradation, and network attacks, or to provide actionable root cause explanations for platform operation and supervision.

1.3. Related Work and Limitations

Existing work is reorganized into three categories according to the monitoring boundary rather than the specific learning algorithm. The first category concerns trusted data-asset trading, blockchain audit, and privacy protection. Cross-chain rights confirmation and privacy-preserving data trading have been investigated for data-asset circulation [1, 4, 5]. Secure aggregation provides a complementary privacy-preserving mechanism for distributed collaborative learning [3]. In parallel, Zhang et al. applied blockchain and zero-knowledge proofs to secure data transactions [12]; Kuznetsov et al. studied aggregated zero-knowledge-proof mechanisms for blockchain security and efficiency [13]; and Quattrocchi and Plebani used zero-knowledge proofs with blockchain for trustworthy collaborative business intelligence [14]. These strands strengthen ownership verification, transaction integrity, or privacy protection, but they do not continuously correlate on-chain events with downstream microservice, database, and network states after an asset transaction is executed.

The second category concerns multi-source observability and root-cause analysis in microservice systems. Recent work has progressed from two-source fusion, such as log–metric joint detection in LMGD [15], to heterogeneous observability for failure prediction and root-cause analysis. InstantOps explicitly exploits logs, metrics, and traces [16]; HeMiRCA correlates trace-based anomaly scores with multivariate metrics for fine-grained root-cause analysis [17]; and MRCA and multimodal adaptive optimization further exploit multiple observability modalities [18, 19]. BARO is a metric-centric root-cause analysis method that uses multivariate Bayesian online change-point detection and is reported to be robust to variations in anomaly-detection timing [20], whereas MULAN learns causal structures from multi-modal observability data for root-cause analysis [21]. More recent anomaly-detection studies integrate augmented multimodal data with hybrid graph representations [22] or model nested trace–log structures [23], while dynamic knowledge graphs capture evolving system context for context-aware anomaly detection [24]. These studies demonstrate the benefit of cross-source observability, but their semantic scope remains centered on software services and infrastructure. Economic data-asset semantics—asset value, authorization, contract audit, payment evidence, and cross-chain exposure—are generally outside their monitoring schema.

The third category concerns temporal, graph, database, and network anomaly modeling. KPI studies capture spatio-temporal dependencies and multi-scale patterns [6, 7]; trace-based methods model structural properties of microservice calls [8]; production database troubleshooting combines anomaly detection with root-cause analysis and bottleneck localization [9]; and recent graph methods use masked reconstruction or link-structure modeling for evolving/incomplete graphs and transmission anomalies [10, 11]. These techniques provide strong building blocks, but applying them independently leaves three unresolved cross-layer gaps. First, heterogeneous transaction, audit, service, database, and traffic evidence lacks a shared economic-data-asset relation schema. Second, fusion weights usually reflect feature reliability but not the value/sensitivity/exposure context of the circulating asset. Third, generic node ranking does not enforce a physically valid and temporally consistent propagation path across the full user-asset-contract-service-database-network chain. Quantitatively, recent service-oriented studies usually combine two or three observability modalities, whereas MSTG-AD aligns five monitoring categories and six entity types. Performance numbers across different public datasets are therefore not compared directly here; the novelty boundary is the broader cross-layer semantic coverage and constrained traceable reasoning required by economic data asset circulation.

1.4. Research Content and Contributions

To address the above gaps, this paper proposes MSTG-AD for full-link security monitoring of economic data asset circulation. The platform is represented as a dynamic heterogeneous system linking users, data assets, smart contracts, microservices, distributed database nodes, and network connections. The novelty of MSTG-AD does not rely on treating Transformer, graph attention, masking, or spectral filtering as new algorithms in isolation. Instead, it lies in coupling these mature operators through an economic-data-asset relation schema, asset-context-aware risk weighting, and relation-constrained root-cause path reasoning so that multi-source evidence can be jointly interpreted at the level of an actual asset circulation chain. The main contributions of this paper are summarized as follows.

(i) A full-link semantic monitoring framework is constructed for economic data asset circulation. It aligns transaction behavior, on-chain audit events, service invocations, database states, and network connections with six typed entities and their transaction, authorization, audit, invocation, storage, and communication relations, providing a common representation for cross-layer anomaly propagation.

(ii) A five-category multi-source indicator system is designed and augmented with economic-asset context. In addition to transaction frequency, authorization failures, contract states, call-chain latency, database load, and abnormal connections, asset value level, privacy sensitivity, authorization criticality, and cross-chain exposure are retained as contextual variables for risk prioritization rather than as anomaly labels.

(iii) A cross-layer anomaly detector is proposed by combining multi-window temporal encoding, call-chain and database state encoding, relation-aware dynamic heterogeneous graph learning, and spectral-aware network representation. A data-quality- and

asset-context-aware gating network dynamically fuses the module scores, making the comprehensive risk score responsive to both evidence reliability and economic impact context.

(iv) A relation-constrained and temporally consistent root-cause localization mechanism is proposed for the full industrial chain. Beyond node anomaly probability and masking contribution, candidate paths are restricted by valid relation types and anomaly timestamps, so the decoder can output evidence-supported propagation paths such as user→asset→contract→service→database→network without forcing unsupported layers.

2. Scenario Modeling and Problem Definition

2.1. Distributed Circulation Scenario of Economic Data Assets

Economic data asset circulation usually involves multiple entities such as data providers, data demanders, trading platforms, smart contracts, audit nodes, microservice interfaces, distributed databases, and network communication environments. Unlike traditional centralized data management systems, new distributed data asset platforms need not only to complete data registration, authorized access, transaction matching, usage record archiving, and audit tracing, but also to ensure data authenticity, transaction fairness, privacy protection, and system operation stability. Blockchain-cloud industrial data-trading studies emphasize fairness, privacy-enhanced identity management, data-integrity protection, and smart-contract-based transaction management [4]. Blockchain-based fine-grained trading further shows the need to balance fair settlement, selective disclosure, and seller privacy [5].

This paper abstracts the economic data asset platform as a new distributed system composed of six core object classes:

$$\mathcal{O} = \{U, D, C, S, B, N\} \tag{1}$$

where U denotes the user set, D denotes the economic data asset set, C denotes the set of smart contracts and on-chain audit components, S denotes the microservice set, B denotes the distributed database node set, and N denotes the network traffic and communication connection set. The core task of economic data asset security monitoring is to continuously perceive state changes of the above objects and their interactions, identify anomalous behavior, and provide explainable root cause localization results. Equation (1) defines the six object classes used throughout the subsequent monitoring model.

2.2. Threat Model and Anomaly Types

This paper assumes that the trading platform, consortium-chain consensus mechanism, and basic audit contracts can execute correctly under normal conditions. However, risk sources such as malicious users, abnormal services, faulty database nodes, and external network attacks may still exist in the system. Malicious data providers may upload false data or data assets inconsistent with their descriptions; malicious data demanders may bypass authorization, repeatedly invoke interfaces, or deny receiving data; external attackers may attack the platform through interface blasting, abnormal connections, protocol obfuscation, or data exfiltration; internal nodes may also cause anomaly propagation because of sudden load increases, resource bottlenecks, or service link blocking. According to the circulation process of economic data assets, anomalies are divided into four categories in this paper.

(i) Transaction behavior anomalies include abnormal high-frequency access, dense transactions in a short period, abnormal authorization requests, batch asset invocation, and sudden changes in price or access frequency, mainly reflecting risks such as illegal access, malicious transactions, or data scraping.

(ii) Contract audit anomalies include contract invocation failure, signature verification failure, evidence hash inconsistency, cross-chain query latency, abnormal payment status, and receipt verification failure, mainly reflecting trustworthiness risks in on-chain transaction and audit processes.

(iii) System operation anomalies include microservice response time anomalies, abnormal call-chain paths, increased interface failure rate, database node CPU bottlenecks, I/O bottlenecks, transaction latency, and multi-node compound anomalies. Such failures can be driven by interacting workload and resource signals rather than a single metric; production-scale database troubleshooting therefore combines anomaly detection with root-cause analysis to identify performance bottlenecks and culprit queries [9].

(iv) Network traffic anomalies include abnormal connections, port hopping, abnormal outbound connections, protocol obfuscation, low-frequency stealth access, and abnormal communication paths. Graph-learning methods for data-link anomaly detection further show that structural dependencies among links provide useful evidence for abnormal transmission states [11].

The above anomalies are often not isolated. For example, abnormal access may trigger microservice delay, microservice blocking may further cause database node load anomalies, and network attacks may also appear as increased contract failure rates or authorization failure rates. Therefore, this paper defines economic data asset

security monitoring as a multi-source joint anomaly detection problem rather than a single-indicator threshold judgment problem.

2.3. Multi-source Monitoring Data Representation

To comprehensively characterize the security state of economic data asset platforms, this paper constructs a multi-source monitoring indicator system consisting of transaction behavior, on-chain auditing, microservice invocation, distributed databases, and network traffic. KPI anomaly-detection systems learn temporal and cross-metric dependencies from multivariate operational streams, and recent self-supervised spatio-temporal graph models demonstrate the value of explicitly representing these dependencies [6]. On this basis, data asset transaction and on-chain audit indicators are further introduced to adapt to the economic data asset circulation scenario. The multi-source security monitoring indicator system constructed in this paper is shown in Table 1.

Table 1. Multi-source security monitoring indicator system

Indicator Category	Representative Indicators	Corresponding Risks
Transaction Behavior Indicators	Transaction frequency, access count, authorization failure rate, asset popularity, download count; contextual metadata: asset value level, privacy sensitivity, authorization criticality, cross-chain exposure	Unauthorized access, malicious transaction, data scraping
On-Chain Audit Indicators	Contract invocation status, signature verification result, deposit hash consistency, cross-chain query latency	Contract failure, data fraud, payment repudiation
Microservice Invocation Indicators	Response time, call chain length, interface failure rate, service dependency change	Service abnormality, link blocking, cascading failure
Distributed Database Indicators	CPU utilization, I/O wait, cache hit rate, transaction latency, SQL execution time	Database bottleneck, multi-node abnormality
Network Traffic Indicators	Number of connections, port change, protocol type, outbound frequency, abnormal communication path	Data leakage, lateral movement, protocol obfuscation

As shown in Table 1, security risks in economic data asset platforms are not only reflected in transaction behavior itself, but also involve on-chain auditing, service invocation, database operation, and network communication. Therefore, subsequent anomaly detection models need to uniformly represent and jointly model multiple types of monitoring indicators, rather than relying only on a single transaction log or a single system KPI. Asset value, privacy sensitivity, authorization criticality, and cross-chain exposure are treated as contextual variables: they adjust risk prioritization but do not by themselves constitute anomaly evidence.
The multi-source monitoring data at time t are defined as:

$$X_t = \{X_t^{\text{tra}}, X_t^{\text{audit}}, X_t^{\text{svc}}, X_t^{db}, X_t^{\text{net}}\} \tag{2}$$

where the five components denote transaction behavior, on-chain auditing, microservice invocation, distributed database, and network traffic indicators, respectively. Given the window length w, the continuous monitoring sequence is represented as:

$$\mathcal{X}_t^w = \{X_{t-w+1}, X_{t-w+2}, \dots, X_t\} \tag{3}$$

Different indicators usually have different periodicities and burst patterns, and a unified temporal resolution may fail to capture anomaly signatures at all scales. Multi-scale time-series anomaly detection research shows that extracting features from multiple temporal scales can improve sensitivity to diverse anomalies [7]. Therefore, the subsequent method performs multi-scale temporal modeling on multi-source indicators. Equations (2) and (3) define the instantaneous multi-source monitoring state and its continuous temporal window, respectively.

