DeST: An Unsupervised Decoupled Spatio-Temporal Framework for Microservice Incident Management

Xiaohui Nie*†, Hang Cui*‡, Changhua Pei*†, Haotian Si*‡, Ke Xiang*‡,
Jingjing Li*, Yanbiao Li*†, Gaogang Xie*, Dan Pei§
*Computer Network Information Center, Chinese Academy of Sciences,
{xhnie, cuihang, chpei, htsi, xiangke, ljj, lybmath, xie}@cnic.cn
†Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences
‡University of the Chinese Academy of Sciences,
§Tsinghua University, peidan@tsinghua.edu.cn

***Abstract*—Effective incident management in large-scale microservice systems demands both accurate anomaly detection (AD) and precise root cause localization (RCL) across heterogeneous data modalities. However, existing approaches often treat these tasks in isolation, resulting in redundant maintenance, delayed response, and the absence of shared diagnostic context. While recent efforts have explored unified frameworks to support both tasks, these approaches often suffer from high false-alarm rates due to cross-modal interference. To address these issues, we propose *DeST* , an unsupervised decoupled spatio-temporal framework that jointly performs anomaly detection and root cause localization. *DeST* proposes a multi-stage fusion strategy that decouples temporal and spatial feature learning to mitigate cross-modal interference and prevent cross-modal interference. Furthermore, it incorporates task-specific modal routing to direct learned representations to different tasks, enhancing both detection and localization accuracy. To ensure robustness against transient noise, *DeST* designs a Differential Multi-Scale Convolutional Network (DMCN) for noise-resistant temporal feature representation. We evaluate *DeST* on two real-world microservice benchmarks, where it achieves a perfect F1-score of 1.00 for anomaly detection and outperforms existing methods in root cause localization accuracy. Ablation studies highlight the effectiveness of key components. Our unified framework reduces false alarms in anomaly detection and streamlines root cause localization, providing a robust and practical solution for microservice incident management.**

***Index Terms*—Microservice System, Anomaly Detection, Root Cause Localization, Spatio-Temporal Model**

I. Introduction

Microservice architectures have become foundational to modern digital infrastructures due to their scalability, agility, and modularity. However, as these systems scale to encompass thousands of loosely coupled components, failure is inevitable, leading to significant operational overhead and financial losses. Recent large-scale service outages at major cloud providers such as Microsoft [1], Google [2], and Alibaba Cloud [3] highlight the critical need for efficient incident management for microservice systems. Figure 1 shows a typical microservice management process, where a microservice system consists of various interconnected services deployed across multiple nodes. These components generate vast amounts of operational data, including metrics [4], [5], logs [6], [7], and traces [8]–[10], which are utilized by Site Reliability Engineers (SREs) to support anomaly detection (AD) and root cause localization (RCL). To ensure system reliability, SREs must continuously monitor and analyze high-dimensional multimodal data to detect anomalies in real time. When an anomaly occurs, operators need to quickly identify the root cause to mitigate the incident. This process is often time-consuming and labor-intensive. As a result, there is a growing demand for integrating AI technologies into microservice incident management.

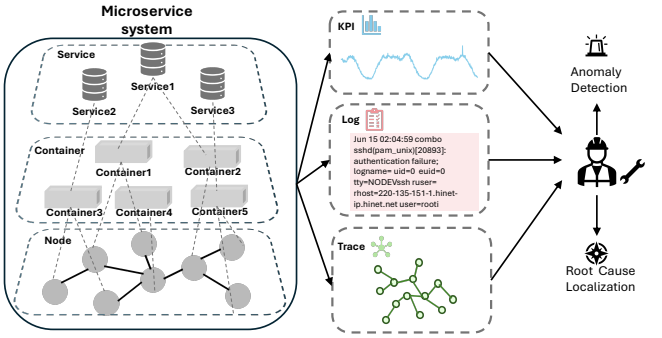

Fig. 1: The hierarchical structure of a microservice system: the system consists of interacting services, deployed on interconnected nodes, each node containing multi-dimensional metrics

Current approaches to intelligent microservice incident management can be broadly categorized into two paradigms. The first adopts separate models for anomaly detection and root cause localization. Anomaly detection is typically performed using univariate or multivariate time series anomaly detection methods [12], [13]. While root cause localization relies on specialized algorithms, such as [14]. However, this decoupled architecture suffers from knowledge isolation between components, redundant maintenance overhead, and de-

* Changhua Pei is the corresponding author.

layed response with the absence of shared diagnostic context. The second paradigm focuses on an all-in-one strategy, which integrates anomaly detection and root cause localization into a unified framework. Representative works include ART [15] and Erdao [16]. Although Erdao achieves good performance via supervised learning, its reliance on labeled fault data hinders its applicability in real-world production environments, where such labels are scarce [8]. In contrast, ART adopts an unsupervised approach that sequentially models channel, temporal dependency, and call dependency, followed by deviation analysis for anomaly detection and root cause localization. However, empirical results show that ART suffers from high false-alarm rates (details in Section II-A). In this context, *our goal is to propose an efficient unsupervised framework that can jointly perform anomaly detection and root cause localization for microservice incident management*. To achieve this goal, it mainly has the following three challenges:

- **How to build an efficient multimodal model with less cross-modal interference?** Microservice systems generate multimodal data, including metrics, logs, and traces, as shown in Figure 1. These modalities exhibit both correlations and specificities. Existing methods, such as ART [15] and Eadro [16], adopt monolithic fusion strategies that indiscriminately combine modalities, often leading to cross-modal interference and elevated false-alarm rates (Section II-A). Effectively modeling inter-dependencies among modalities while mitigating cross-modal interference to reduce cross-modal interference remains a significant challenge.
- **How can task-relevant knowledge be selectively routed to downstream modules?** Current methods propagate fused knowledge indiscriminately to downstream tasks, resulting in a task-knowledge mismatch that degrades both anomaly detection precision and root cause localization accuracy. Specifically, anomaly detection primarily relies on system-level temporal patterns, and the inclusion of irrelevant spatial information can hinder its effectiveness. In contrast, root cause localization depends on fine-grained spatial relationships across services. This conflict highlights the need for a principled task-specific modal routing mechanism, which is absent in current frameworks.
- **How to efficiently model time-series data with transient noise?** Time-series data serve as the primary input for incident management, encompassing service-level KPIs and node-level metrics. Logs can also be transformed into time-series representations, enabling effective multimodal fusion [15]. However, these time-series data in microservices often contain noise due to workload fluctuations and inherent randomness in distributed systems, posing major challenges for reliable temporal modeling. Existing methods lack robustness against such noise, resulting in frequent false alarms (details in Section II-C). This highlights the need for temporal representation techniques that are both expressive and resilient to noise.

In this paper, we propose *DeST* , an unsupervised DEcoupled Spatio-Temporal framework that can jointly perform anomaly detection and root cause localization through three key innovations. *(1) To address challenge one*, *DeST* proposes a **Multi-Stage Fusion Strategy**. *DeST* processes multimodal data through two dedicated pathways: *Temporal Fusion* for noise-resilient temporal pattern learning, and *Spatial Fusion* for capturing instance-level dependencies. Specifically, raw KPIs, logs, and traces are transformed into time-series sequences and instance dependency graphs to enable effective multimodal fusion. *(2) To address challenge two*, *DeST* utilizes a **Task-specific Modal Routing mechanism**. It routes system-level temporal patterns to the anomaly detection task, while directing instance-level spatial relationships to the root cause localization task, thereby avoiding task-knowledge mismatch and enhancing performance. *(3) To address challenge three*, *DeST* design a novel **Differential Multi-Scale Convolutional Network (DMCN)** for microservice time-series data modeling. It combines multiscale differential operators to suppress observational noise while preserving critical temporal patterns. Temporal features are aggregated at the system level for anomaly detection via adaptive thresholding. When anomalies are detected, the spatial fusion stage activates to analyze instance-level causality using graph attention networks over call graphs, with a contrastive aggregation mechanism comparing system-level and instance-level embeddings via cosine similarity to pinpoint root causes. Extensive experiments on industrial-grade datasets validate the effectiveness of *DeST* . It achieves a perfect anomaly detection F1-score of 1.00 on both *D1* (46 instances) and *D2* (18 instances), outperforming the previous state-of-the-art ART by 6.2% on *D1* (from 0.942 to 1.00) and 9.0% on *D2* (from 0.917 to 1.00). Moreover, the total time cost for *DeST* 's AD and RCL also outperforms the baseline and remains under one minute.

