**Artificial Intelligence and Computing Innovations**

Intelligence Science Publishing

https://www.ispublishing.com

ISSN: 3070-4944

Vo.4, No.2, 2024

# *Research on a Graph Contrastive Learning-Based Anomaly Detection Method for Backend Microservice Call Graph Structure Change Awareness*

**Yuruo Gong[1*], Arjun Mehta[2], Zixuan Zhou[2]**

[1]Rice University, Houston, USA

[2]University of Southern California, Los Angeles, USA

**Corresponding author: yuruogong.work@gmail.com*

***Abstract:*** Addressing the challenges of complex call relationships, dynamic changes in runtime states, and hidden anomaly propagation paths in backend microservice systems, this paper proposes a graph-based contrastive learning anomaly detection method that is aware of call graph structure changes. This method constructs a time-varying call graph based on the microservice call chain, incorporating service node states, dependencies, and their cross-time window structural changes into a unified modeling framework. It introduces a graph difference representation mechanism to characterize the evolutionary information between call graphs at adjacent time points and combines graph encoding, temporal aggregation, view generation, and change-guided representation learning to enhance the model's ability to distinguish between normal evolutionary behavior and anomalous perturbation patterns. Furthermore, it utilizes change-aware attention and adaptive message passing mechanisms to enhance the representation weights of key nodes and important dependencies, enabling the model to more effectively capture potential anomaly features such as local structural imbalances, abnormal link reconstruction, and service interaction offsets. Compared to methods relying solely on static topology or single-indicator modeling, the proposed method more fully mines the dynamic correlation information and structural semantics within the microservice system, thereby improving the accuracy and stability of anomaly identification. Comparative experimental results show that the proposed method exhibits good overall performance in anomaly detection tasks, verifying the effectiveness of combining structural change perception with graph contrastive learning, and demonstrating that backend microservice anomaly modeling from the perspective of time-varying call graphs has strong feasibility and application value.

***Keywords:*** Cloud-native operations and maintenance; dynamic graph representation learning; service dependency modeling; anomaly representation learning

## 1 Introduction

With the widespread adoption of cloud-native architectures, container orchestration technologies, and continuous delivery models, backend systems are rapidly evolving from traditional monolithic structures into distributed, collaborative systems composed of numerous microservices[1]. Microservice architecture improves system scalability, maintainability, and deployment flexibility through service decomposition. However, it also makes the dependencies, call paths, and operational states between services more dynamic, heterogeneous, and complexly coupled. In high-concurrency, frequent-iteration, and multi-tenant environments, factors such as service instance elastic scaling, version changes, link rerouting, and resource contention continuously alter the call relationship structure[2]. This causes system behavior to no longer be limited to a single node or local metrics, but rather to exhibit holistic, interconnected changes across services, links, and time series. This complex evolutionary characteristic makes traditional anomaly detection methods, which rely on static rules, threshold alerts, or single-metric monitoring, increasingly inadequate for accurately depicting the true state of the system and meeting the practical needs of modern backend systems for high reliability and intelligent operation and maintenance[3].

In microservice scenarios, anomalies often do not appear in isolation but propagate and amplify across different components through service dependency networks, leading to performance degradation, request failures, cascading failures, and even systemic outages[4,5]. Compared to traditional application environments, backend microservice anomalies are characterized by strong concealment, complex propagation paths, diverse causes, and blurred boundaries. Some anomalies may initially manifest as slight changes in call topology, abnormal reconstruction of local connection patterns, or sudden shifts in service interaction intensity. These signals may not be readily identifiable through routine log counts, resource utilization, or single-point response times. Therefore, modeling solely from the perspective of node attributes or independent time series is insufficient to fully reveal the structural roots behind anomalies[6]. Abstracting microservice systems into call graphs and understanding anomaly behavior from a unified perspective of graph structure, node relationships, and evolution has become a key direction for improving the accuracy, foresight, and interpretability of anomaly detection, and also provides a more systematic modeling foundation for perceiving the operational status of complex backend systems.