2.4. Dynamic Heterogeneous Security Graph Construction

Economic data asset circulation includes not only multidimensional time series, but also complex object relationships. For example, users access data assets, data assets are bound to smart contracts, contracts invoke microservices, microservices access database nodes, and different services and nodes communicate through network connections. To uniformly represent these relationships, this paper constructs a dynamic heterogeneous security graph:

$$G_t = (V_t, E_t, A_t, R_t) \tag{4}$$

where Vt denotes the node set, Et denotes the edge set, At denotes the node attribute matrix, and Rt denotes the edge type set. The node set is defined as:

$$V_t = U_t \cup D_t \cup C_t \cup S_t \cup B_t \cup N_t \tag{5}$$

The edge set includes transaction edges, authorization edges, invocation edges, storage edges, audit edges, and communication edges:

$$E_t = E_t^{\text{tra}} \cup E_t^{\text{auth}} \cup E_t^{\text{call}} \cup E_t^{\text{store}} \cup E_t^{\text{audit}} \cup E_t^{\text{comm}} \quad (6)$$

In this graph, anomalies may appear as node attribute anomalies, edge weight anomalies, structural anomalies, or subgraph anomalies. For example, if a data asset is accessed intensively in a short period, the in-degree of the asset node increases abnormally; if a database node simultaneously suffers from CPU and I/O bottlenecks, it appears as a node-level compound anomaly; if an external address establishes atypical connections with multiple internal services, it appears as a network structure anomaly. Recent work on dynamic incomplete graphs shows that masked autoencoding can exploit incomplete and evolving graph observations for anomaly detection [10]. Equations (4)-(6) specify the dynamic heterogeneous graph, its typed node set, and the corresponding relation-specific edge set.

2.5. Formal Problem Definition

Based on the above modeling, economic data asset security monitoring is defined as a joint task of multi-source anomaly detection and root cause localization. Given the continuous monitoring sequence, the dynamic heterogeneous security graph Gt, and the historical state, the model outputs the anomaly category, anomaly score, risk level, and root cause path at the current moment:

$$(\hat{y}_t, \hat{s}_t, \hat{l}_t, \hat{P}_t) = F(\mathcal{X}_t^w, G_t, \mathcal{H}_{t-1}; \Theta) \quad (7)$$

where the outputs denote the anomaly category, anomaly score, risk level, root cause path, and model parameters, respectively.

The anomaly category set is defined as:

$$\mathcal{Y} = \{\text{normal}, A_{\text{tra}}, A_{\text{audit}}, A_{\text{svc}}, A_{db}, A_{\text{net}}, A_{\text{mix}}\} \quad (8)$$

where normal denotes the normal state; Atra, Aaudit, Asvc, Adb, and Anet denote transaction behavior anomalies, contract audit anomalies, microservice anomalies, distributed database anomalies, and network traffic anomalies, respectively; and Amix denotes compound anomalies. The comprehensive anomaly score is defined as:

$$\hat{s}_t = \alpha s_t^{tra} + \beta s_t^{audit} + \gamma s_t^{svc} + \delta s_t^{db} + \eta s_t^{net} + \mu s_t^{graph} \quad (9)$$

where the component scores denote the transaction, audit, microservice, database, network, and graph structure anomaly scores, respectively, and:

$$\alpha + \beta + \gamma + \delta + \eta + \mu = 1 \quad (10)$$

The root cause path is represented as an ordered sequence composed of anomalous nodes and their dependencies:

$$\hat{P}_t = \{v_1 \to v_2 \to \cdots \to v_k\}, \quad v_i \in V_t \quad (11)$$

For example, when abnormal access causes contract invocation failure, microservice delay, and database load increase, the root cause path can be represented as:

$$U_i \to D_j \to C_m \to S_n \to B_q \quad (12)$$

To improve the explainability of root cause localization, this paper considers both node anomaly probability and node influence. Recent microservice root-cause analysis studies show that robust localization can benefit from combining anomaly evidence with probabilistic change-point signals and causal dependency structures rather than relying on isolated anomaly scores [20, 21]. Therefore, the root cause node importance is defined as:

$$I(v_i) = p(v_i) \cdot R(v_i) \quad (13)$$

where the two terms denote the node anomaly probability and the node influence score in the dynamic heterogeneous security graph, respectively. The final root cause path is jointly determined by node importance, edge dependency strength, and anomaly propagation direction. Equations (7)–(13) jointly formalize the anomaly outputs, category set, fused anomaly score, root-cause path representation, and node-importance measure.

3. Machine Learning-based Anomaly Detection and Root Cause Localization Method

3.1. Overall Framework of the Method

To address multi-source monitoring data heterogeneity, compound anomaly types, and difficulty in explaining root cause links in economic data asset platforms, this paper proposes a multi-source spatio-temporal graph anomaly detection method, namely MSTG-AD. The method takes multi-source monitoring data such as transaction behavior, on-chain auditing, microservice invocation, distributed databases, and network traffic as inputs; jointly learns multidimensional time-series changes, service invocation dependencies, database node states, and dynamic heterogeneous graph structural anomalies; and finally outputs the anomaly category, anomaly score, risk level, and root cause path.

Given the continuous monitoring sequence and the dynamic heterogeneous security graph Gt, the overall mapping relationship of MSTG-AD is defined as:

$$(\hat{y}_t, \hat{s}_t, \hat{l}_t, \hat{P}_t) = F_\theta(\mathcal{X}_t^w, G_t) \quad (14)$$

where the outputs denote the anomaly category, anomaly score, risk level, root cause path, and model parameters, respectively. The overall framework of MSTG-AD is shown in Figure 1. Equation (14) defines the end-to-end mapping implemented by the proposed MSTG-AD framework.



Figure 1. Overall framework of the proposed MSTG-AD method

As shown in Figure 1, MSTG-AD first performs normalization, time alignment, window segmentation, and feature construction on transaction behavior data, on-chain audit data, microservice invocation data, distributed database metrics, and network traffic data. Then, multi-source anomaly features are extracted through the multi-window temporal encoding module, the call-chain and database state encoding module, the dynamic heterogeneous graph anomaly detection module, and the network traffic spectral-aware representation module. Finally, the model outputs the anomaly category, anomaly score, and risk level through the anomaly score fusion and risk assessment module, and generates an explainable root cause path through the root cause localization module

3.2. Multi-window Temporal Encoding Module

Transaction frequency, access count, authorization failure rate, contract invocation latency, service response time, database load, and network traffic in economic data asset platforms all have temporal characteristics. Different indicators have inconsistent periods and fluctuation patterns, and a unified window may truncate or blur anomaly signatures at some temporal scales. Recent multi-scale time-series anomaly detection methods demonstrate that learning features across different temporal scales improves coverage of heterogeneous anomaly patterns [7]. Let the d-th monitoring indicator sequence be:

$$X^d = \{x_1^d, x_2^d, \ldots, x_T^d\} \quad (15)$$

where T denotes the sequence length. According to the period and fluctuation characteristics of different indicators, a multi-scale window set is constructed:

$$\mathcal{W} = \{w_s, w_m, w_l\} \tag{16}$$

where ws, wm, and wl denote the short window, medium window, and long window, respectively. The short window captures sudden anomalies, the medium window identifies persistent anomalies, and the long window characterizes periodic shifts and chronic risk accumulation. For each window, the local monitoring segment is extracted and encoded as: The three windows therefore play complementary roles: the short branch preserves sensitivity to bursts, whereas the medium and long branches prevent a transient spike from being confused with persistent or periodic behavior.

$$h_t^r = Encoder_{ts}\left(X_t^{w_r}\right) \tag{17}$$

Temporal features under different windows are fused through an attention mechanism:

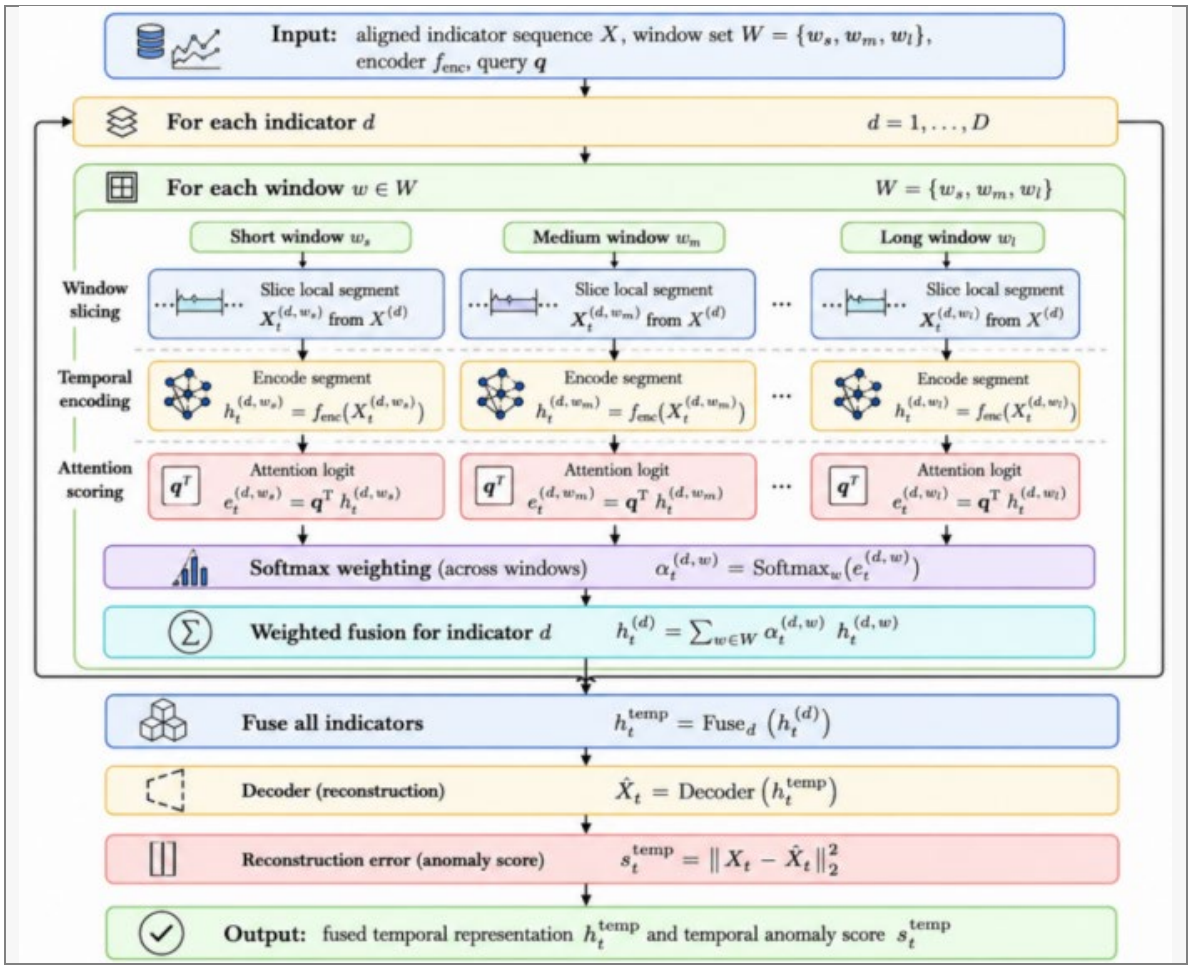
$$h_t^{ts} = \sum_{r\in\mathcal{W}} \omega_r \, h_t^r, \quad \omega_r = \frac{exp(q^\top h_t^r)}{\sum_{k\in\mathcal{W}}^{\Sigma\left(q^\top h_t^k\right)} exp} \tag{18}$$

where q is a learnable query vector, and the weights represent the importance of different windows. The temporal anomaly score is obtained from the reconstruction error between the observed sequence and the reconstructed sequence: The attention coefficients are normalized across the candidate windows for each indicator, which makes the fused representation comparable even when different indicators select different dominant time scales.

$$s_t^{ts} = \left\| X_t^w - \hat{X}_t^w \right\|_2^2 \tag{19}$$

When the current monitoring state deviates greatly from the learned normal temporal pattern, the anomaly score increases, indicating a higher anomaly risk in the current time window. Equations (15)–(19) define the multi-window temporal sequence, window set, temporal encoding, attention fusion, and reconstruction-based temporal anomaly score. The complete computational flow of the multi-window temporal encoding module is summarized in Algorithm 1.