The main contributions of this work are as follows:

- We conducted a comprehensive empirical study on the performance of existing multimodal models in incident management, revealing three key limitations: cross-modal interference, task–knowledge mismatch in downstream tasks, and sensitivity to transient noise in microservice time series modeling, each contributing to high false alarm rates and suboptimal incident management performance.
- We are the first to propose an unsupervised decoupled spatio-temporal framework for microservice incident management, achieving state-of-the-art performance in both anomaly detection (AD) and root cause localization (RCL) tasks. *DeST* proposed three novel techniques: multi-stage fusion strategy and task-specific modal routing mechanism, noise-resistant time series modeling(DMCN), significantly enhancing the performance of both anomaly detection (AD) and root cause localization (RCL). To facilitate reproducibility and further research, we have released our source codes at https://github.com/CSTCloudOps/DeST

- As the novel time-series modeling approach in *DeST* , DMCN is a lightweight, noise-resilient architecture that employs differential convolution to effectively capture multi-scale temporal patterns in microservice environments. It also demonstrates strong generalization and achieves a new state-of-the-art on the AIOps dataset in EASYTSAD benchmark [17], which includes widely used univariate time-series anomaly detection datasets[1].

II. Motivation

This section elaborates on the design principles of our framework by exploring three key topics: *1) How effective are existing multimodal fusion frameworks in microservice incident management? 2) How to elegantly fuse and share multimodal data? 3) How to perform temporal modeling efficiently in microservice scenarios?* The first topic aims to investigate the limitations of current multimodal fusion frameworks in modeling microservice multimodal data through an empirical study. The second topic provides guidance on how to extract correlations from multimodal data while preserving the specificity of knowledge, and how to elegantly share this knowledge after unified modeling. The third topic discusses time series features in microservice environments and the consideration for enabling efficient time-series modeling.

A. How effective are existing multimodal fusion frameworks in microservice incident management?

Modern microservice systems generate heterogeneous operational data streams (metrics, logs, traces), as shown in Figure 1. Current multimodal fusion frameworks typically adopt an *all-in-one* strategy for knowledge sharing (as shown in Figure 3), where all metrics and dependency graphs are input into models to generate unified representations. Through unsupervised contrastive learning, these frameworks attempt to reconstruct normal temporal-spatial patterns for anomaly detection and root cause localization. The all-in-one strategy enables models to learn temporal-spatial feature interactions, predicting future representations based on historical data. However, using redundant knowledge for downstream task modeling not only increases model complexity, leading to higher computational overhead, but can also cause cross-modal interference, thereby decreasing model effectiveness. Taking the ART method (use all-in-one strategy) as an example, our analysis of Dataset *D1* reveals a key issue in this paradigm. As shown in Figure 3, instances with constant zero values are still predicted with false fluctuations, learned from neighboring nodes through the model's topological connections. These erroneous fluctuations, induced by the all-in-one strategy, propagate anomalies through dependency edges, significantly increasing the model's false alarms.

The all-in-one strategy forces instances to inherit fluctuations from adjacent nodes through topological feature fusion, resulting in cascading error propagation during anomaly detection. By fusing metric and topological features of the microservice system within the model, the graph-based modeling captures explicit inter-instance relationships through channel dependencies. Our empirical measurements reveal heterogeneous correlation distributions of instances across different channels. We calculated inter-instance correlations across three channels in dataset *D1* in Figure 4, channel-specific dependency patterns exhibit significant variance across service channels. High correlation coefficients ($\rho > 0.8$) in Channel 1 correspond to low correlations ($\rho < 0.2$) in Channels 2-3. *This observation indicates that all-in-one modeling of all channel dependencies compels the model to learn spurious correlations, ultimately increasing the false alarm of anomaly detection.*

B. How to elegantly fuse and share multimodal data?

In Section II-A, our empirical analysis reveals that indiscriminate fusion of multimodal microservice data induces spurious fluctuation modeling, elevating false alarm rates in anomaly detection. This phenomenon stems from conflicting requirements between two fundamental aspects of incident management. *1) Temporal Feature Learning*: Focuses on capturing metric-specific evolutionary patterns through normal operational data. Training on historical temporal sequences enables models to learn channel-specific fluctuation signatures (e.g., stable variations in disk usage vs. volatile patterns in CPU utilization) and intrinsic metric periodicity reflecting service-specific operational rhythms. *2) Topological Feature Learning:* Encodes spatial dependencies between microservice instances through structural propagation patterns of anomalies across service call graphs, where proximal instances exhibit elevated anomaly scores during failures, while distal nodes maintain normal operation.

The dilemma in multimodal fusion: Anomaly detection demands precise temporal modeling to minimize false positives, yet spatial feature incorporation introduces cross-modal interference. Conversely, root cause localization requires joint spatiotemporal modeling to trace anomaly propagation pathways. To resolve this dichotomy, we propose a multi-stage Fusion architecture with Task-Specific Routing (Figure 2). *Stage 1: Temporal Fusion,* focuses on metric-specific temporal pattern learning for anomaly detection while preserving non-spatial knowledge. The temporal features are utilized for anomaly detection and serve as input for the spatial fusion in spatial fusion when an anomaly occurs. *Stage 2: Spatial Fusion*, learns call-relationship patterns from invocation data exclusively for root cause localization. This staged decoupling prevents spatial feature interference during anomaly detection while enabling systematic diagnosis through controlled knowledge integration.

C. How to perform temporal modeling efficiently in microservice scenarios?

In a microservice system, time series information reflects the important operational status of services. However, there is a large amount of natural noise in the time series of microservice systems [17]. For example, sudden traffic spikes from specific user groups or transient resource contention

[1]https://adeval.cstcloud.cn/

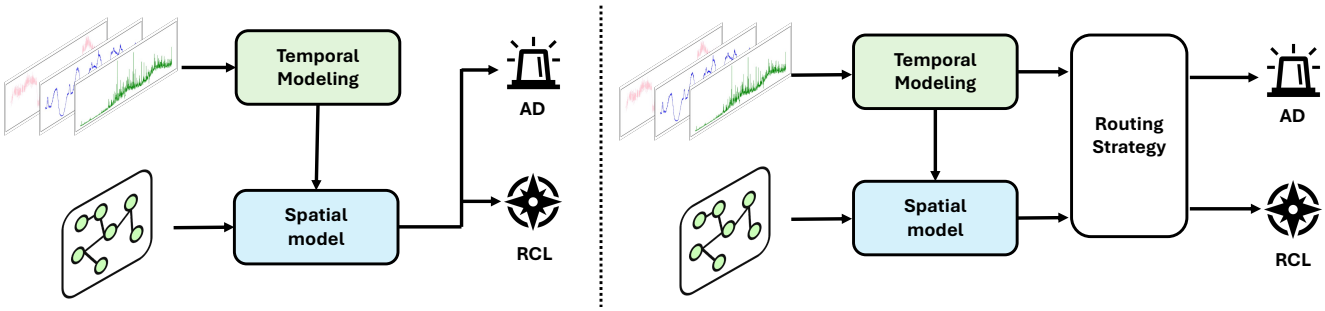

Fig. 2: The left side shows the traditional *all-in-one* fusion strategy, where all multimodal data is combined at once. The right side shows a multi-stage fusion strategy proposed by us, where data is fused step by step to share useful knowledge for different downstream tasks of microservice incident management.