Graph learning methods offer an important technical path for modeling microservice call relationships. They can integrate service node attributes, call edge information, and topological context in a unified representation space, thereby effectively uncovering potential patterns in complex dependencies. However, existing methods still face several prominent challenges when dealing with the continuous changes in the graph structure during backend system operation. On the one hand, call graphs are not static but evolve with traffic distribution, service deployment, fault injection, and business strategy adjustments, making static graph modeling difficult to accurately reflect the true semantics of system operation. On the other hand, microservice anomalies are typically characterized by scarce samples, unstable categories, and high label acquisition costs, rendering supervised methods relying on large amounts of labeled data significantly limited in real-world scenarios. Against this backdrop, graph contrastive learning, by constructing consistency constraints from different perspectives,

can learn more discriminative structural representations under weak or even unsupervised conditions, providing a new solution for anomaly detection in complex dynamic environments. Further enhancing the ability to perceive changes in call graph structure could potentially distinguish the essential differences between normal evolution and anomalous disturbances[7].

Research on graph contrastive learning for anomaly detection, focusing on perceiving changes in the backend microservice call graph structure, has significant theoretical and practical value. Theoretically, this direction helps promote the cross-integration of graph representation learning, dynamic graph modeling, and anomaly detection methods, enriching the research system of intelligent perception and structured modeling for complex software systems, and providing a new analytical framework for understanding anomaly propagation mechanisms, structural disturbance patterns, and system evolution laws. From an application perspective, constructing anomaly detection methods oriented towards dynamic changes in call graphs helps improve the backend system's capabilities in early fault identification, risk warning, root cause localization, and stability assurance, reducing manual troubleshooting costs and operational response delays, and enhancing the continuous availability and service quality of large-scale online services. Against the backdrop of ongoing digital transformation and the continuous expansion of business system scale, this research has significant practical value in supporting the construction of highly reliable backend infrastructure, promoting the development of intelligent operations and maintenance, and ensuring the safe and stable operation of critical business systems.

## 2 Method

To characterize anomaly emergence in backend microservices from a structure-aware perspective, the running system is first represented as a time-evolving invocation graph rather than a collection of isolated service metrics, because abnormal behavior in practice is usually reflected by coordinated changes of dependency paths, interaction intensity, and local connectivity patterns. This paper also presents the overall model architecture, as shown in Figure 1.

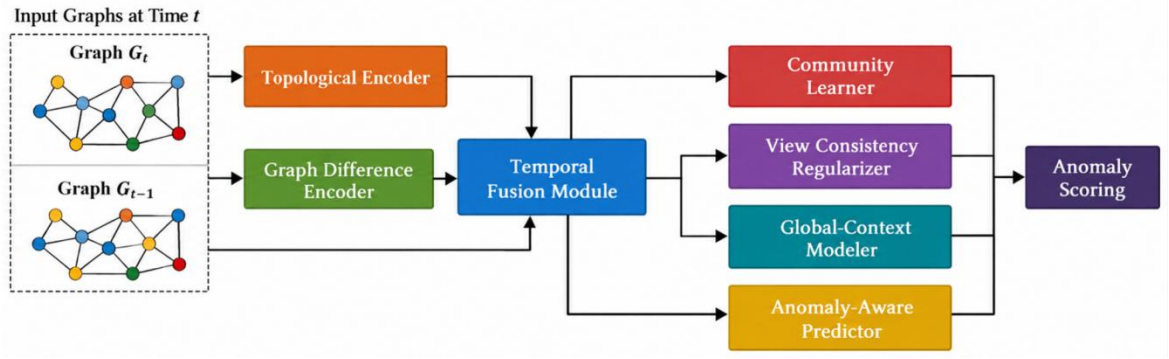



Figure 1. Overall model architecture

At each observation window $t$, the service ecosystem is modeled as a directed graph with node attributes and edge relation features, so that both operational semantics and call dependencies can be encoded within a unified representation space.