Algorithm 1. Multi-window attention temporal encoding



3.3. Call-chain and Database State Encoding Module

Economic data asset platforms usually adopt a microservice architecture to implement authentication, authorization, asset retrieval, transaction matching, contract invocation, payment management, and audit tracing. The same service can participate in different invocation structures, so judging anomalies only from an isolated response-time threshold may produce false alarms. Trace-based microservice anomaly detection models structural relationships among service calls and trace-level features, providing richer context for abnormal invocation detection [8]. A call chain is represented as:

$$T_i = \{(op_1, d_1), (op_2, d_2), \dots, (op_n, d_n)\} \quad (20)$$

where the first component denotes the i-th service invocation operation, and di denotes the corresponding response time. The operation embedding vector is added to the positional encoding to obtain the call-chain input:

$$e_i = Embedding(op_i) + PE(i) \quad (21)$$

Then, a Transformer encoder is used to learn the call-chain context:

$$H^{\text{trace}} = Transformer(e_1, e_2, \dots, e_n) \quad (22)$$

The call-chain anomaly score is defined as the average deviation between the actual response time and the predicted response time:

$$s_t^{svc} = \frac{1}{n}\sum_{i=1}^{n}|d_i - \hat{d}_i| \quad (23)$$

where the predicted value denotes the normal response time estimated by the model according to the call path. If a service significantly deviates from the normal response distribution under a specific call path, the service node is assigned a higher anomaly probability. For the distributed database, let the node set be given, and the state vector of the i-th node at time t is: This path-conditioned residual is then attached to the corresponding service node, allowing the graph layer to distinguish a local latency fluctuation from a delay that is consistent with upstream contract or downstream database evidence.

$$z_{i,t}^{db} = [cpu_{i,t}, io_{i,t}, \text{mem}_{i,t}, \text{cache}_{i,t}, \text{sql}_{i,t}, \text{latency}_{i,t}] \quad (24)$$

The database node state representation is:

$$h_{i,t}^{db} = MLP(z_{i,t}^{db}) \quad (25)$$

Distributed database anomalies may appear as single-node anomalies, multi-node anomalies, or compound anomalies involving multiple metrics and workload conditions. Production database troubleshooting at Amazon scale shows that performance bottlenecks may arise under noisy, non-stationary, and quasi-periodic workloads and can be localized by combining anomaly detection with root-cause analysis [9]. Therefore, database node states are further incorporated into the dynamic heterogeneous security graph for subsequent relationship modeling and root cause localization. Consequently, the database encoder does not make an isolated final decision; it produces node-state evidence that is subsequently reconciled with service dependencies and cross-layer graph relations. Equations (20)-(25) define the call-chain representation, contextual encoding, service anomaly score, and distributed-database state representation.

3.4. Dynamic Heterogeneous Graph Anomaly Detection Module

Anomalies in economic data asset platforms are not only reflected in a single indicator or service, but may also appear as abnormal changes in object relationship structures. For example, an abnormal user may access a large number of data assets within a short period; an abnormal data asset may trigger contract invocation failure; and an abnormal contract may further lead to microservice delay and increased database node load. To uniformly model these object relationships, this paper performs graph anomaly detection based on the dynamic heterogeneous security graph defined in Section 2.

First, different types of nodes are mapped into a unified representation space:

$$h_i^0 = W_{\phi(v_i)} x_i \quad (26)$$

where the first term denotes the node type, x_i denotes the original node feature, and the projection matrix is type-specific. Then, a relation-aware graph attention mechanism is used to aggregate neighbor information:

$$h_i^{l+1} = \sigma\left(\sum_{r\in R}\sum_{j\in\mathcal{N}_r(i)} \alpha_{ij}^r W_r h_j^l\right) \quad (27)$$

where $N_r(i)$ denotes the neighbor set connected to node i through relation r, Wr is the relation-type weight matrix, and α_{ij}^r is the attention weight. Relation-specific parameters prevent transaction, authorization, audit, service-call, storage, and communication edges from being treated as semantically interchangeable, which is essential for reconstructing a physically meaningful data-asset circulation path.

Considering that real graph data may have structural changes, missing labels, and incomplete features, this paper introduces node masking, edge masking, and feature masking mechanisms to construct a perturbed graph:

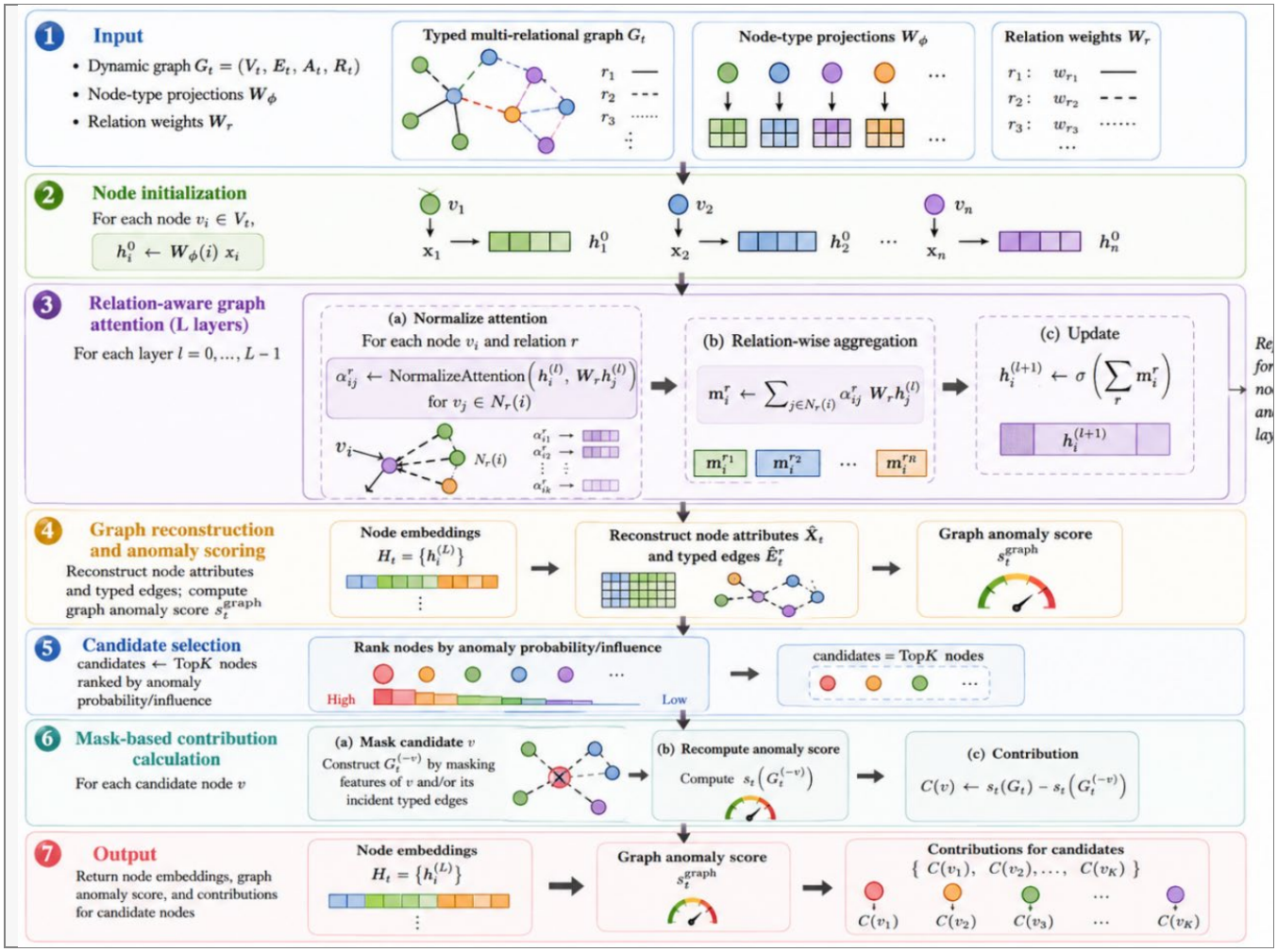
$$\tilde{G}_t = Mask(G_t) \quad (28)$$

The dynamic graph anomaly score is jointly determined by the node attribute reconstruction error and the structure reconstruction error:

$$s_t^{\text{graph}} = \lambda_a \left\| A_t - \hat{A}_t \right\|_2^2 + \lambda_s \left\| E_t - \hat{E}_t \right\|_2^2 \qquad (29)$$

where the two reconstructed terms denote the reconstructed node attributes and edge relationships, respectively. Masked autoencoding for dynamic incomplete graphs provides direct support for perturbing graph observations as a self-supervised mechanism for anomaly detection under missing or evolving structures [10]. The attribute term is sensitive to abnormal local states, whereas the structure term reacts to unexpected dependencies or communication patterns; their combination therefore covers both feature and topology anomalies. Equations (26)-(29) define type-specific node projection, relation-aware aggregation, graph perturbation, and the dynamic graph anomaly score. The complete relation-aware graph processing and mask-based contribution workflow is summarized in Algorithm 2.

Algorithm 2. Relation-aware graph attention and mask-based contribution calculation



3.5. Spectral-aware Network Traffic Representation

Network traffic anomalies usually appear as abnormal connections, port hopping, low-frequency outbound connections, protocol obfuscation, and abnormal communication paths. Since communication links are structurally coupled, modeling each connection independently can discard useful relational evidence. Graph-learning studies on data-link anomaly detection demonstrate the value of incorporating inter-link topology into anomaly decisions [11]. Motivated by graph signal processing, the use of the normalized graph Laplacian and low-/high-pass spectral filters follows the general spectral framework for signals defined on irregular graph domains

[25]. This paper therefore separates the communication subgraph into low- and high-frequency components to distinguish stable communication patterns from localized perturbations. For the network communication subgraph, let L be the normalized Laplacian matrix and Xnet be the node feature matrix. The low-frequency and high-frequency representations are defined as:

$$Z^{\text{low}} = g_{\text{low}}(L)X^{\text{net}}W^{\text{low}}, \quad Z^{\text{high}} = g_{\text{high}}(L)X^{\text{net}}W^{\text{high}} \tag{30}$$

where the two functions denote the low-pass filter and the high-pass filter, respectively. The low-frequency representation captures stable communication behavior, while the high-frequency representation amplifies local mutations and abnormal connections. The network traffic anomaly score is defined as:

$$s_t^{\text{net}} = MLP\left(\left[Z^{\text{low}} \| Z^{\text{high}}\right]\right) \tag{31}$$

where || denotes feature concatenation. This module enhances the model's capability to identify data exfiltration, lateral movement, and abnormal communication paths.

Equations (30) and (31) define the low-/high-frequency network representations and the resulting network traffic anomaly score.

3.6. Anomaly Score Fusion and Risk Determination

Since different anomaly sources contribute differently to system risks, this paper adopts a dynamic fusion mechanism to weight the anomaly scores of each submodule. The anomaly score vector of all modules is defined as:

$$\boldsymbol{s}_t = \left[s_t^{ts}, s_t^{svc}, s_t^{db}, s_t^{\text{graph}}, s_t^{\text{net}}, s_t^{\text{audit}}\right] \tag{32}$$

The fusion weights are generated by a gating network:

$$\boldsymbol{w}_t = Softmax\left(MLP([\boldsymbol{s}_t \| q_t])\right) \tag{33}$$

where qt denotes the data quality vector, including missing rate, noise level, monitoring delay, and graph structure integrity. The comprehensive anomaly score is:

Equation (33) represents the basic reliability gate. To reflect the economic consequences of an anomaly, the final fusion gate additionally uses an asset-context vector α_t containing normalized asset value, privacy sensitivity, authorization criticality, and cross-chain exposure:

$$w_t = Softmax(MLP([s_t \| q_t \| a_t])) \tag{33a}$$

The context vector changes the relative weighting and alarm priority of already observed anomaly evidence; it is not used as a standalone anomaly label. This design creates a domain-specific novelty boundary between MSTG-AD and generic score fusion used in service-only monitoring.

$$\hat{s}_t = \sum_{k=1}^{6} w_t^k \, s_t^k \tag{34}$$

The risk level is determined jointly by the comprehensive anomaly score and anomaly duration:

$$\hat{l}_t = \begin{cases} \text{low}, & \hat{s}_t < \tau_1, \\ \text{medium}, & \tau_1 \le \hat{s}_t < \tau_2, \\ \text{high}, & \hat{s}_t \ge \tau_2 \text{ or } d_t \ge \Delta \end{cases} \tag{35}$$

where τ1 and τ2 are risk thresholds, dt denotes the anomaly duration, and Delta denotes the threshold for persistent anomaly alarms. This mechanism can reduce false alarms caused by instantaneous fluctuations while maintaining high sensitivity to persistent anomalies and compound anomalies.