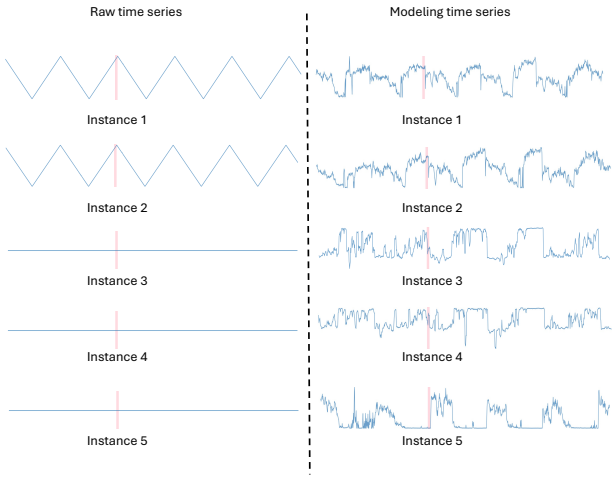

Fig. 3: The false alarm in instance 5, where its modeling time series is influenced by the surrounding instance under the all-in-one fusion strategy adopted by ART.

across distributed nodes can induce short-term fluctuations that resemble anomalies. Current temporal modeling methods, designed for stationary or periodic patterns, struggle to distinguish genuine anomalies from noise in such dynamic environments. Moreover, existing approaches often employ overly complex architectures to model temporal dependencies and lack explicit mechanisms to suppress noise, inadvertently amplifying noise and leading to high false positive rates. For instance, transformer-based models [18] utilize multi-head attention to model long-range dependencies, yet their sensitivity to minor metric variations exacerbates overfitting. Similarly, hybrid architectures [19], [20] often conflate transient noise with genuine anomalies due to redundant parameterization. As shown in Figure 5, state-of-the-art models misclassify noise-induced fluctuations as anomalies, undermining reliability in real-world deployments.

To address this issue, we propose Differential Temporal Convolution, a lightweight yet robust architecture tailored for microservice temporal modeling. Differential Temporal Convolution addresses these limitations through differential noise suppression and multi-scale convolution learning. Differential noise suppression computes 1st-order temporal differences $\Delta X_t = X_t - X_{t-1}$ to amplify sustained trends while attenuating transient noise. Then multi-scale convolution processes ΔX through parallel convolution streams including fine-grained kernels to learn local anomaly detection and coarse-grained kernels to direct long-term trend modeling. Full connection fusion of multi-scale features enables robust pattern recognition while maintaining computational efficiency critical for large-scale microservice deployments.

III. Preliminary

This section formalizes the mathematical foundation for joint anomaly detection (AD) and root cause localization (RCL) in microservice systems, establishing key definitions and task-specific modal routing principles.

A. Problem Formulation

A microservice system comprises *microservice instances* - deployable units executing specific functionalities (e.g., processing user requests). These instances communicate through predefined protocols, forming a dynamic topological structure. Each instance generates three modalities of observability data:

KPI: Structured multivariate time series $\mathbf{X}_{\text{metric}}^{(i,t)} \in \mathbb{R}^{C_1}$ capturing performance indicators (CPU usage, memory consumption) sampled at fixed intervals, representing real-time operational states.

Log: Semi-structured textual records aggregated into temporal frequency sequences through sliding window processing, formally represented as $\mathbf{X}_{\text{log}}^{(i,t)} \in \mathbb{R}^{C_2}$.

Traces: Directed acyclic graphs (DAGs) recording request execution paths and service dependencies, with each snapshot $\mathbf{X}_{\text{trace}}^{(i,t)} \in \mathbb{R}^{C_3}$ containing latency measurements and parent-child relationships. The System Behavior Graph (SBG) is defined as $G = (V, E, F)$ where: V is a set of all microservice instances, E includes directed edges (v_i, v_j) representing actual service invocations and F represents latent features derived from multimodal observations. In addition, traces come with RTT, duration, and other information, represented as $\mathbf{X}_{\text{trace}}^{(i,t)} \in \mathbb{R}^{C_3}$ through sliding window.

Given a microservice system with N instances, let $\mathbf{X} = \{\mathbf{X}_{\text{metric}}, \mathbf{X}_{\text{log}}, \mathbf{X}_{\text{trace}}\}$ denote multimodal data collected over T timesteps. The joint state of instance i at time t is represented as: $\mathbf{X}^{(i,t)} = [\mathbf{X}^{(i,t)}\text{metric} \parallel \mathbf{X}^{(i,t)}\text{log} \parallel \mathbf{X}_{\text{trace}}^{(i,t)}] \in \mathbb{R}^k$, $k = C_1 + C_2 + C_3$, with$\mathcal{N}(i,t)$ denoting its topological neighborhood derived from trace graphs. Our framework addresses two interdependent tasks:

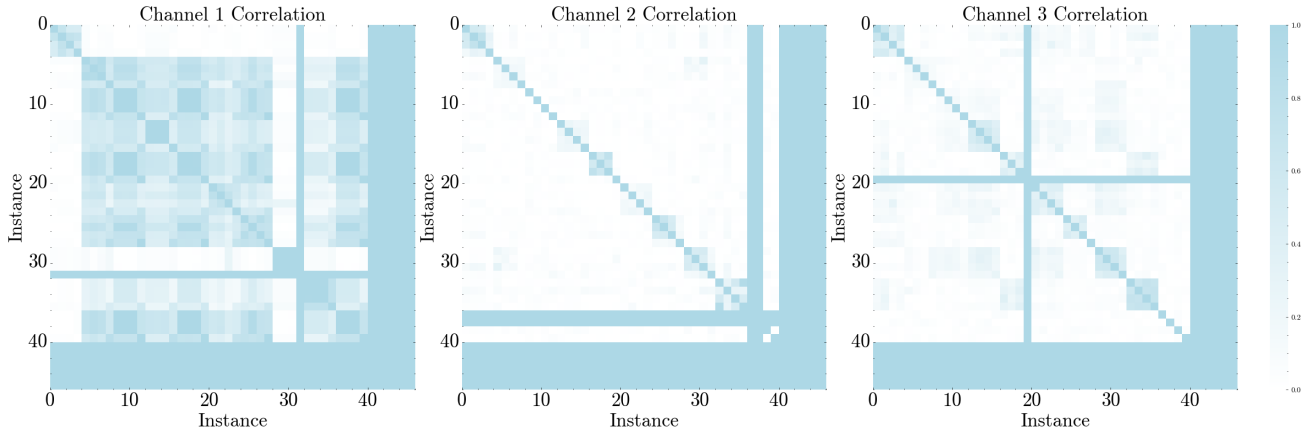

Fig. 4: Instance correlation under different channels (*i.e.,* KPIs), three graphs represent three channels, and horizontal and vertical coordinates represent 46 instances

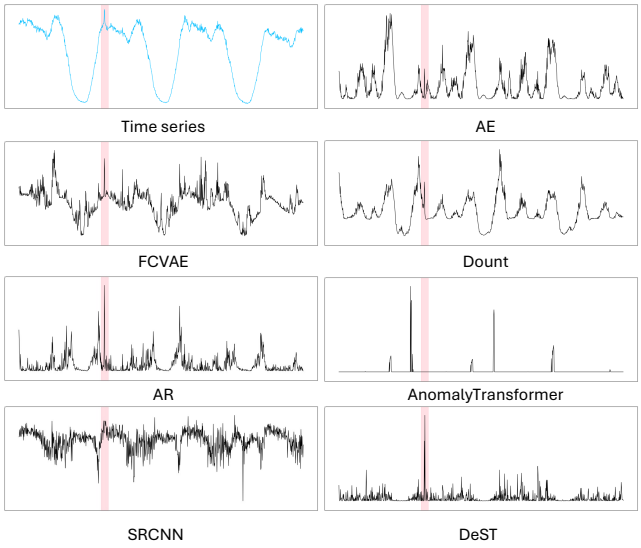

Fig. 5: Anomaly detection case study in microservice time series monitoring. The original time series is the first blue curve, with pink-shaded regions indicating expert-labeled anomaly intervals. Black curves represent the anomaly scores generated by different detection methods using temporal modeling.