$$\mathcal{G}_t = \left(\mathcal{V}_t, \mathcal{E}_t, \boldsymbol{X}_t, \boldsymbol{R}_t\right) \tag{1}$$

Here, $\mathcal{V}_t$ denotes the set of service instances or logical services active in window $t$, while $\mathcal{E}_t$ records directed invocation dependencies, $\boldsymbol{X}_t$ contains node-level observations such as latency, load, or error statistics, and $\boldsymbol{R}_t$ preserves edge-level descriptors including call frequency, response fluctuation, and interaction type. Since the key difficulty does not lie only in describing one graph snapshot but in exposing how structural organization changes over time, a graph difference operator is introduced to quantify the evolution between adjacent windows and to preserve variation cues that may otherwise be smoothed out during representation learning.

$$\Delta\mathcal{G}_t = \Phi\left(\mathcal{G}_t, \mathcal{G}_{t-1}\right) \tag{2}$$

Through this design, the model is encouraged to distinguish ordinary service adaptation from suspicious topology perturbation, which is essential because elastic scaling, routing adjustment, and transient traffic bursts may also alter the invocation graph without necessarily indicating a fault. For this reason, the node representation is not generated only from the current snapshot, but from a joint aggregation of contemporaneous topology and temporal variation signals, allowing the learned embedding to preserve both stable dependency semantics and recent structural drift.

$$\boldsymbol{H}_t = f_\theta\left(\mathcal{G}_t, \Delta\mathcal{G}_t\right) \tag{3}$$

Such a formulation improves the sensitivity of the detector to structure-aware anomalies, as the encoder can focus on whether an observed change is compatible with normal graph evolution instead of merely measuring whether a metric exceeds a predefined threshold.

Building upon the dynamic graph representation, a contrastive learning strategy is adopted to learn discriminative embeddings without relying on dense anomaly annotations, because labeled fault samples in real backend systems are often sparse, incomplete, and unstable across services. Two correlated views are generated for each graph window by applying structure-preserving transformations to topology and attributes, where one view retains the major invocation backbone and the other emphasizes perturbed local dependency patterns, thereby forcing the encoder to capture invariant semantics under admissible graph variations. Given the paired views $\widetilde{\mathcal{G}}_t^{(1)}$ and $\widetilde{\mathcal{G}}_t^{(2)}$, their graph-level representations are projected into a contrastive space and optimized by maximizing the agreement of positive pairs while separating unrelated windows.

$$\mathcal{L}_{con} = -\log\frac{\exp\left(sim\left(\boldsymbol{z}_t^{(1)}, \boldsymbol{z}_t^{(2)}\right)/\tau\right)}{\sum_{k=1}^{N}\exp\left(sim\left(\boldsymbol{z}_t^{(1)}, \boldsymbol{z}_k^{(2)}\right)/\tau\right)} \tag{4}$$

In this objective, $\mathbf{z}_t^{(1)}$ and $\mathbf{z}_t^{(2)}$ are projected embeddings of the two views from the same time window, $\mathrm{sim}(\cdot,\cdot)$ denotes cosine similarity, $\tau$ is a temperature coefficient controlling distribution sharpness, and $N$ is the number of comparison instances within the mini-batch. By constraining semantically consistent graph states to remain close in latent space, the learned encoder becomes less vulnerable to superficial fluctuations and more capable of identifying structurally meaningful deviations.

Contrastive masked autoencoding for self-supervised anomaly representation learning further motivates recovering obscured topology and attribute segments from contextual graph dependencies, so that the encoder must retain regular structural semantics even when local evidence is incomplete or partially perturbed[8]. Knowledge graph-augmented semantic fusion likewise motivates integrating neighborhood relations and temporal context into a coherent representation, enabling the two graph views to preserve complementary structural signals during contrastive alignment[9].