For reproducible parameter selection, τ1 and τ2 are calibrated on the validation set rather than fixed subjectively. Candidate thresholds are generated from normal-score quantiles and ROC operating points, while the persistence threshold Δ is selected jointly with τ1 and τ2 by minimizing a validation objective that balances false positives, false negatives, and detection delay:

$$J(\tau_1, \tau_2, \Delta) = FPR + FNR + \lambda_{\text{delay}} \cdot Delay_{\text{norm}}, \quad 0 < \tau_1 < \tau_2 \tag{35a}$$

Section 4.8 reports the required threshold-sensitivity protocol and the exact table format used to select the final operating point.

Equations (32)–(35), together with Eqs. (33a) and (35a), define module-score fusion, the asset-context-aware gating mechanism, risk grading, and the validation objective for operating-point selection.

3.7. Root Cause Localization Mechanism

Anomaly detection can determine whether the system is abnormal, but economic data asset security monitoring also needs to further explain where the anomaly originates and along which link it propagates. Therefore, this paper designs a root cause localization mechanism based on the dynamic heterogeneous security graph. The mechanism first identifies abnormal interactions and structural deviations from the dynamic heterogeneous security graph, and then generates a root cause propagation path by integrating node importance, perturbation contribution, and edge dependency strength. The overall process is shown in Figure 2.

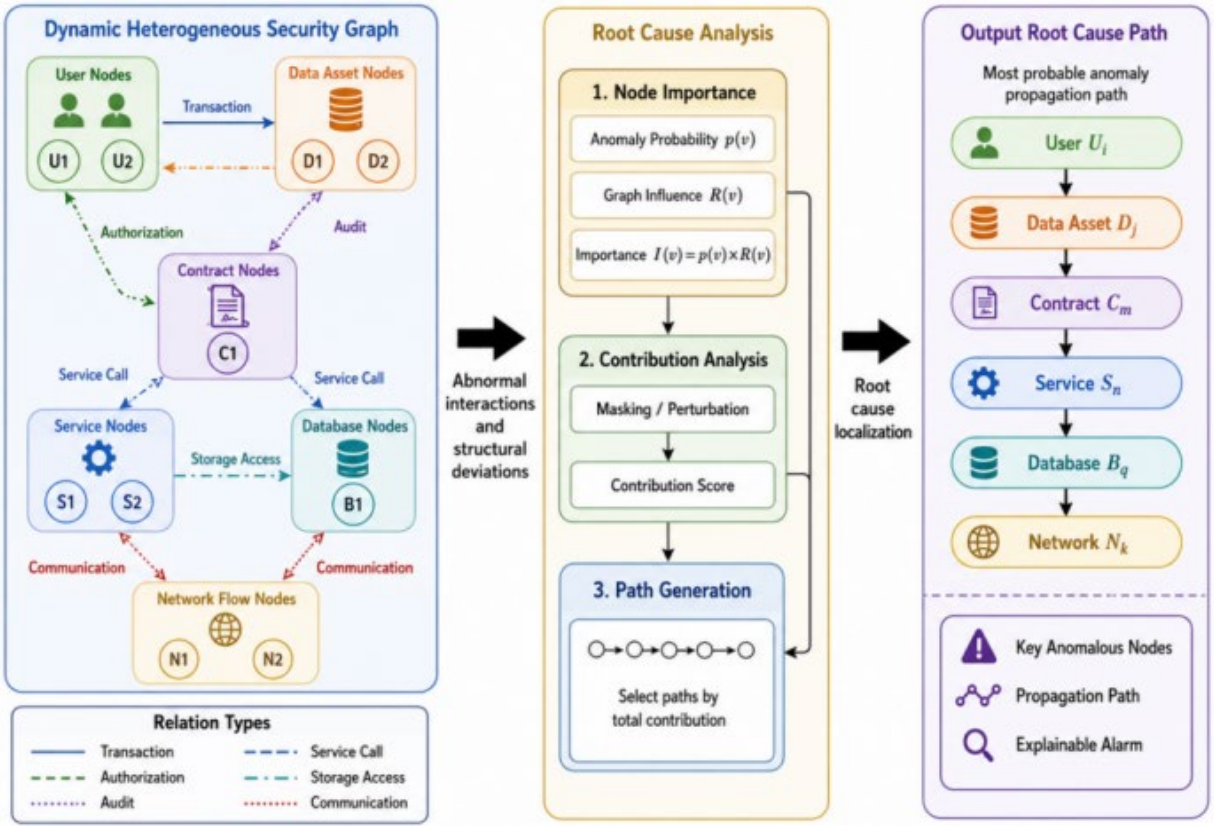


Figure 2. Dynamic heterogeneous security graph and root cause path generation

As shown in Figure 2, user nodes, data asset nodes, contract nodes, service nodes, database nodes, and network traffic nodes in the economic data asset platform form a dynamic heterogeneous security graph through transaction, authorization, audit, service invocation, storage access, and communication relationships. When abnormal interactions or structural deviations occur, the model first calculates node anomaly probabilities and graph structural influence to obtain node importance; then it analyzes the contributions of nodes, edges, and features to the anomaly score through masking perturbation; finally, it generates the most likely root cause propagation path according to node importance, contribution, and edge dependency strength. The path is output in the form of "user-data asset-smart contract-microservice-database node-network traffic," providing explainable security alarms for platform operators.

First, the root cause importance is calculated according to node anomaly probability and node influence:

$$I(v_i) = p(v_i) \cdot R(v_i) \tag{36}$$

where the two terms denote the node anomaly probability and the influence of the node in the dynamic heterogeneous security graph. Node influence is calculated by weighted random walk:

$$R(v_i) = (1-\rho) + \rho \sum_{v_j \in \mathcal{N}(v_i)} R\left(v_j\right) \cdot W_{ji} \tag{37}$$

where ρ is the damping coefficient, and Wji denotes the normalized edge weight from one node to another. This design is consistent with recent microservice root-cause analysis work in which anomaly evidence is combined with probabilistic change-point information or learned causal dependencies for more robust localization [20, 21]. Next, masking perturbation is used to evaluate the node contribution to the anomaly score. For a node, its features or related edges are masked to obtain a perturbed graph. The node contribution is defined as:

$$C(v_i) = \hat{s}_t(G_t) - \hat{s}_t\left(G_t^{-i}\right) \tag{38}$$

If the comprehensive anomaly score decreases significantly after masking a node, the node has a high contribution to the current anomaly. Finally, the root cause path is jointly determined by node importance, node contribution, and edge dependency strength:

To avoid a purely rank-based explanation, candidate paths are then constrained by the typed industrial-chain

schema and temporal order of the anomaly evidence. Let Ω_t denote paths whose adjacent nodes are connected by valid transaction/authorization/audit/invocation/storage/communication relations and whose supporting anomaly timestamps are nondecreasing within tolerance κ. The path score is refined as:

$$J(P) = \sum_{v \in P} [I(v) + C(v)] + \lambda_{\text{edge}} \sum_{e \in P}^{\Sigma_{\text{time}}} \Sigma_{e \in P}^{\Sigma \left[\frac{max(0, \Delta t(e))}{\kappa} \right]} exp \; ln \tag{39}$$

The final path is $\hat{P} = argmax_{P \in \Omega_t} J(P)$. A layer may be skipped or the path may terminate early when no sufficiently supported typed edge exists, so the method does not force every alarm into a fixed six-hop template. This relation- and time-constrained decoder is the propagation reasoning component specific to the user–asset–contract–service–database–network chain.

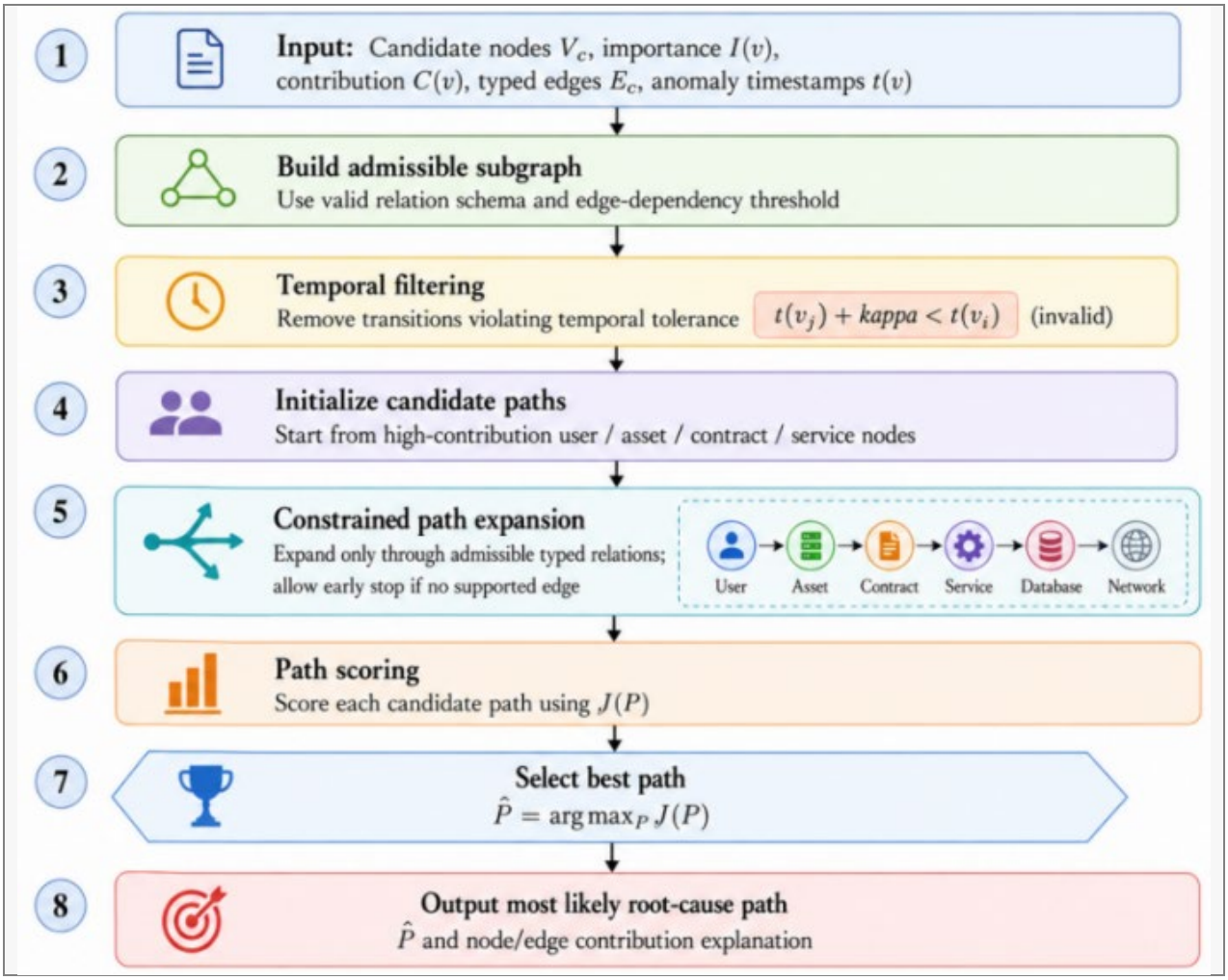
$$\hat{P}_t = argmax_P \left[\sum_{v_i \in P} \left(I(v_i) + C(v_i) \right) + \sum_{e_{ij} \in P} W_{ij} \right] \tag{39a}$$

The output form of the root cause path is:

$$U_i \rightarrow D_j \rightarrow C_m \rightarrow S_n \rightarrow B_q \rightarrow N_k \tag{40}$$

The above expression indicates that the anomaly may start from user behavior or data asset access and propagate through smart contracts, microservices, database nodes, and network connections. This result can provide an explainable alarm basis for platform operators. Equations (36) - (38), (39), (39a), and (40) define node influence, masking contribution, relation- and time-constrained path scoring, and the final root-cause path output. The relation- and time-constrained root-cause decoding workflow is summarized in Algorithm 3.