Anomaly Detection (AD): Calculate a system-level anomaly score and judge $y_t \in \{0, 1\}$ indicating whether the system is abnormal at time t.

Root Cause Localization (RCL): If $y_t = 1$, output a ranked list $\mathbf{P}_t = [p_1, \ldots, p_N] \in [0, 1]^N$, where p_i represents the likelihood of instance i being the root cause.

B. Design Principles

Our architectural design is governed by two complementary principles derived from the unique characteristics of microservice multimodal data:

Anomaly Detection Principle: The AD task requires robust temporal modeling to distinguish normal and abnormal system states. As established in Section II, microservice time series exhibit two critical characteristics: (1) inherent noise in individual instance measurements, and (2) noise amplification

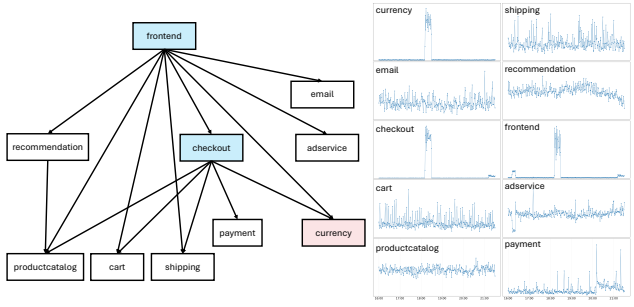

Fig. 6: The architecture of the fault injection system, where red nodes represent the nodes that inject network anomalies and blue nodes represent their adjacent nodes.

through spatial message passing in conventional graph neural networks. This dual noise effect leads to overfitting and excessive false alarms. Our design addresses these challenges through two key mechanisms:

- *Multi-Stage Fusion*: We decouple temporal modeling from service dependencies during feature extraction, preventing noise propagation through the call graph. This isolation creates a protected temporal subspace where instance-level patterns can be learned without cross-modal interference from neighboring nodes.
- *Differential Convolution*: Applied exclusively to the temporal domain, this operation enhances noise resilience through differential operation. This suppresses high-frequency noise while preserving abrupt anomaly signatures.

Root Cause Localization Principle: While spatial modeling is intentionally suppressed during anomaly detection to prevent noise propagation, our systematic analysis reveals its critical role in root cause analysis. To validate this hypothesis, we conduct a controlled experiment on the system[2], a deployed e-commerce platform comprising 10+ interdependent microservices shown in Figure 6. First and foremost, the experimental does fault Injection into microservices' pods by

[2]https://github.com/GoogleCloudPlatform/microservices-demo

chaosmesh[3]. We inject a network attack targeting the currency service (2025-05-05 18:11:31 to 18:29:31) and monitor latency metrics across all services during fault propagation. It demonstrates a key phenomena that direct neighbors (checkout, frontend) exhibit amplified anomaly signals than other services due to dependency-driven propagation. These findings motivate us to utilize graph attention networks do spatial modeling, analyzing anomaly signals amplified by dependency-driven aggregation.

IV. Design

A. Overview of framework

DeST is an unsupervised framework designed to automatically and accurately detect anomalies and locate root causes in microservice environments. As shown in Figure 7, our architecture leverages a multi-stage fusion strategy and task-specific modal routing to process heterogeneous monitoring data (KPIs, logs, and traces) through three synergistic modules: Temporal Fusion Module, Spatial Fusion Module, and Downstream Task Module.

The workflow begins by preprocessing multi-modal streams into normalized time windows. The Temporal Fusion Module employs historical temporal windows to generate predictive representations for subsequent timesteps. These representations simultaneously serve two purposes: 1) enabling real-time anomaly detection through the downstream diagnosis module. 2) feeding into the Spatial Fusion Module for unsupervised representation learning. When anomalies are detected, the system activates the root cause localization component, which analyzes the propagated spatial-temporal patterns from both modules to perform root cause localization.

B. Temporal Fusion

The goal of this module is to perform time series modeling for each monitoring data source separately. Referring to [15], [16], we transform multimodal data (metrics, logs, and traces) into time series. For metrics, which naturally form time series, we treat them directly as such. For logs, we count the frequency of log events per minute to construct a time series. For traces, we extract latencies and convert them into a time series. Finally, we concatenate these data to form an integrated multivariate time series$M = [M_{\text{metric}} \parallel M_{\text{log}} \parallel M_{\text{trace}}]$. By performing a z-score normalization, using a sliding history window on M , a normalized input sequence can be obtained, $X = \{X^{(1)}, \ldots, X^{(T)}\}$, where T is the length of the time window. The snapshot of the microservice system at time t can be represented as $X^{(t)} \in R^{N \times K}$, where N is the number of instances and K is the number of data channels. $x_{i,j}^{(t)}$ denotes the normalized value of channel j on instance i at time t.

Our temporal fusion module employs a novel Differential-Convolutional Architecture (DCA) to capture multi-scale temporal patterns in microservice systems. As shown in Figure 7, this component consists of three key operations:

[3]https://github.com/chaos-mesh/chaos-mesh

1) Differential Feature Extraction: To address the challenges of metric drift and transient anomalies in microservice systems, for the input sequence $X = \{X^{(1)}, \ldots, X^{(T)}\}$, The module first computes 1-th order temporal differences to amplify system state transitions:

$$\Delta^{(k)} x_t = \Delta^{(k-1)} x_t - \Delta^{(k-1)} x_{t-1} \tag{1}$$

where $k \in 1, 2, 3...$denotes the difference order. This operation enhances sensitivity to metric evolution patterns while suppressing static noise.

2) Multi-Scale Convolutional Learning: In order to capture time trends for each dimension of the microservice system, We implement parallel temporal convolutions with complementary receptive fields:

$$h_1 = \text{ReLU}(W_{conv1} * \Delta X + b_1) \tag{2}$$

$$h_2 = \text{ReLU}(W_{conv2} * \Delta X + b_2) \tag{3}$$

Improving from the classical TCN [21], but breaking through its single time scale limitation, the two-stream convolutional architecture can capture the short and long-term in microservice system: fine-grained detectors use convolution kernels with small receptive fields to capture transient fluctuations(such as spikes and drops in requests), and coarse-grained detectors use convolution kernels with large receptive fields to capture trend evolution(such as CPU usage consumption).

3) Adaptive Feature Fusion: The final prediction combines convolutional outputs through dimension-aware transformation:

$$\hat{y} = W_{fc} \cdot \text{Flatten}([h_1 \oplus h_2]) + b_{fc} \tag{4}$$

where $\oplus$ denotes channel-wise concatenation.