Beyond global graph consistency, the proposed framework further emphasizes change-aware weighting so that the most informative structural variations contribute more strongly to anomaly-sensitive representation learning. Instead of treating all services and invocation links equally, the method assigns adaptive importance to nodes according to their temporal state shift and relational influence, which is beneficial because anomalies in microservice systems often originate from a few critical hubs and then propagate along dominant dependency paths. A variation-guided attention coefficient is therefore constructed to modulate message passing over the evolving call graph.

$$\alpha_{ij}^{t} = \frac{exp\left(\psi\left(\boldsymbol{h}_{i}^{t}, \boldsymbol{h}_{j}^{t}, \delta_{ij}^{t}\right)\right)}{\sum_{k \in \mathcal{N}(i)} exp\left(\psi\left(\boldsymbol{h}_{i}^{t}, \boldsymbol{h}_{k}^{t}, \delta_{ik}^{t}\right)\right)} \tag{5}$$

Within this mechanism, $\boldsymbol{h}_{i}^{t}$ and $\boldsymbol{h}_{j}^{t}$ denote hidden states of services $i$ and $j$ at time $t$, $\delta_{ij}^{t}$ measures the change intensity on the edge $(i, j)$, $\mathcal{N}(i)$ is the neighbor set of node $i$, and $\psi(\cdot)$ is a learnable compatibility function. As a result, the encoder pays greater attention to structurally unstable or behaviorally unusual dependencies, which improves the ability to capture weak anomaly precursors hidden in large-scale and noisy service interactions.

Unified semantic matching with logistic similarity learning provides a calibrated comparison principle for evolving graph embeddings: smoothly bounded similarity scores can distinguish coherent service adaptation from structural mismatches while retaining stable decision margins under noisy dependency changes[10]. Frequency-guided Transformer modeling with spectral enhancement further informs the temporal branch by preserving multi-scale periodic and abrupt variation patterns, strengthening sensitivity to call-graph shifts that are not evident in a single observation window[11].

Finally, anomaly scoring is derived from the discrepancy between the learned normal evolution manifold and the observed embedding trajectory, so that the decision process is aligned with the representation objective rather than imposed by an external rule base. A compact scoring head maps the dynamic graph embedding to a scalar risk estimate, and larger deviation values indicate that the current invocation structure is less compatible with historically coherent service behavior.

$$s_{t} = \sigma\left(\boldsymbol{w}^{\top} \boldsymbol{g}_{t} + b\right) \tag{6}$$

Here, $\boldsymbol{g}_{t}$ is the graph-level summary obtained from $\boldsymbol{H}_{t}$, $\boldsymbol{w}$ and $b$ are trainable parameters, and $\sigma(\cdot)$ denotes a nonlinear activation that stabilizes the score range. To preserve both representation consistency and anomaly sensitivity during optimization, the final objective combines contrastive supervision with structural regularization on temporal evolution.

$$\mathcal{L} = \mathcal{L}_{\text{con}} + \lambda \mathcal{L}_{\text{reg}} \tag{7}$$

Through this joint formulation, the method does not merely encode static call dependencies but learns whether the ongoing graph transition itself is reasonable under normal service dynamics, which provides a principled foundation for anomaly detection in backend microservice environments characterized by continuous topology change, strong dependency coupling, and complex propagation behavior.

## 3 Experimental Results and Analysis

### 3.1 Dataset

This paper selects a publicly distributed tracing dataset built on the TrainTicket microservice benchmark system as the research object. This dataset originates from an open-source data publishing platform, with the underlying system being the open-source TrainTicket microservice benchmark, providing relatively complete service call chain information and request-level tracing records. Simultaneously, the data includes samples related to reserve-ticket business requests and provides structural information directly related to call path modeling, such as execution paths, thus effectively supporting structural modeling and anomaly analysis tasks for backend microservice call graphs. Compared to data containing only host metrics or isolated log fragments, this dataset is more suitable for abstracting a request into a call graph or call subgraph, thereby characterizing inter-service dependencies, link propagation processes, and local topology perturbation features.