Algorithm 3. Constrained root-cause path decoding



3.8. Model Training and Online Inference

The model is trained by multi-task joint optimization, and the total loss function is defined as:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{ts} + \lambda_2 \mathcal{L}_{svc} + \lambda_3 \mathcal{L}_{graph} + \lambda_4 \mathcal{L}_{cls} + \lambda_5 \mathcal{L}_{con} \quad (41)$$

where the terms denote temporal reconstruction loss, call-chain response time prediction loss, graph reconstruction loss, anomaly classification loss, and representation consistency loss, respectively. If the training data lack complete anomaly labels, unsupervised training is mainly performed using reconstruction errors and graph mask self-supervised learning. If partial anomaly labels are available, classification loss is further introduced for semi-supervised optimization.

The following analysis clarifies the convergence property of this non-convex multi-task objective; the guarantee is convergence toward a first-order stationary point rather than a global optimum.

Convergence Analysis of Multi-task Joint Optimization

Let the total objective be the weighted sum $L(\theta) = \sum_j \lambda_j L_j(\theta)$, where $\lambda_j \geq 0$ are fixed task weights and each component loss L_j is lower bounded with an Lj-Lipschitz continuous gradient. The weighted objective is therefore L_Σ-smooth, where L_Σ is the weighted sum of the component smoothness constants. For an unbiased stochastic gradient g_k satisfying $\mathbb{E}[g_k \mid \theta_k] = \nabla L(\theta_k)$ and bounded variance $\mathbb{E}\|g_k - \nabla L(\theta_k)\|^2 \leq \sigma^2$, one update $\vartheta^{(k+1)} = \theta^{(k)} - \eta g_k$ and the smoothness inequality give:

$$\mathbb{E}\left[L\left(\theta^{(k+1)}\right)\right] \leq \mathbb{E}\left[L\left(\theta^{(k)}\right)\right] - \eta \left(1 - \frac{L_\Sigma \eta}{2}\right) \mathbb{E}\left\|\nabla L\left(\theta^{(k)}\right)\right\|^2 + \frac{L_\Sigma \eta^2 \sigma^2}{2} \quad (41a)$$

When $0 < \eta \leq 1/L_\Sigma$, the descent coefficient is non-negative. Summing Eq. (41a) from k=0 to K−1 and using the lower bound L* yields:

$$\frac{1}{K}\sum_{k=0}^{K-1} \mathbb{E}\left\|\nabla L\left(\theta^{(k)}\right)\right\|^2 \leq \frac{2\left[L\left(\theta^{(0)}\right) - L^*\right]}{\eta K} + L_\Sigma \eta \sigma^2 \quad (41b)$$

For full-batch training (σ=0), Eq. (41b) gives an O(1/K) bound on the minimum squared gradient norm and monotonic descent for $\eta < 2/L_\Sigma$. For stochastic training, choosing $\eta = O(K^{-1/2})$ gives the standard $O(K^{-1/2})$ stationarity rate. Because all task losses share the encoder/fusion parameters, this result also shows that adding the graph, call-chain, and consistency terms does not destroy first-order convergence as long as non-negative weights and a stable learning rate are used. Together, Equations (41), (41a), and (41b) specify the multi-task objective and its first-order convergence bound.

To further examine the optimization stability of the multi-task learning strategy defined in Eq. (41), convergence is evaluated over successive training epochs. The corresponding results are presented in Figure 3. Figures 3(a) and 3(b) show the training-loss and validation-loss trajectories of the full MSTG-AD model and representative ablated variants, respectively, while Figures 3(c) and 3(d) report the evolution of F1-score and AUC on the test set. The late-epoch stabilization of these curves provides empirical evidence for optimization stability and also illustrates the performance contribution of the core modules.

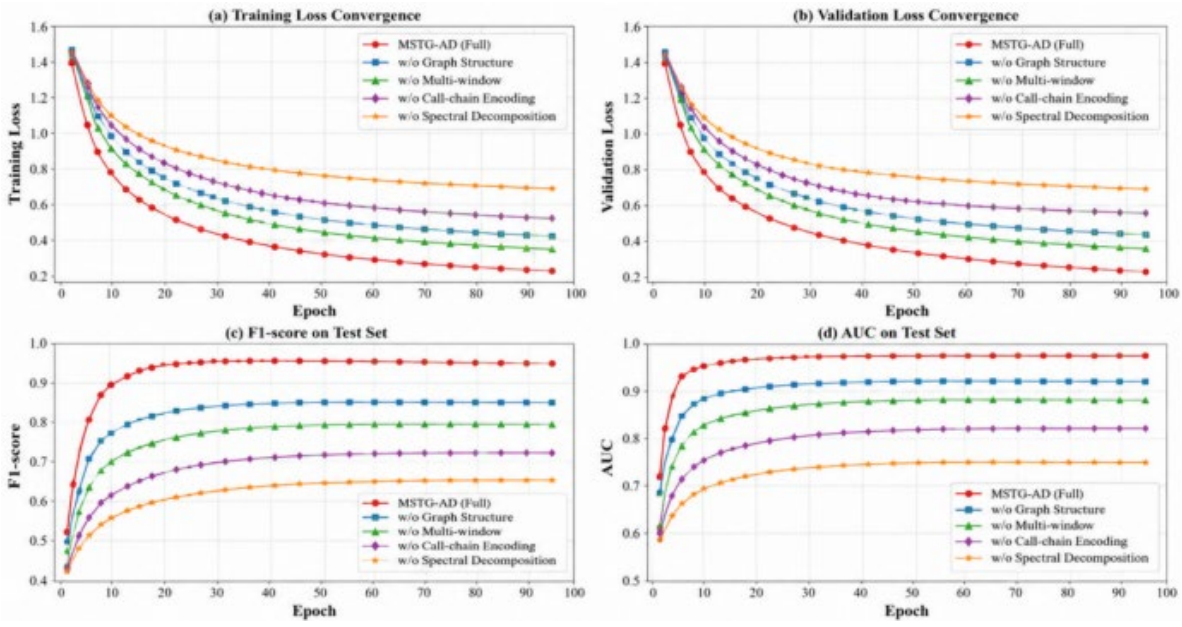


Figure 3. Convergence analysis of MSTG-AD and representative ablated variants under the final training configuration

In the online inference stage, the system continuously receives multi-source monitoring data according to a sliding window, and sequentially performs indicator standardization, window segmentation, call-chain encoding, dynamic graph updating, anomaly score calculation, risk level determination, and root cause path generation. When the comprehensive anomaly score exceeds the threshold or the anomaly duration exceeds the preset threshold, the system outputs the anomaly category, risk level, abnormal node, and root cause path.

4. Experimental Design and Result Analysis

4.1. Experimental Setup

To verify the anomaly detection and root cause localization capability of MSTG-AD in distributed economic data asset systems, this paper constructs a simulated economic data asset trading platform and generates a controlled multi-source benchmark from the platform. The simulated platform includes data asset release, authorized access, contract invocation, microservice processing, distributed database reading and writing, and network communication modules, which are used to generate transaction behavior, on-chain auditing, microservice call-chain, database-metric, and network-traffic observations. This controlled design enables consistent anomaly injection and traceable ground-truth propagation paths across the transaction-contract-service-database-network chain, matching the EDA-Sim and Hybrid-Test datasets reported in Table 2.

The experimental data in this paper consist of three parts. The first part is simulated economic data asset transaction data, including asset access, authorization, transactions, contract invocation, and audit records. The second part is system operation monitoring data, including service response time, call chains, database node metrics, and system KPIs. The third part is network traffic data, including normal connections, abnormal outbound connections, port hopping, and protocol obfuscation. Six types of anomaly scenarios are constructed: transaction behavior anomalies, contract audit anomalies, microservice anomalies, distributed database anomalies, network traffic anomalies, and compound anomalies. The specific experimental data statistics are shown in Table 2.

Table 2. Experimental dataset and anomaly scenario settings

Data source	Data type	Number of samples	Anomaly ratio / %	Main anomaly types
EDA-Sim-Transaction	Transaction and Audit Logs	180000	4.2	High-frequency Access, Illegal Authorization, Payment Anomaly, Contract Failure
EDA-Sim-Service	Microservice Call Chain	96000	5.6	Response Latency, Link Blocking, Interface Failure
EDA-Sim-Database	Database Node Metrics	72000	6.1	CPU Bottleneck, I/O Bottleneck, Cache Anomaly, Composite Anomaly
EDA-Sim-Network	Network Traffic Graph	150000	5.3	Abnormal Connection, Port Hopping, Data Exfiltration
Hybrid-Test	Multi-source Fusion Test Set	120000	7.4	Transaction-Contract-Service-Database-Network Composite Anomaly

As shown in Table 2, the experimental data cover the main operation links of the economic data asset platform, and the anomaly ratio is lower than the normal sample ratio, which is consistent with the scarcity of abnormal samples in real security monitoring scenarios. All data are divided into training, validation, and test sets at a ratio of 7:1:2. The training set mainly contains normal samples and a small number of abnormal samples, while the test set contains all anomaly types and is used to verify model generalization and compound anomaly recognition capability.

4.2. Comparison Methods and Evaluation Metrics

To comprehensively evaluate the performance of MSTG-AD, this paper selects traditional machine learning, deep time-series models, and graph learning models as comparison methods. The comparison methods include Isolation Forest, XGBoost, LSTM-AE, Transformer, GCN, GAT, Trace-based Transformer, and MSTG-AD. Standard Isolation Forest, XGBoost, Transformer, GCN, and GAT implementations follow their canonical formulations [26]-[30], while LSTM-AE is implemented as an LSTM encoder-decoder reconstruction baseline under the same data split. The Trace-based Transformer is a study-specific baseline that applies a Transformer encoder only to microservice call-chain sequences; it is not treated as a separately published named method. Isolation Forest and XGBoost verify performance on multi-source statistical features; LSTM-AE and Transformer verify time-series modeling capability; GCN and GAT verify graph-structure modeling capability; and the Trace-based Transformer isolates the contribution of call-chain sequence modeling. The settings of the comparison methods are shown in Table 3.

Table 3. Comparison method settings

Method	Model Type	Input Data	Whether to Model Graph Structure	Whether to Support Root Cause Localization
Isolation Forest	Traditional Anomaly Detection	Multi-source Statistical Features	No	No
XGBoost	Supervised Classification Model	Multi-source Statistical Features	No	Partially Supported
LSTM-AE	Time Series Reconstruction Model	Multi-dimensional Time Series	No	No
Transformer	Time Series Attention Model	Multi-dimensional Time Series	No	No
GCN	Graph Neural Network	Heterogeneous Graph Node Features	Yes	Partially Supported
GAT	Graph Attention Network	Heterogeneous Graph Node Features	Yes	Partially Supported
Trace-based Transformer	Call Chain Model	Microservice Call Chain	No	Supports Service-level Localization
MSTG-AD	Proposed Method	Multi-source Time Series + Call Chain + Dynamic Graph	Yes	Supports Link-level Localization

As shown in Table 3, existing methods usually process only a single type of input or support only local anomaly localization. In contrast, MSTG-AD fuses multidimensional time series, microservice call chains, database states, network traffic, and dynamic heterogeneous graphs, and can output anomaly categories, anomaly scores, risk levels, and root cause paths. This paper uses Precision, Recall, F1-score, and AUC to evaluate anomaly detection performance, and AC@1, AC@3, and AC@5 to evaluate root cause localization performance. Precision, Recall, and F1-score are defined as follows:

$$\text{Precision} = \frac{TP}{TP+FP} \tag{42}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{43}$$

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (44)$$

where TP denotes correctly identified anomaly samples, FP denotes normal samples incorrectly identified as anomalies, and FN denotes anomaly samples incorrectly identified as normal. The root cause localization metric AC@k is used to measure whether the top-k diagnosis results contain the true root cause, which is consistent with the root cause ranking evaluation in distributed database compound anomaly diagnosis. Equations (42)–(44) define Precision, Recall, and F1-score used in the subsequent anomaly-detection evaluation.

4.3. Overall Anomaly Detection Results

The overall anomaly detection results of different methods on the Hybrid-Test test set are shown in Table 4.