C. Spatial Fusion

Improving from the previous work [22]–[24], in order to model the interaction between instances, failure propagation graph learning module implements a Dynamic Graph Attention Network to model fault propagation patterns in microservice systems. The architecture operates through two fundamental operations:

1) Multi-Head Graph Attention Mechanism: For node v_i and its neighbors $N(i)$, the attention coefficient α_{ij} between nodes is computed as:

$$e_{ij} = \text{LeakyReLU}\left(A^{\top}[Wh_i \parallel Wh_j]\right) \tag{5}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik})} \tag{6}$$

where $W \in R^{d' * d}$ is the learnable weight matrix and A denotes the attention vector. Our implementation employs dual attention heads to capture complementary dependency patterns:

$$hi' = \Big|\Big|_{m=1}^{2} \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(m)} W^{(m)} h_j\right) \tag{7}$$

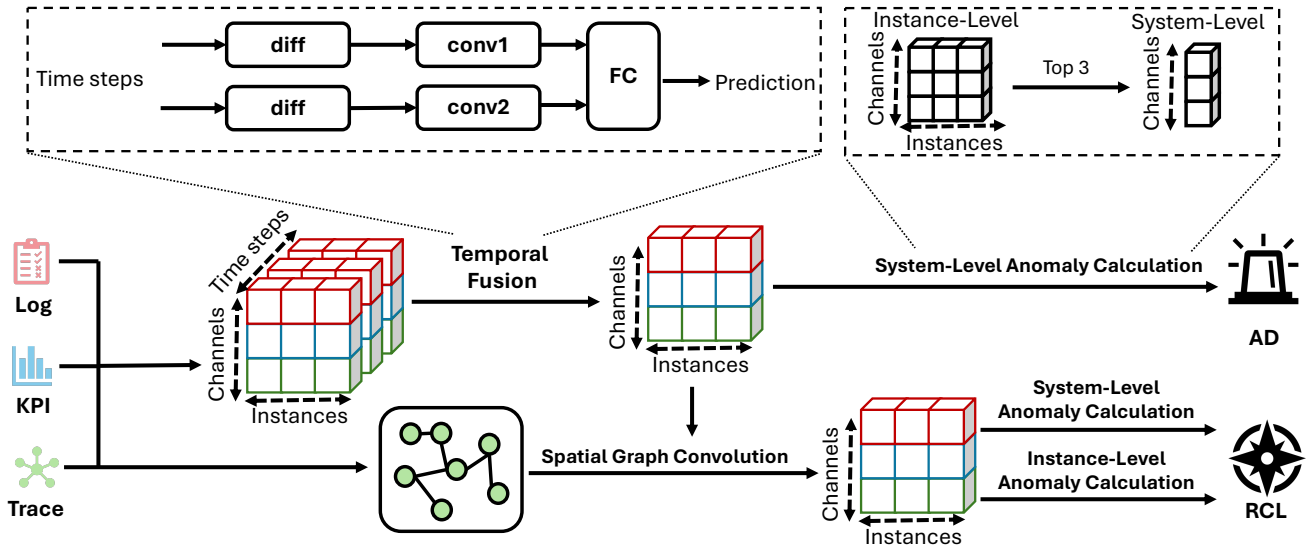

Fig. 7: Architecture of *DeST* : (1) Temporal Fusion Module captures multi-scale metric temporal patterns, (2) Spatial Fusion Module models instance spatial dependencies, (3) Downstream Task Module integrates results for anomaly detection and root cause localization.

2) Hierarchical Graph Encoding: The encoder stack contains L graph attention layers with progressive feature refinement:

$$H^{(0)} = \text{ELU}(W_{\text{in}}X + b_{\text{in}}) \tag{8}$$

$$H^{(l)} = \text{ELU}\left(\frac{1}{M}\sum_{m=1}^{M}\sum_{j\in\mathcal{N}(i)}\alpha_{ij}^{(m,l)}W^{(m,l)}H_j^{(l-1)}\right) \tag{9}$$

$$Z = \text{Dropout}(H^{(L)}) \tag{10}$$

where $Z \in R^{N\times d_{\text{out}}}$ represents the fault propagation scores across N nodes. Initial node embeddings are projected through parametric mapping, each layer l performs neighborhood-aware feature aggregation and final node representations combine structural patterns.

D. Downstream Task

DeST continues to perform the downstream task of anomaly detection for the system, and performs the downstream task of root cause analysis when an exception occurs.

1) Anomaly Detection: As analyzed in Section II-A, temporal patterns dominate anomaly detection while spatial dependencies introduce noise. We therefore formulate the anomaly score as:

$$s_t^{(i,k)} = |\hat{X}_t^{(i,k)} - X_t^{(i,k)}|_2^2 \tag{11}$$

where:$\hat{X}_t \in R^{N\times k}$ representative predicted metrics from temporal fusion, $X_t \in R^{N\times k}$ representative observed metrics, N is the number of instances, K is the number of channels. The system-level anomaly score($score_s > \tau$) aggregates anomaly score as:

$$score_s = \sum_{i=1}^{N}\sum_{k=1}^{K} s_t^{(i,k)} \tag{12}$$

Following established practices [17], [25], [26], we employ Extreme Value Theory (EVT) for dynamic thresholding

$$\tau = \mu + \gamma\sigma \tag{13}$$

where γ is derived from the generalized Pareto distribution fitting historical anomaly extremes.

2) Root Cause Localization: when $score_s > \tau$, root cause analysis activates, this module uses predicted metrics from failure propagation graph learning as $\hat{X}_t$, then calculate the anomaly score for each instance with anomaly score on the microservice system of cosine similarity.

$$\rho_i = \frac{score_i \cdot score_s}{|score_i||score_s|} \tag{14}$$

Instances with higher cosine similarity are considered as root cause sets.

V. Evaluation

Our evaluation addresses three critical research questions through comprehensive experiments on industrial microservice datasets:

- **RQ1:** Does *DeST* achieve state-of-the-art performance in both anomaly detection (AD) and root cause localization (RCL) for microservice systems?
- **RQ2:** How does our multi-staged knowledge fusion mitigate cross-modal interference compared to conventional all-in-one fusion strategies?
- **RQ3:** Does *DeST* 's temporal modeling generalize across diverse time-series datasets?

A. Experiment Setting

1) Datasets: In order to maintain fairness with the existing methods, our evaluation adopts two industry-standard datasets representing distinct microservice architectures to ensure comprehensive benchmarking:

TABLE I: Experimental results on datasets *D1* and *D2*, - means the method does not cover the problem.

Method	D1						D2					
	AD			RCL			AD			RCL		
	Precision	Recall	F1	Top1	Top3	AVG@5	Precision	Recall	F1	Top1	Top3	AVG@5
ART	0.899	0.990	0.942	0.667	**0.810**	0.776	0.877	0.960	0.917	0.722	0.889	0.870
Eadro	0.425	0.946	0.586	0.137	0.315	0.302	0.767	0.935	0.842	0.157	0.315	0.310
Hades	0.866	0.863	0.865	-	-	-	0.867	0.868	0.868	-	-	-
CAD	0.896	0.969	0.931	-	-	-	0.770	0.832	0.800	-	-	-
DeST w/o graph	**1.0**	**1.0**	**1.0**	0.655	0.714	0.738	**1.0**	**1.0**	**1.0**	0.685	0.852	0.811
DeST w/o msf	0.896	0.969	0.931	**0.678**	0.774	0.764	0.829	0.846	0.837	**0.759**	**0.907**	**0.881**
DeST	**1.0**	**1.0**	**1.0**	**0.678**	**0.810**	**0.786**	**1.0**	**1.0**	**1.0**	**0.741**	**0.907**	0.878

D1: A cloud-deployed e-commerce simulation with authentic business traffic patterns and replayed failure scenarios in May 2022. This dataset originates from a production system handling lots of users with weekly software updates, where failed changes will cause substantial economic losses. Key characteristics include: 46-node architecture (40 microservices + 6 VMs), Diverse hardware/software failure modes(Container Hardware, Network, CPU, Memory, Disk)

D2: AIOps Challenge 2021 benchmark[4] from a banking management system with 18 heterogeneous components. This dataset is composed of 18 heterogeneous components (microservices, servers, databases, containers). The failures consist of resource (CPU/Memory/Disk) and JVM-related failures from Jan-Jun 2021, which are conducted the labeling process separately and cross-checked the labels by the operator to ensure consensus.

Following established evaluation [15], [27], [28], we split each dataset using the first timestamp in the last 40% of failure cases as the cutoff point to ensure temporal continuity between training and testing phases.

2) Baselines: We benchmark against four state-of-the-art approaches covering key methodology paradigms:

- **CAD** [12]: Multi-gate Mixture-of-Experts architecture with metric-specific expert selection and dual gating mechanisms, optimized for baseline drift scenarios.
- **HADES** [13]: Semi-supervised cross-modal framework integrating causal CNNs for metrics and transformers for logs, enhanced with dynamic attention alignment.
- **Eadro** [16]: Multi-source framework combining Hawkes processes (logs), dilated convolutions (KPIs), and graph attention networks (traces) for joint anomaly-propagation analysis.
- **ART** [15]: Unsupervised unified model with transformer-GRU-GraphSAGE architecture, achieving previous state-of-the-art performance on both *D1*/*D2* benchmarks.