From a data organization perspective, this dataset consists of preprocessed tracing results. Samples are published in CSV format, with each record containing a tracing ID and duration information for each component, differentiated according to the microservice component where the anomaly occurred. Furthermore, the original data generation process combines a microservice tracing system with a performance anomaly injection mechanism, ensuring that it not only retains the structural characteristics of normal call chains but also includes anomaly scenario information that can be used for anomaly detection research. Since the method in this paper emphasizes the perception of changes in the call graph structure and explores the differences between normal evolution and abnormal perturbations through graph representation learning, this dataset has strong matching in terms of request-level call paths, component-level latency changes, and abnormal location annotations, and can provide a relatively reliable data foundation for microservice call graph construction, dynamic relationship modeling, and abnormal representation learning.

### 3.2 Experimental Results and Analysis

To verify the effectiveness of the proposed method in microservice call chain anomaly detection scenarios, representative works highly related to this research direction were selected as comparison objects. These methods mainly focus on distributed tracing, execution path modeling, graph structure representation, and microservice anomaly identification, effectively covering representative technical routes in this field. Based on a unified evaluation criterion, comparison results of each method under

four commonly used metrics are further presented to comprehensively illustrate the differences in detection performance. The experimental results are shown in Table 1.

Table 1. Experimental results compared with other models

| Method | ACC | Precision | Recall | AUC |
|---|---|---|---|---|
| Nedelkoski et al.[12] | 91.84 | 90.27 | 89.96 | 90.11 |
| Meng et al.[13] | 93.28 | 92.14 | 91.73 | 91.93 |
| Kohyarnejadfard et al.[14] | 94.12 | 93.45 | 93.08 | 93.26 |
| Chen et al.[15] | 95.03 | 94.11 | 93.84 | 93.97 |
| Xie et al.[16] | 95.46 | 94.88 | 94.35 | 94.61 |
| Tang et al.[17] | 94.67 | 93.92 | 93.57 | 93.74 |
| Panahandeh et al.[18] | 95.18 | 94.36 | 94.08 | 94.22 |
| Ours | 96.34 | 95.62 | 95.28 | 95.45 |

The proposed algorithm demonstrates strong advantages in classification accuracy, discrimination stability, and anomaly identification completeness, indicating that it can effectively mine the structural information and state change features of the backend microservice call graph. Since microservice anomalies are often accompanied by dependency perturbations, local interaction imbalances, and link behavior deviations, relying solely on static representations or local observations is often insufficient to accurately characterize anomaly patterns. Our method, by introducing a mechanism for perceiving the evolution of the call graph, enables the model to capture more identifiable anomaly representations in complex service relationships, thus exhibiting higher reliability and consistency in overall identification performance.

While accurately identifying anomaly samples, our method also effectively distinguishes normal samples, demonstrating that the learned representation possesses not only strong anomaly sensitivity but also good structural robustness. This shows that combining graph structure change modeling with graph contrastive learning can effectively alleviate the identification difficulties caused by ambiguous anomaly boundaries, complex propagation paths, and frequent dynamic evolution in microservice scenarios, making the model closer to the operational characteristics of real backend systems. Therefore, the method presented in this paper can provide more effective technical support for anomaly detection in backend microservice environments, and further verifies the rationality and necessity of anomaly modeling based on the dynamic structure of the call graph.

As shown in Figure 2, the proposed algorithm maintains good anomaly recognition and discrimination stability even with varying time window lengths, indicating its strong modeling capability for dynamic structural information in the backend microservice call graph. Because the model simultaneously considers call relationship evolution, local structural perturbations, and temporal state changes during representation learning, it can continuously extract discriminative anomaly features at different time granularities, demonstrating good adaptability and robustness. This shows that combining a structural change perception mechanism with graph contrastive learning effectively enhances the model's ability to characterize anomaly patterns in complex microservice environments, thus providing more reliable support for anomaly detection in backend systems.