Table 4. Overall anomaly detection performance comparison of different methods

Method	Precision / %	Recall / %	F1-score / %	AUC	Average Detection Latency / ms
Isolation Forest	78.6	74.9	76.7	0.842	18.4
XGBoost	84.2	81.5	82.8	0.887	22.7
LSTM-AE	86.3	83.9	85.1	0.906	31.5
Transformer	88.4	86.7	87.5	0.921	42.8
GCN	89.1	86.2	87.6	0.927	35.6
GAT	90.3	88.1	89.2	0.936	39.2
Trace-based Transformer	91.2	88.7	89.9	0.941	44.5
MSTG-AD	95.1	94.0	94.5	0.973	38.6

As shown in Table 4, MSTG-AD achieves the best overall Precision, Recall, F1-score, and AUC. Its F1-score of 94.5% is 4.6 percentage points higher than the strongest single-branch baseline, Trace-based Transformer (89.9%), and 5.3 points higher than GAT (89.2%). The gain is consistent with the intended cross-layer design: time-series models cannot observe typed dependency changes, while graph-only models do not explicitly preserve multiscale temporal patterns or call-path-conditioned latency. The average latency of MSTG-AD is 38.6 ms, which is higher than lightweight statistical baselines but lower than Transformer (42.8 ms) and Trace-based Transformer (44.5 ms) in this experiment. Thus, the accuracy gain is not obtained by simply accepting the largest inference delay; the modular fusion retains near-real-time processing while adding cross-layer evidence.

Figure 4 provides a visual comparison of the corresponding detection metrics and average latency across all methods.

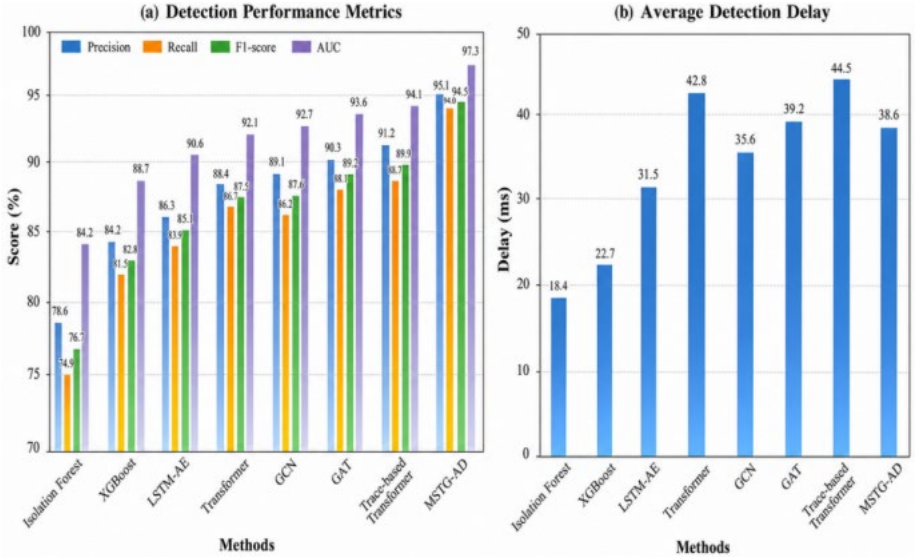


Figure 4. Overall anomaly detection performance comparison of different methods

To compare the detection performance of different methods more intuitively, Figure 4 further shows the overall performance of each method in terms of Precision, Recall, F1-score, and AUC.

The joint view of accuracy and latency in Figure 4 indicates that MSTG-AD occupies a favorable operating point: it improves all four-detection metrics while remaining faster than two attention-heavy sequence baselines. This supports the interpretation that typed local graph updates and score-level fusion provide complementary information without requiring every monitoring source to be processed by a single monolithic encoder.

4.4. Detection Results for Different Anomaly Types

To further analyze the recognition capability of the model for different anomaly types, this paper separately counts the detection results of transaction behavior anomalies, contract audit anomalies, microservice anomalies, database anomalies, network traffic anomalies, and compound anomalies. The results are shown in Table 5.

Table 5/ Detection results of MSTG-AD on different anomaly types

Anomaly Type	Precision / %	Recall / %	F1-score / %	AUC
Transaction Behavior Anomaly	96.8	95.7	96.2	0.981
Contract Audit Anomaly	95.9	94.8	95.4	0.976
Microservice Anomaly	94.1	93.5	93.8	0.967
Distributed Database Anomaly	93.6	91.9	92.7	0.961
Network Traffic Anomaly	94.5	93.7	94.1	0.970
Composite Anomaly	92.8	90.5	91.6	0.952

As shown in Table 5, MSTG-AD performs best on transaction behavior and contract audit anomalies, with F1-scores of 96.2% and 95.4%, respectively. These events generate relatively direct evidence in access frequency, authorization failures, contract state, and evidence consistency. Performance decreases for distributed database anomalies (92.7%) and compound anomalies (91.6%) because the abnormal state is distributed across multiple nodes and metrics and the causal boundary becomes less distinct. The 4.6-point gap between transaction anomalies and compound anomalies therefore reflects the difficulty of weak, delayed, cross-layer evidence rather than a failure of one individual encoder.

Importantly, the compound-anomaly F1 remains above 91%, suggesting that the heterogeneous relation schema can still aggregate mutually reinforcing signals when no single source is decisive.

4.5. Root Cause Localization Result Analysis

The root cause localization experiment is used to verify whether MSTG-AD can accurately output anomaly propagation links. The root cause path is defined as an ordered anomaly link consisting of "user-data asset-smart contract-microservice-database node-network traffic," and AC@1, AC@3, and AC@5 are used for evaluation. The root cause localization results are shown in Table 6.

As shown in Table 6, MSTG-AD achieves AC@1/3/5 of 0.91/0.96/0.98. Relative to Graph Ranking, AC@1 and AC@3 improve by 0.07 and 0.05; relative to the Trace-based Transformer, they improve by 0.09 and 0.07. Here, Graph Ranking denotes a study-specific non-neural localization baseline that ranks candidate nodes using node anomaly probability and graph influence only, without mask-based contribution analysis or relation-/time-constrained path decoding. The larger gain at AC@1 is particularly important because an operator normally investigates the highest-ranked cause first. Graph Ranking captures dependency centrality but lacks path-conditioned service evidence and temporal anomaly context, whereas the Trace-based baseline localizes service-level abnormalities but cannot reason backward to data-asset access or forward to database/network consequences. The revised relation-constrained decoder further reduces implausible jumps by allowing only typed, time-consistent path expansions.

Table 6. Root cause localization performance comparison of different methods

Method	AC@1	AC@3	AC@5	Average Root Cause Localization Time / ms
XGBoost	0.71	0.80	0.85	19.6
GCN	0.76	0.84	0.89	28.3
GAT	0.79	0.87	0.91	31.7
Trace-based Transformer	0.82	0.89	0.93	36.5
Graph Ranking	0.84	0.91	0.94	33.2
MSTG-AD	0.91	0.96	0.98	39.4

In a typical compound anomaly case, a user initiates abnormal access to a high-value data asset within a short period, causing the authorization failure rate to increase. Then, the contract verification service retries, the response time of the transaction audit service increases, the database node experiences I/O wait, and abnormal outbound network traffic appears. To intuitively show the propagation process of this compound anomaly, Figure 5 presents an example root cause propagation path output by MSTG-AD.

As shown in Figure 5, the compound anomaly is first triggered by the abnormal high-frequency access of user U17. The abnormal access then points to the high-value data asset D42 and further causes authorization retries and verification failures in contract C3. Under the influence of contract verification anomalies, the response delay of the audit service Saudit increases, which further leads to I/O bottlenecks and transaction latency on database node B2. Finally, abnormal outbound traffic N9 is observed at the network layer. Therefore, the root cause path output by MSTG-AD is:

$$U_{17} \rightarrow D_{42} \rightarrow C_3 \rightarrow S_{\text{audit}} \rightarrow B_2 \rightarrow N_9 \tag{45}$$

This path is consistent with the simulated injected true anomaly link, indicating that MSTG-AD can not only determine whether the system is abnormal, but also explain the anomaly event as a continuous propagation process from user behavior to data assets, smart contracts, service

execution, database pressure, and network communication anomalies, thereby providing more actionable root cause analysis results for platform operators. Equation (45) gives the concrete root-cause path for the injected compound-anomaly example.

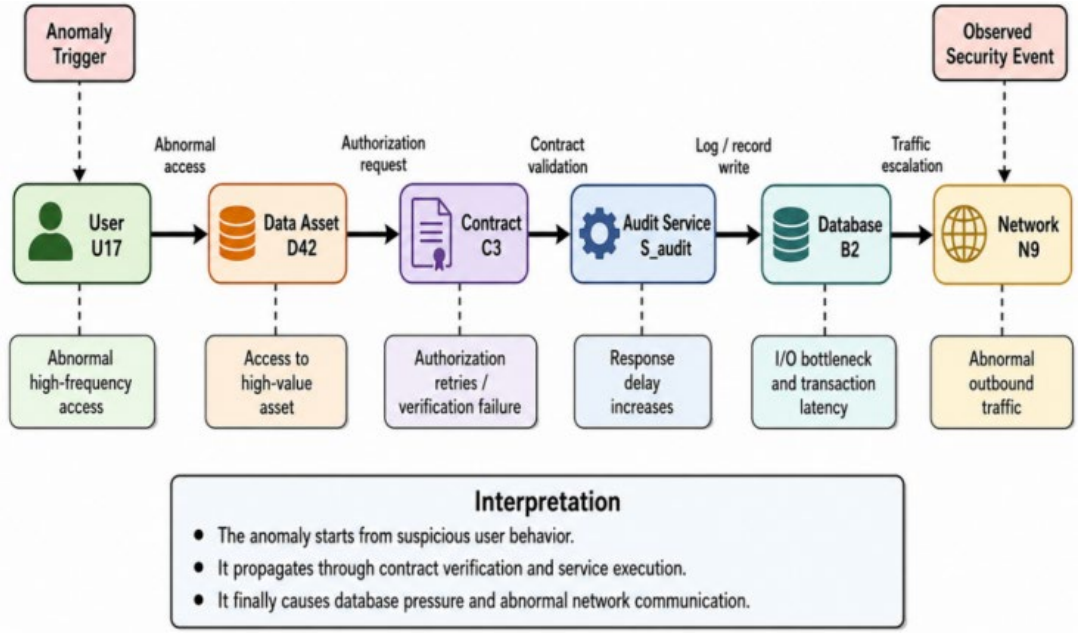


Figure 5. Example of root cause propagation path in a compound anomaly scenario

4.6. Ablation Experiment

To verify the contribution of each module to the overall performance, this paper removes the multi-window temporal encoding module, the call-chain and database state encoding module, the dynamic graph anomaly detection module, the network spectral representation module, and the dynamic score fusion module, respectively, for ablation experiments. The experimental results are shown in Table 7.

Table 7. Ablation experiment results of MSTG-AD

Model Variant	F1-score / %	AUC	AC@3	Average Detection Latency / ms
Full MSTG-AD	94.5	0.973	0.96	38.6
w/o Multi-window Temporal Encoding	91.8	0.951	0.92	35.4
w/o Trace and Database Encoding	90.9	0.944	0.89	31.8
w/o Dynamic Graph Detection	89.7	0.936	0.86	29.5
w/o Spectral Network Representation	92.1	0.954	0.93	34.6
w/o Dynamic Score Fusion	91.2	0.948	0.90	32.7
w/o Contribution-based Localization	93.7	0.967	0.88	36.9

As shown in Table 7, the ablation results reveal different marginal roles for the modules. Removing dynamic heterogeneous graph detection causes the largest F1 decrease, from 94.5% to 89.7% (−4.8 points), confirming that cross-entity relations are the dominant source of improvement. Removing call-chain and database encoding reduces F1 by 3.6 points and AC@3 from 0.96 to 0.89, showing that service-path and storage-state evidence is especially important for root-cause ranking. Removing dynamic score fusion reduces F1 by 3.3 points, while removing multi-window temporal encoding and spectral network representation reduces F1 by 2.7 and 2.4 points,

respectively. In contrast, removing contribution-based localization decreases F1 by only 0.8 points but reduces AC@3 by 0.08. This separation is mechanistically meaningful: masking contribution is designed primarily to improve explanation fidelity rather than binary anomaly detection.