3) Evaluation Metrics: Anomaly detection is a binary classification problem, used to determine whether an anomaly has occurred. In the task of anomaly detection, TP stands for the detected anomalies correctly, TN represents that the model did not give a warning during the normal period, FP represents false alarms, and FN represents the quantity of missed anomalies. Evaluation metrics consist of standard precision(precision $=$ TP$/$(TP $+$ FP)), recall(recall $=$ TP$/$(TP $+$ FN)) and F1 metrics(F1-score $= \frac{2\cdot\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}}$).

For root cause localization, the metric need to quantify the system's ability to surface ground truth root causes within practical investigation budgets. For each failure instance i with true root cause g_i , we compute: $\text{TopK}_i = \mathbb{I}\left(g_i \in P_{i,[1:K]}\right)$ where $\mathbb{I}$ is the indicator function and $P_{i,[1:K]}$ denotes the ranked list of TopK predicted candidates. The system-level accuracy aggregates over N evaluated failures: $\text{TopK} = \frac{1}{N}\sum_{i=1}^{N}\text{TopK}_i$.

AVG@5 Composite Metric: Addresses operational requirements for progressive fault investigation by evaluating performance across practical diagnostic depths (K from 1 to 5):

$$\text{AVG@5} = \frac{1}{5}\sum_{K=1}^{5}\text{TopK}(K) \tag{15}$$

This compound metric emphasizes consistent ranking performance, reflecting real-world troubleshooting workflows where engineers progressively inspect top candidates.

B. RQ1: Performance on downstream tasks for microservice incident management

In the context of tasks for microservice incident management, a high false positive (FP) rate in AD signifies a substantial number of false alarms, which can result in considerable resource wastage and diminished confidence in the model among operations personnel. Besides, A high FN rate indicates many false negatives, and when anomalies are not detected, considering the accuracy of RCL becomes meaningless. As illustrated in Table 1, *DeST* achieves perfect F1 scores (1.00) across both datasets, demonstrating improvements of AD ranging from 5.8% to 41.4% over baseline methods while maintaining leading RCL accuracy. This achievement sets a new benchmark for state-of-the-art end-to-end solutions for microservice incident management.

As shown in Table II, in the joint task involving anomaly detection and root cause localization, *DeST* demonstrates significantly better time efficiency than ART. Specifically, on the two benchmark datasets D1 and D2, the end-to-end processing time of *DeST* was only 59.55 seconds and 42.50 seconds respectively, successfully achieving the minute

[4]https://www.aiops.cn/gitlab/aiops-nankai/data/trace/aiops2021

TABLE II: Time Cost Comparison on D1 and D2 (seconds)

Method	D1		D2	
	AD	RCL	AD	RCL
ART	40.081	76.002	138.166	44.089
DeST	11.681	47.873	15.376	27.119

response goal. In contrast, the joint task time consumption of ART reached 116.08 seconds and 182.26 seconds respectively, which verified the help of *DeST* in terms of architecture optimization for efficiency.

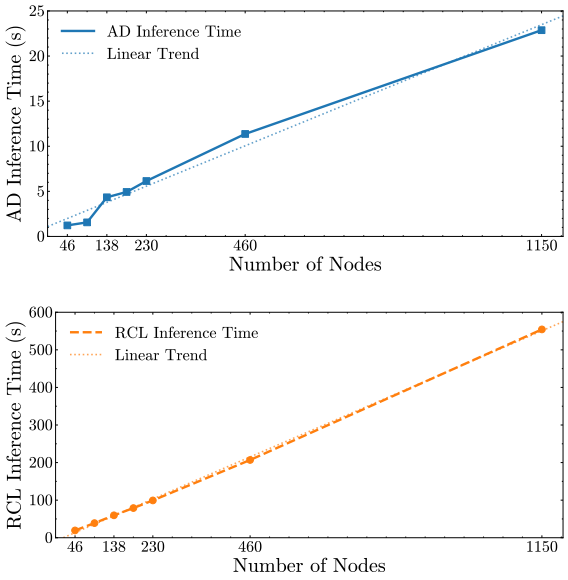

Fig. 8: Large-Scale Node Experiment

To further assess model deployability, we scaled the datasets to thousands of nodes and billions of records. As shown in Figure 8, large-scale experiments demonstrate that our anomaly detection and root cause localization methods maintain efficiency, confirming the scalability and robustness of our approach for larger deployments. Furthermore, Dest can theoretically be accelerated through parallel preprocessing techniques, such as GPU acceleration for DMCN convolutions in AD, and batched or subgraph strategies for GAT in RCL.With modest distributed computing resources, achieving the one-minute processing time for over 1000 microservices is well within reach.

C. RQ2: Effective Modeling via Knowledge Fusion

To thoroughly investigate RQ2 and validate our framework's capability to mitigate cross-modal interference, we conducted a systematic comparison between the conventional all-in-one fusion strategy (adopted by ART) and our proposed fusion approach. To avoid the impact of individual case contingency, we monitor mean squared error (MSE) as the primary metric to monitor convergence patterns during training. by comparing the convergence effect of the loss, we assess the fitting performance of both training frameworks. As shown in Figure

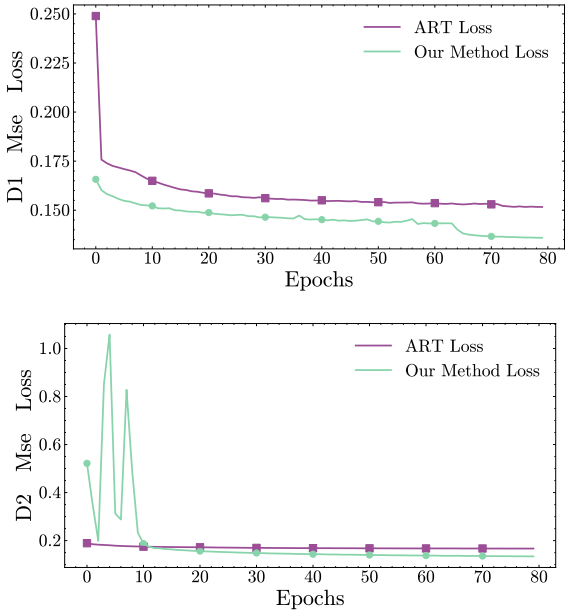

Fig. 9: Convergence of all-in-one strategy and fusion strategy in datasets *D1* and *D2*

9, compared to the MSE score of the all-in-one strategy, the fusion strategy achieves significantly lower MSE values. For instance, on D1, fusion reduces the final MSE by 11.8% (from 0.152 to 0.136), while on D2, the improvement reaches 24.6% (from 0.167 to 0.134). This stark contrast highlights the limitations of indiscriminate feature integration: the all-in-one strategy introduces noisy spatial-temporal interactions during early training phases, leading to unstable gradient updates and suboptimal feature representations. In contrast, our approach decouples temporal and spatial learning into dedicated modules, allowing each branch to specialize in domain-specific patterns before selectively fusing task-relevant knowledge.

D. Ablation Study

To quantify the contribution of each module in *DeST* , we performed an ablation study by systematically removing key components and evaluating their impact on performance (Table 1).

Spatial Dependency Module (*DeST* w/o Graph): The graph encoder plays a critical role in capturing spatial dependencies between instances, enabling the model to understand the relational structure of interconnected microservices. To further validate the importance of spatial dependencies, we perform an ablation study by disabling the graph attention network. The results demonstrate a substantial drop in root cause localization performance, with AVG@5 scores decreasing by 6.25% on *D1* (0.786 $\rightarrow$ 0.738) and 8.2% on *D2* (0.878 $\rightarrow$ 0.811). This confirms that spatial dependencies modeling is indispensable for accurately inferring fault causality across interconnected nodes. Without this module, the framework fails to fully exploit the spatial relationships, which significantly hampers its ability to pinpoint the root causes of incidents in a complex microservice environment.