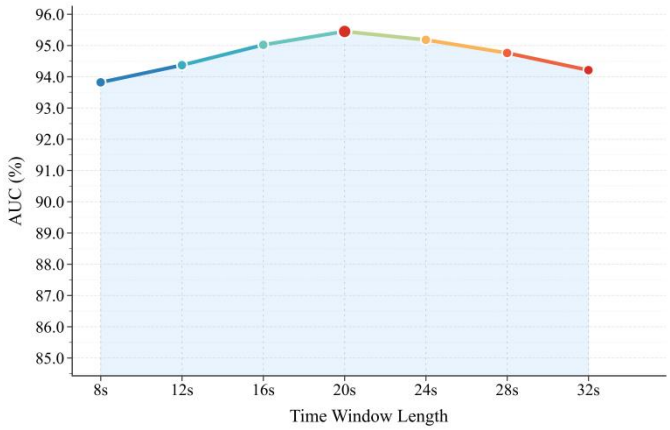



Figure 2. The impact of time window length hyperparameter sensitivity on AUC

## 4 Conclusion

This paper addresses the challenges of anomaly detection in backend microservice environments, including strong structural dynamism, complex dependencies, and concealed anomaly propagation. It proposes a graph-contrastive learning-based anomaly detection method oriented towards awareness of call graph structure changes. Starting from the real-world operational characteristics of microservice systems, this research incorporates inter-service call relationships, node state information, and the graph structure evolution process into a unified modeling framework. This allows anomaly detection to move beyond identifying single-point metric fluctuations or local state anomalies, focusing instead on the formation mechanism and structural representation of anomalies within the overall dependency network. By tightly integrating call graph change information with the representation learning process, the model can more fully characterize the essential differences between normal evolution and anomalous perturbations, thus providing a more scenario-appropriate technical path for anomaly identification in complex backend systems.

From a methodological perspective, the significance of this work lies not only in constructing an anomaly detection framework for dynamic graph structures but also in emphasizing the intrinsic connection between structural information, temporal changes, and representation consistency in microservice systems. Traditional anomaly detection methods often struggle to balance structural dependencies and dynamic evolution. This paper, by introducing a change-aware mechanism and graph-contrastive learning, enables the model to learn more discriminative anomaly representations even under weak labeling or complex noise conditions. This modeling approach helps improve the model's ability to identify local link anomalies, dependency refactoring anomalies, and propagating anomalies. It also further illustrates that backend microservice anomaly detection should shift from static state analysis to dynamic structural understanding. Therefore, anomaly modeling based on the evolutionary patterns of the call graph not only improves the completeness of problem characterization but also expands the research space for graph learning methods in intelligent operation and maintenance of software systems.

From an application value perspective, this research has strong practical significance for supporting the stable operation of cloud-native systems, improving intelligent operation and maintenance capabilities, and ensuring the continuity of critical businesses. With the widespread deployment of

microservice architectures in scenarios such as internet platforms, financial services, e-commerce, industrial internet, and smart government, system scale is constantly expanding, and service coupling relationships are continuously strengthening. The accuracy, timeliness, and interpretability of anomaly detection have become important factors affecting system reliability. The method presented in this paper, by characterizing abnormal behavior at the call graph level, can provide a more granular operational awareness foundation for complex backend systems, helping to reduce manual troubleshooting costs, shorten fault discovery and response cycles, and provide support for subsequent fault localization, root cause analysis, risk warning, and service governance. This research has significant potential for widespread application in digital infrastructures facing high concurrency, strong dynamism, and high availability requirements, and can provide valuable references for intelligent operation and maintenance construction in related fields.