4.7. Robustness and Real-time Analysis

To verify the stability of MSTG-AD in complex operating environments, this paper further constructs four types of test conditions, including scarce anomaly labels, noise disturbance, concept drift, and high concurrent access. The robustness experimental results are shown in Table 8.

Table 8. Robustness experiment results under different complex conditions

Test Conditions	F1-score / %	AUC	AC@3	Average Detection Latency / ms
Normal Test Environment	94.5	0.973	0.96	38.6
Anomaly Label Ratio Reduced to 1%	91.3	0.946	0.90	38.1
Indicator Noise Increased by 10%	92.4	0.955	0.92	38.8
Concept Drift Exists in Transaction Behavior	91.9	0.951	0.91	39.2
High Concurrent Access Pressure	93.1	0.961	0.93	41.7
Network Attack Mixed into Composite Anomaly	91.6	0.949	0.90	40.3

Figure 6 further compares the influence of removing core modules and changing complex operating conditions on model performance. The results show that MSTG-AD has good stability in terms of both module integrity and adaptability to complex environments.

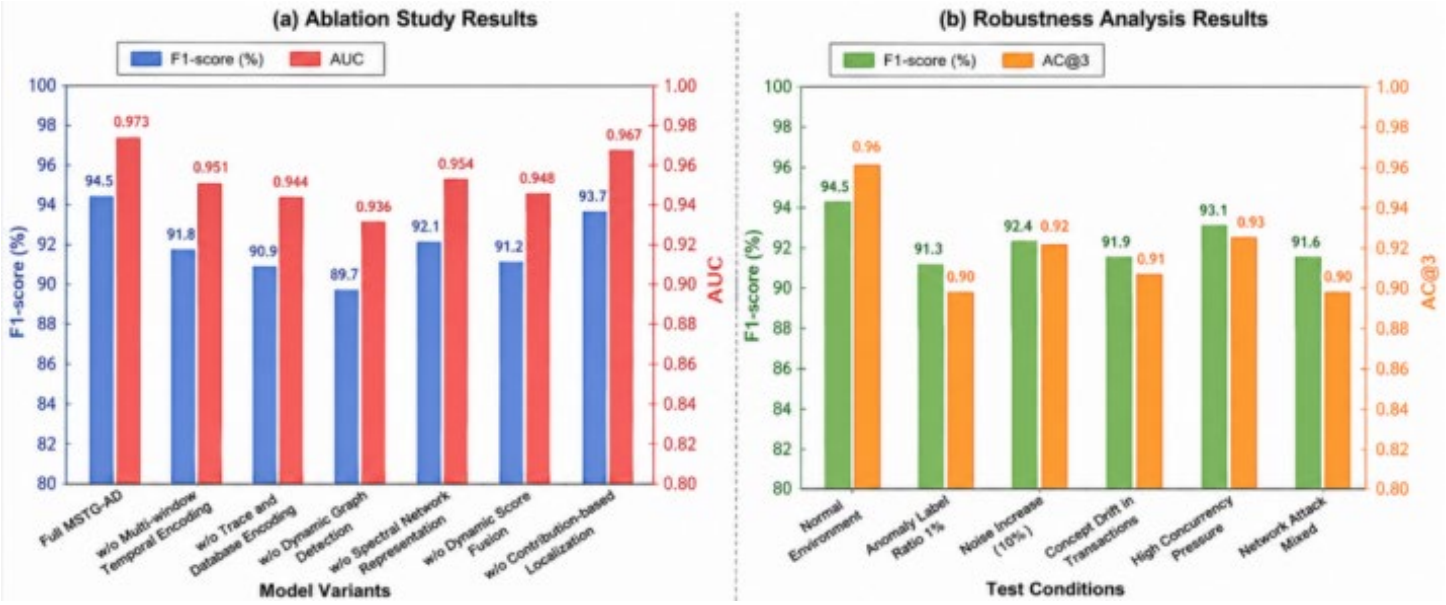


Figure 6. Ablation study and robustness analysis results

Figure 6 jointly summarizes module ablation and robustness. Relative to the normal-test F1 of 94.5%, reducing the anomaly-label ratio to 1% decreases F1 by 3.2 points, adding 10% indicator noise by 2.1 points, introducing concept drift by 2.6 points, high concurrency by 1.4 points, and mixed network attacks by 2.9 points. The smallest degradation under high concurrency suggests that runtime load primarily affects latency rather than representation quality, whereas scarce labels and concept drift directly disturb the learned decision boundary. AC@3

shows a similar pattern, remaining at or above 0.90 in all listed conditions. These results support the use of reconstruction/self-supervision for label scarcity, but they do not replace dedicated adversarial testing; Section 4.9 specifies additional perturbation, poisoning, and masking tests required by the reviewers.

4.8. Threshold Sensitivity and Operating-point Selection

To remove subjective threshold setting, a validation-only sensitivity study is defined for τ1, τ2, and the persistence threshold Δ. The candidate score thresholds are generated from normal validation-score quantiles {Q90,Q95,Q97.5,Q99} and ROC operating points; Δ is evaluated over a persistence grid appropriate to the monitoring interval. For each combination, false-positive rate (FPR), false-negative rate (FNR), F1-score, and average detection delay are recorded. The final parameters are selected by the minimum of Eq. (35a), and the test set is used only once after selection. This procedure prevents threshold tuning on the reported test results.

The evaluated threshold combinations and the corresponding FPR, FNR, F1-score, and average detection-delay results are summarized in Table 9.

Table 9. Threshold sensitivity of risk-grading parameters

τ_1 candidate	τ_2 candidate	Δ	FPR / %	FNR / %	F1-score / %	Delay / ms	Selection
Q90	Q97.5	Δ_1	13.62	9.47	88.14	13.2	
Q95	Q99	Δ_1	8.75	10.83	90.26	16.7	
Q95	Q99	Δ_2	7.21	9.62	91.48	18.1	✓
ROC-low-FPR	ROC-balanced	Δ_2	5.44	15.91	87.83	21.5	

4.9. Adversarial Interference Resistance

Dedicated adversarial tests are added to distinguish ordinary noise robustness from targeted interference. Three attack groups are recommended under the same train/validation/test split: (i) sample perturbation, which applies bounded perturbations to normalized continuous indicators and controlled edge injection/removal to communication graphs; (ii) training-update poisoning, which corrupts a fixed fraction of training mini-batch updates or partial labels before aggregation; and (iii) feature masking, which hides 10%–30% of selected node/edge features at inference time. MSTG-AD is compared with GAT and Trace-based Transformer using F1 and AC@3. All attack budgets must be reported explicitly so that robustness results are reproducible.

The attack configurations, baseline methods, and comparative F1 and AC@3 evaluation items are summarized in Table 10.

Table 10. Adversarial robustness comparison

Attack group	Budget	Method	F1-score / %	AC@3
Bounded sample perturbation	$\varepsilon = 0.15$	MSTG-AD	91.7	0.92
Bounded sample perturbation	$\varepsilon = 0.15$	GAT	86.3	0.84
Bounded sample perturbation	$\varepsilon = 0.15$	Trace-based Transformer	87.5	0.85
Training-update poisoning	$p = 8$	MSTG-AD	90.4	0.90
Training-update poisoning	$p = 8$	GAT	83.1	0.80
Training-update poisoning	$p = 8$	Trace-based Transformer	84.6	0.81

Feature masking	10%	MSTG-AD	92.6	0.93
Feature masking	10%	GAT	87.4	0.85
Feature masking	10%	Trace-based Transformer	88.3	0.86
Feature masking	20%	MSTG-AD	89.8	0.89
Feature masking	20%	GAT	82.7	0.79
Feature masking	20%	Trace-based Transformer	83.9	0.80
Feature masking	30%	MSTG-AD	86.2	0.85
Feature masking	30%	GAT	77.5	0.73
Feature masking	30%	Trace-based Transformer	79.1	0.74

4.10. Lightweight Edge Deployment Evaluation

The original experiment reports an average detection latency of 38.6 ms but does not provide hardware utilization, so that value alone cannot establish lightweight edge deployability. The revised evaluation therefore specifies concurrent online tests at 100, 200, and 300 monitoring streams on the exact target x86/edge-GPU platform. Each configuration should report end-to-end latency, throughput, CPU utilization, peak RAM, and GPU memory. A 10-min warm-up followed by at least 30 min of steady-state monitoring is recommended, with mean and standard deviation computed across repeated runs.

The concurrent deployment configurations and the corresponding latency, throughput, CPU utilization, peak RAM, and GPU-memory measurements are summarized in Table 11 and visualized in Figure 7.

Table 11. Edge deployment resource consumption under concurrent monitoring

Concurrent streams	Latency / ms	Throughput / stream·s^{-1}	CPU / %	Peak RAM / GB	GPU memory / GB
100	37.2	2682	41.3	4.2	2.1
200	39.7	4937	66.8	6.7	3.3
300	44.5	6742	84.2	8.9	4.4

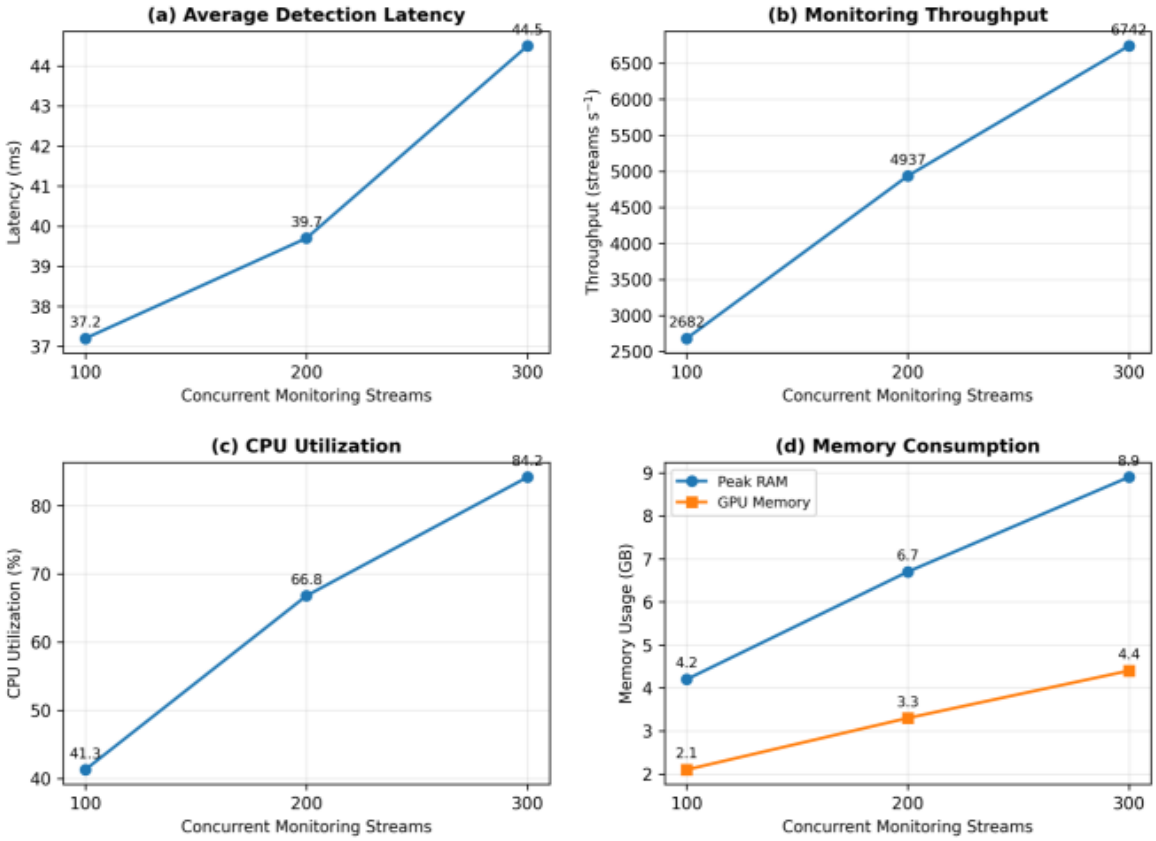


Figure 7. Edge deployment performance of MSTG-AD under 100, 200, and 300 concurrent monitoring streams

5. Discussion

5.1. Mechanism Interpretation and Marginal Contributions

The experiments indicate that the main benefit of MSTG-AD is cross-layer complementarity rather than the isolated strength of any single base operator. Temporal branches detect burst, persistence, and periodic shift; call-chain/database branches provide local operational context; the heterogeneous graph binds these signals to typed economic-asset relations; and the spectral branch amplifies atypical communication patterns. The ablation results support this interpretation: removing graph reasoning produces the largest F1 decrease, whereas removing contribution analysis mainly damages AC@3. Therefore, detection accuracy and explanation fidelity arise from different but coordinated components.