Fusion Strategy (*DeST* w/o msf): Compared to all-in-one strategy methods such as ART, *DeST* introduces a new multi-stage fusion strategy, along with different frameworks for modeling temporal and spatial information. To demonstrate the effectiveness of the multi-stage fusion strategy, we replaced it with a simpler feature concatenation approach (all-in-one fusion) while keeping the temporal and spatial fusion modules intact. The results revealed a significant performance drop in anomaly detection, with F1-score decreasing by 7.4% on D1 ($1.0 \rightarrow 0.931$) and 19.5% on *D2* ($1.0 \rightarrow 0.837$). This drop underscores the necessity of a multi-stage fusion strategy that allows the framework to retain and process the nuances of both temporal and spatial dependencies separately before merging them. In contrast, the all-in-one fusion approach, by treating spatial and temporal features equally at the same stage, fails to preserve the distinctive information carried by each modality, leading to less accurate anomaly detection.

E. RQ3: Temporal Modeling in Microservice Incident Management

To rigorously evaluate *DeST* 's temporal modeling capabilities in microservice environments, we conduct extensive experiments on the EASYTSAD benchmark (ISSRE-UTS benchmark) using the AIOPS dataset. This dataset comprises operational metrics collected from five major internet companies (Sogou, eBay, Baidu, Tencent, and Alibaba), with data sampled at 1–2 minute intervals. The dataset captures diverse failure scenarios, including latency spikes, resource contention, and cascading failures, making it ideal for validating time-series modeling robustness in dynamic microservice systems. We compare *DeST* against 11 state-of-the-art (SOTA) baselines spanning classical and deep learning paradigms:

- AE [29]: Reconstructs time windows using autoencoders; anomalies flagged by reconstruction error.
- Donut [30]: Employs VAE to model latent distributions and uncertainty for anomaly scoring.
- LSTMAD [31]: Utilizes multi-step LSTM predictions to compute forecast errors.
- FCVAE [19]: Decomposes time series into frequency components for multi-scale modeling.
- TimesNet [20]: Transforms time series into 2D space via frequency-based folding.
- SRCNN [32]: Leverages spectral residual analysis and CNNs for anomaly detection.
- EncDecAD [33]: Combines LSTM-based encoder-decoder architectures to capture long-term dependencies.
- AR [34]: Do anomaly detection by using the method of autoregression
- SubLOF [35]: Traditional distance-based outlier detection using local outlier factors.
- TranAD [18]: Integrates transformers and adversarial training to enhance sensitivity.
- AnomalyTransformer(AT) [36]: Contrasts attention distributions between normal and abnormal patterns.

To address threshold bias, as illustrated in Figure 10, we employ Best F1 as a primary metric that theoretically achieves optimal F1-scores through threshold optimization for all methods. However, this metric leads to artificial score inflation from redundant counting of consecutive anomalies in prolonged events. Recognizing that operational teams prioritize sustained anomaly event detection over isolated point anomalies, we complement this with three critical operational metrics: Event F1 for segment-level detection that treats continuous anomaly segments as single events to evaluate detection capability independent of duration, Event AuPRC to assess precision-recall balance at the event level, and Event F1 k-delay – a stringent metric requiring detection within k timesteps of anomaly onset to quantify system responsiveness. This latter metric specifically reflects the critical need for timely alerting in microservice incident management, where delayed detection impedes rapid root cause analysis. This multi-metric framework systematically quantifies both detection accuracy and operational responsiveness, aligning theoretical optimization with practical microservice incident management workflows.

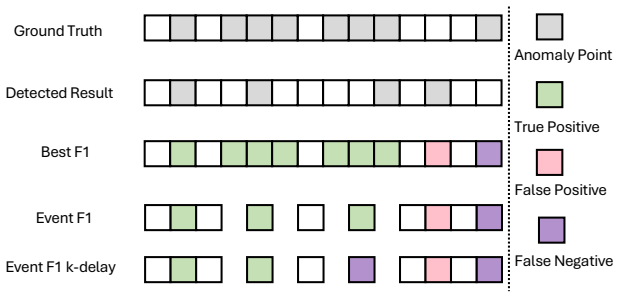

Fig. 10: Evaluation Metrics Specification.

As shown in Figure 11, *DeST* achieves highest Best-F1 score while surpassing all baselines in event-centric metrics. Best F1: *DeST* significantly outperforms other baselines, demonstrating its robustness in point-wise anomaly detection. Event F1 and Auprc: *DeST* attains 0.84 and 0.85, respectively, surpassing all competitors. This highlights its superiority in detecting persistent anomalies critical for microservice operations. Event F1 k-delay: Across varying detection windows (5, 10, and 15 timesteps), *DeST* consistently achieves the highest scores. This rapid alerting capability provides root cause localization with precious lead time for proactive diagnosis and mitigation.

To evaluate the denoising capability of the differential operator in microservice time-series modeling, we conduct comprehensive ablation studies comparing two variants: 1) the complete *DeST* framework with differential operations, and 2) *DeST* w/o Diff, where we systematically remove the differential processing module. The comparative evaluation across three industry-standard datasets (*D1*, *D2*, and AIOPS Challenge Dataset) reveals that the differential operator consistently demonstrates superior noise suppression, as shown in Figure 12.

VI. Related Work

Current methodologies for microservice incident management primarily follow two technical paradigms: single-task

	Best F1	Event F1	Event F1 5-delay	Event F1 10-delay	Event F1 15-delay	Event Auprc
AE	0.88	0.71	0.66	0.68	0.68	0.72
AR	0.89	0.74	0.69	0.71	0.71	0.75
Dount	0.85	0.69	0.66	0.67	0.67	0.69
EncDecAD	0.9	0.77	0.71	0.73	0.73	0.77
LSTMAD	0.93	0.77	0.72	0.74	0.74	0.78
FCVAE	0.92	0.78	0.74	0.75	0.75	0.79
TimesNet	0.81	0.65	0.6	0.62	0.62	0.64
SRCNN	0.53	0.15	0.057	0.084	0.089	0.075
SubLOF	0.73	0.38	0.24	0.32	0.33	0.31
TranAD	0.75	0.62	0.56	0.59	0.6	0.56
Anomaly Transformer	0.58	0.33	0.24	0.27	0.28	0.24
DeST	0.94	0.84	0.79	0.8	0.81	0.85

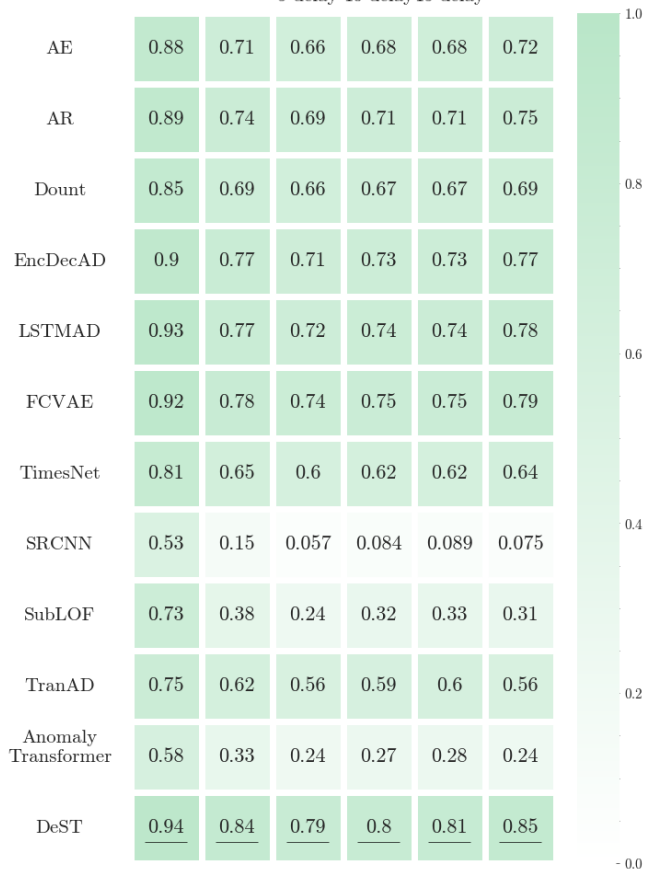

Fig. 11: Overall performance ranking using different metrics. The row names denote the names of metrics, while the column names denote the names of the methods and the best score of each column is underlined.

approaches that model problems in isolation, and multi-task techniques that exploit cross-modal relationships.