Future research can be further deepened around microservice environments with larger scale, stronger heterogeneity, and higher real-time performance. On the one hand, a multi-source operational data collaborative learning mechanism can be introduced based on existing structural change modeling to more closely integrate call graphs with log semantics, indicator timing, event context, and other information, thereby enhancing the completeness of anomaly representation and the ability to understand system state. On the other hand, more efficient dynamic graph representation update strategies and online detection mechanisms can be explored to adapt to the characteristics of frequent service changes, continuous topology reconstruction, and rapid request pattern drift in actual production environments. Simultaneously, research on the interpretability of anomaly detection results, cross-system migration capabilities, and linkage mechanisms with automated O&M decisions will also help promote the relevant methods from theoretical modeling to wider engineering implementation. As the scale and complexity of distributed systems continue to increase, anomaly detection research oriented towards dynamic call graph structures still has broad development prospects and will play a more profound role in areas such as intelligent O&M, service governance, and critical system reliability assurance.

## References


[1] C. Zhang, X. Peng, C. Sha, 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, pp. 623-634.

[2] X. Peng, "Large-scale trace analysis for microservice anomaly detection and root cause localization," in Proceedings of the Federated Africa and Middle East Conference on Software Engineering, 2022, pp. 93-94.

[3] S. Zhang, P. Jin, Z. Lin, et al., "Robust failure diagnosis of microservice system through multimodal data," IEEE Transactions on Services Computing, vol. 16, no. 6, pp. 3851-3864, 2023.

[4] J. Huang, Y. Yang, H. Yu, et al., "Twin graph-based anomaly detection via attentive multi-modal learning for microservice system," in Proceedings of the 2023 38th IEEE/ACM International Conference on Automated Software Engineering, IEEE, 2023, pp. 66-78.

[5] C. Zhao, M. Ma, Z. Zhong, et al., "Robust multimodal failure detection for microservice systems," in Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2023, pp. 5639-5649.

[6] H. Chen, P. Chen, B. Wang, et al., "Graph neural network based robust anomaly detection at service level in SDN driven microservice system," Computer Networks, vol. 239, p. 110135, 2024.

[7] S. Fang, J. Chao, Z. Xiang, et al., "Research on Anomaly Detection in Microservice Based on Graph Neural Networks," Computer Fraud and Security, vol. 2024, no. 9, pp. 44-56, 2024.

[8] J. Zhou, "Contrastive Masked Autoencoding for Self-Supervised Anomaly Representation Learning in Distributed Task Systems," 2024.

[9] Z. Zhu, "Knowledge Graph-Augmented Semantic Fusion for Large Language Model-Based Long-Text Classification," 2024.

[10] R. Hu, "Unified Semantic Matching with Logistic Similarity Learning for Cold-Start Advertising Recommendation," 2024.

[11] Y. Lyu, J. Jiang, and J. Hu, "Frequency-Guided Transformer Modeling with Spectral Enhancement for Accurate Time Series Forecasting," Transactions on Computational and Scientific Methods, vol. 4, no. 10, 2024.

[12] S. Nedelkoski, J. Cardoso, and O. Kao, "Anomaly detection and classification using distributed tracing and deep learning," in Proceedings of the 2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, IEEE, 2019, pp. 241-250.

[13] L. Meng, F. Ji, Y. Sun, et al., "Detecting anomalies in microservices with execution trace comparison," Future Generation Computer Systems, vol. 116, pp. 291-301, 2021.

[14] I. Kohyarnejadfard, D. Aloise, S. V. Azhari, et al., "Anomaly detection in microservice environments using distributed tracing data analysis and NLP," Journal of Cloud Computing, vol. 11, no. 1, p. 25, 2022.

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

[16] Z. Xie, C. Pei, W. Li, et al., "From point-wise to group-wise: A fast and accurate microservice trace anomaly detection approach," in Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2023, pp. 1739-1749.

[17] H. Tang, Y. Guo, J. Yang, et al., "Microservice anomaly diagnosis with graph convolution network based on implicit microservice dependency," in Proceedings of the 2023 9th International Conference on Computing and Artificial Intelligence, 2023, pp. 437-441.

[18] M. Panahandeh, A. Hamou-Lhadj, M. Hamdaqa, et al., "ServiceAnomaly: An anomaly detection approach in microservices using distributed traces and profiling metrics," Journal of Systems and Software, vol. 209, p. 111917, 2024.