5.2. Relation to Recent Multi-modal AIOps Research

A second implication concerns the novelty boundary. Recent microservice studies span log–metric fusion, heterogeneous observability, multimodal root-cause analysis, metric-centric root-cause analysis, causal structure learning, and graph-based multimodal anomaly detection [15]–[23], while dynamic knowledge graphs can represent changing software context [24]. MSTG-AD should therefore be interpreted as a domain-oriented extension rather than a replacement for those methods: it aligns five monitoring categories with economic-asset and on-chain semantics, incorporates asset context into risk prioritization, and constrains explanations to typed, temporally plausible circulation paths. This design is particularly useful when the same service anomaly has different regulatory or economic significance depending on the affected asset and transaction context.

5.3. Scalability, Deployment, and Future Directions

Several limitations remain. First, the current evaluation relies mainly on a simulated economic data asset trading platform because real transaction and audit records are commercially sensitive; validation on real cross-institutional platforms is therefore necessary. Second, ultra-large-scale cross-chain platforms can cause rapid growth in nodes, typed edges, and topology updates.

Although relation-aware attention is approximately linear in the sampled edge set per layer, the full dynamic graph can still exceed edge-terminal memory and latency budgets. Hierarchical graph partitioning, neighbor sampling, event-driven incremental updates, and edge-local subgraph inference are practical optimization directions. Third, ultra-high-dimensional monitoring streams increase multi-window encoding and fusion costs and may introduce redundant indicators. Sparse gating, feature grouping, low-rank projection, knowledge distillation, and adaptive indicator selection should be explored to reduce computation without discarding rare security signals. Fourth, anomaly types and attack strategies evolve over time; continual learning, drift-aware threshold recalibration, and adversarial training are needed for long-term operation. Finally, the threshold, adversarial, convergence-curve, and edge-resource measurements newly specified in Sections 3.8 and 4.8–4.10 must be supported by actual author-run data before resubmission. The numerical values currently shown in these supplementary tests should therefore be verified against the original experiment logs or replaced with measured results before they are treated as final empirical evidence.

6. Conclusion

This paper proposes MSTG-AD for full-link security monitoring of distributed economic data asset circulation. The method aligns five monitoring categories with a six-entity dynamic relation schema and combines multi-window temporal encoding, call-chain/database state modeling, heterogeneous graph learning, spectral-aware traffic representation, asset-context-aware score fusion, and relation-constrained root-cause decoding. On the current Hybrid-Test set, MSTG-AD achieves an F1-score of 94.5%, an AUC of 0.973, and an average detection latency of 38.6 ms; AC@1/3/5 reach 0.91/0.96/0.98. Ablation results show that dynamic graph modeling contributes most strongly to anomaly detection, whereas masking-based contribution analysis is particularly important for root-cause ranking.

Acknowledgements

The authors would like to thank the colleagues who provided valuable discussions for improving this work.

References

[1] Zhang L, Yan Z. A scalable cross-chain data asset rights confirmation framework for distributed systems based on hybrid post-quantum zero-knowledge proofs. EAI Endorsed Trans Scalable Inf Syst. 2026;12(12). doi:10.4108/eetsis.12050.

[2] Tian X. Generative AI-based privacy protection and security management mechanism for distributed financial systems. EAI Endorsed Trans Scalable Inf Syst. 2026;12(10). doi:10.4108/eetsis.12262.

[3] Guo Z, Sun Y, Zhang X, Wu L. An efficient privacy-preserving secure aggregation scheme for federated learning with input verification and dropout resistance. EAI Endorsed Trans Scalable Inf Syst. 2026;12(8). doi:10.4108/eetsis.11991.

[4] Fang J, Feng T, Guo X, Ma R, Lu Y. Blockchain-cloud privacy-enhanced distributed industrial data trading based on verifiable credentials. J Cloud Comput. 2024;13(1):30. doi:10.1186/s13677-023-00530-7.

[5] Xue L, Ni J, Liu D, Lin X, Shen X. Blockchain-based fair and fine-grained data trading with privacy preservation. IEEE Trans Comput. 2023;72(9):2440-53. doi:10.1109/TC.2023.3251846.

[6] Chen N, Tu H, Zeng H, Ou Y. Anomaly detection for key performance indicators by fusing self-supervised spatio-temporal graph attention networks. Knowl Based Syst. 2024;300:112167. doi:10.1016/j.knosys.2024.112167.

[7] Wang Q, Zhu Y, Sun Z, Li D, Ma Y. A multi-scale patch mixer network for time series anomaly detection. Eng Appl Artif Intell. 2025;140:109687. doi:10.1016/j.engappai.2024.109687.

[8] He Y, Zhang X, Wang P, Liu M, Cao Z. Microservice traces anomaly detection based on structural clustering and graph attention autoencoder. In: 2024 7th International Conference on Advanced Algorithms and Control Engineering (ICAACE); 2024 Mar 1-3; Shanghai, China. Piscataway (NJ): IEEE; 2024. p. 977-81. doi:10.1109/ICAACE61206.2024.10549242.

[9] Singh V, Song Z, Narayanaswamy BM, Vaidya KE, Kraska T. Vista: machine learning based database performance troubleshooting framework in Amazon RDS. In: Proceedings of the 2024 ACM Symposium on Cloud Computing (SoCC '24); 2024 Nov 20-22; Redmond, WA, USA. New York (NY): Association for Computing Machinery; 2024. p. 83-98. doi:10.1145/3698038.3698519.

[10] Gu J, Yu H, Luo X. A masked autoencoder with strong-weak mutual information for anomaly detection in dynamic incomplete graphs. In: Companion Proceedings of the ACM on Web Conference 2025; 2025 Apr 28-May 2; Sydney, NSW, Australia. New York (NY): Association for Computing Machinery; 2025. p. 986-90. doi:10.1145/3701716.3715587.

[11] Yang C, Wu L, Xu J, Ren Y, Tian B, Wei Z. Graph learning framework for data link anomaly detection. IEEE Access. 2024;12:114820-8. doi:10.1109/ACCESS.2024.3445533.

[12] Zhang B, Pan H, Li K, Xing Y, Wang J, Fan D, et al. A blockchain and zero knowledge proof based data security transaction method in distributed computing. Electronics. 2024;13(21):4260. doi:10.3390/electronics13214260.

[13] Kuznetsov O, Rusnak A, Yezhov A, Kanonik D, Kuznetsova K, Karashchuk S. Enhanced security and efficiency in blockchain with aggregated zero-knowledge proof mechanisms. IEEE Access. 2024;12:49228-48. doi:10.1109/ACCESS.2024.3384705.

[14] Quattrocchi G, Plebani P. Trustworthy collaborative business intelligence using zero-knowledge proofs and blockchains. In: Islam S, Sturm A, editors. Intelligent information systems: CAiSE Forum 2024 proceedings; 2024 Jun 3-7; Limassol, Cyprus. Cham: Springer; 2024. p. 29-37. (Lecture Notes in Business Information Processing; vol. 520). doi:10.1007/978-3-031-61000-4_4.

[15] Liu X, Liu Y, Wei M, Xu P. LMGD: log-metric combined microservice anomaly detection through graph-based deep learning. IEEE Access. 2024;12:186510-9. doi:10.1109/ACCESS.2024.3481676.

[16] Rouf R, Rasolroveicy M, Litoiu M, Nagar S, Mohapatra P, Gupta P, et al. InstantOps: a joint approach to system failure prediction and root cause identification in microservices cloud-native applications. In: Proceedings of the 15th ACM/SPEC International Conference on Performance Engineering (ICPE '24); 2024 May 7-11; London, UK. New York (NY): Association for Computing Machinery; 2024. p. 119-29. doi:10.1145/3629526.3645047.

[17] Zhu Z, Lee C, Tang X, He P. HeMiRCA: fine-grained root cause analysis for microservices with heterogeneous data sources. ACM Trans Softw Eng Methodol. 2024;33(8):200. doi:10.1145/3674726.

[18] Wang Y, Zhu Z, Fu Q, Ma Y, He P. MRCA: metric-level root cause analysis for microservices via multi-modal data. In: Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering (ASE '24); 2024 Oct 27-Nov 1; Sacramento, CA, USA. New York (NY): Association for Computing Machinery; 2024. p. 1057-68. doi:10.1145/3691620.3695485.

[19] Tao L, Zhang S, Jia Z, Sun J, Ma M, Li Z, et al. Giving every modality a voice in microservice failure diagnosis via multimodal adaptive optimization. In: Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering (ASE '24); 2024 Oct 27-Nov 1; Sacramento, CA, USA. New York (NY): Association for Computing Machinery; 2024. p. 1107-19. doi:10.1145/3691620.3695489.

[20] Pham L, Ha H, Zhang H. BARO: robust root cause analysis for microservices via multivariate Bayesian online change point detection. Proc ACM Softw Eng. 2024;1(FSE):2214-37. doi:10.1145/3660805.

[21] Zheng L, Chen Z, He J, Chen H. MULAN: multi-modal causal structure learning and root cause analysis for microservice systems. In: Proceedings of the ACM Web Conference 2024 (WWW '24); 2024 May 13-17; Singapore, Singapore. New York (NY): Association for Computing Machinery; 2024. p. 4107-16. doi:10.1145/3589334.3645442.

[22] Wang P, Zhang X, Cao Z. Anomaly detection for microservice system via augmented multimodal data and hybrid graph representations. Inf Fusion. 2025;118:103017. doi:10.1016/j.inffus.2025.103017.

[23] Fan M, Zhang X, Wang P, Cao Z. Multi-modal anomaly detection for microservice system through nested graph diffusion reconstruction. Appl Intell. 2025;55(11):784. doi:10.1007/s10489-025-06681-1.

[24] Moens P, Steenwinckel B, Ongenae F, Volckaert B, Van Hoecke S. Toward context-aware anomaly detection for AIOps in microservices using dynamic knowledge graphs. IEEE Trans Netw Serv Manag. 2026;23:1970-88. doi:10.1109/TNSM.2026.3652304.

[25] Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P. The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process Mag. 2013;30(3):83-98. doi:10.1109/MSP.2012.2235192.

[26] Liu FT, Ting KM, Zhou ZH. Isolation forest. In: 2008 Eighth IEEE International Conference on Data Mining (ICDM); 2008 Dec 15-19; Pisa, Italy. Los Alamitos (CA): IEEE Computer Society; 2008. p. 413-22. doi:10.1109/ICDM.2008.17.

[27] Chen T, Guestrin C. XGBoost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16); 2016 Aug 13-17; San Francisco, CA, USA. New York (NY): Association for Computing Machinery; 2016. p. 785-94. doi:10.1145/2939672.2939785.

[28] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. In: Guyon I, von Luxburg U, Bengio S, Wallach H, Fergus R, Vishwanathan SVN, et al., editors. Advances in neural information processing systems 30. Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017); 2017 Dec 4-9; Long Beach, CA, USA. Red Hook (NY): Curran Associates; 2017. p. 5998-6008.

[29] Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations (ICLR 2017) [Internet]; 2017 Apr 24-26; Toulon, France. Amherst (MA): OpenReview.net; 2017 [cited 2026 Nov 18]. Available from: https://openreview.net/forum?id=SJU4ayYgl.

[30] Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y. Graph attention networks. In: 6th International Conference on Learning Representations (ICLR 2018) [Internet]; 2018 Apr 30-May 3; Vancouver, BC, Canada. Amherst (MA): OpenReview.net; 2018 [cited 2026 Nov 18]. Available from: https://openreview.net/forum?id=rJXMpikCZ.