Single-task Techniques: Conventional methods employ autoencoder-based reconstruction frameworks where architectures detect anomalies through reconstruction errors (AE [29]). Subsequent variants incorporate probabilistic modeling (Donut [30]) and frequency-space transformations (TimesNet [20]). Modern enhancements include attention mechanisms (AnomalyTransformer [36]) and frequency-constrained variational autoencoders (FCVAE [19]). Advanced solutions capture cross-metric and cross-modal correlations through architectures: CAD [12] implements metric-sensitive Mixture-of-Experts with dual gating mechanisms, while HADES [13] performs heterogeneous pattern fusion through synchronized processing of metric data via causal CNNs and log sequences via transformer networks, coupled with dynamic attention alignment. While optimized for temporal pattern recognition, these approaches suffer from redundant feature engineering and maintenance costs. The absence of unified contextual representations additionally leads to delayed incident resolution.

Multi-task Techniques: Contemporary unified frameworks co-optimize anomaly detection and root cause analysis: 1) Supervised Paradigms: Eadro [16] demonstrates multimodal fusion of logs, KPIs, and traces for anomaly propagation modeling, However, its dependence on labeled fault data restricts practical deployment in annotation-scarce environments. 2) Unsupervised Paradigms: ART [15] pioneers a cascaded transformer-GRU-GraphSAGE architecture modeling channel, temporal, and call dependencies. Despite achieving state-of-the-art performance, its fixed modality processing hierarchy introduces cross-modal interference during feature encoding.

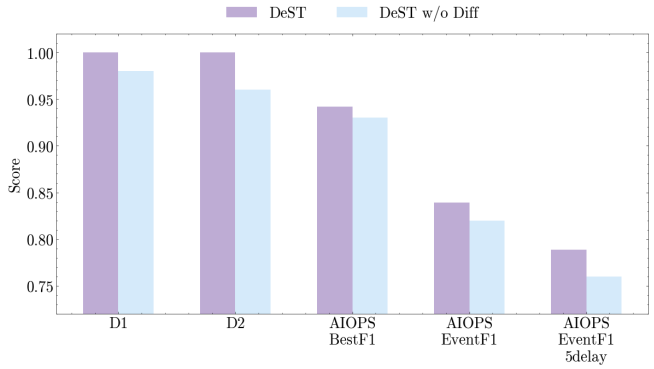

Fig. 12: Performance of *DeST* and *DeST* w/o Diff in different datasets

VII. Conclusion

In this paper, we present *DeST* , a novel unsupervised spatio-temporal framework tailored for end-to-end microservice incident management. *DeST* addresses three fundamental challenges that have hindered prior approaches: (1) cross-modal interference in all-in-one fusion strategies, (2) task-knowledge mismatch between anomaly detection and root-cause localization, and (3) sensitivity to transient noise in microservice time series modeling. To overcome these, we propose a multi-stage fusion architecture that decouples temporal and spatial representation learning, alongside a task-specific routing mechanism that directs distinct knowledge representations to the appropriate downstream tasks. Furthermore, we introduce the Differential Multi-Scale Convolutional Network (MCDCN), which suppresses high-frequency noise while effectively capturing both short- and long-term patterns. Extensive experiments on two industrial benchmarks (D1, D2) and the EASYTSAD AIOPS dataset demonstrate that *DeST* not only achieves perfect F1-scores (1.00) for system-level anomaly detection(improving over the previous SOTA by up to 5.8% in D1 and 8.3%), but also sets new records in event-level detection metrics and obtains highest localization accuracies. Ablation studies validate the contributions of each module: both the decoupled spatial graph component and the multi-stage fusion strategy significantly reduce false alarms and accelerate convergence compared to all-in-one fusion baselines. By explicitly isolating and then selectively integrating modality-specific knowledge, *DeST* delivers a robust solution for incident management in large-scale microservice systems.

VIII. Acknowledgement

This work is supported by the CNIC Young Scientists Research Fund under Grant NO. E5553701, the National Natural Science Foundation of China (62202445), and the National Natural Science Foundation of China-Research Grants Council (RGC) Joint Research Scheme (62321166652).

References

[1] Microsoft, "Microsoft's azure networking takes a worldwide tumble," https://www.theregister.com/2024/07/30/microsofts_azure_portal_outage/, 2025.

[2] Google, "Google cloud services hit by outage in paris," https://thenewstack.io/google-cloud-services-hit-by-outage-in-paris/, 2023.

[3] Alibaba, "Alibaba cloud health dashboard," https://status.aliyun.com/#/historyEvent, 2025.

[4] S. Zhang, T. Xu, J. Zhu, Y. Sun, P. Jin, B. Shi, and D. Pei, "Privacy-preserving mts anomaly detection for network devices through federated learning," *Information Sciences*, vol. 690, p. 121590, 2025.

[5] Y. Sun, Y. Guo, M. Liang, X. Wen, J. Kuang, S. Zhang, H. Li, K. Xia, and D. Pei, "Multivariate time series anomaly detection based on pre-trained models with dual-attention mechanism," in *2024 IEEE 35th International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 2024, pp. 73–78.

[6] S. Zhang, Y. Ji, J. Luan, X. Nie, Z. Chen, M. Ma, Y. Sun, and D. Pei, "End-to-end automl for unsupervised log anomaly detection," in *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, 2024, pp. 1680–1692.

[7] C. Pei, Z. Liu, J. Li, E. Zhang, L. Zhang, H. Zhang, W. Chen, D. Pei, and G. Xie, "Self-evolutionary group-wise log parsing based on large language model," in *2024 IEEE 35th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2024, pp. 49–60.

[8] Z. Xie, S. Zhang, Y. Geng, Y. Zhang, M. Ma, X. Nie, Z. Yao, L. Xu, Y. Sun, W. Li *et al.*, "Microservice root cause analysis with limited observability through intervention recognition in the latent space," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 6049–6060.

[9] Z. Yao, H. Ye, C. Pei, G. Cheng, G. Wang, Z. Liu, H. Chen, H. Cui, Z. Li, J. Li *et al.*, "Sparserca: Unsupervised root cause analysis in sparse microservice testing traces," in *2024 IEEE 35th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2024, pp. 391–402.

[10] Z. Yu, Q. Ouyang, C. Pei, X. Wang, W. Chen, L. Su, H. Jiang, X. Wang, J. Li, and D. Pei, "Causality enhanced graph representation learning for alert-based root cause analysis," in *2024 IEEE 24th International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*. IEEE, 2024, pp. 77–86.

[11] Z. Yao, C. Pei, W. Chen, H. Wang, L. Su, H. Jiang, Z. Xie, X. Nie, and D. Pei, "Chain-of-event: Interpretable root cause analysis for microservices through automatically learning weighted event causal graph," in *Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering*, 2024, pp. 50–61.

[12] H. Si, C. Pei, Z. Li, Y. Zhao, J. Li, H. Zhang, Z. Diao, J. Li, G. Xie, and D. Pei, "Beyond sharing: Conflict-aware multivariate time series anomaly detection," in *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2023, pp. 1635–1645.

[13] C. Lee, T. Yang, Z. Chen, Y. Su, Y. Yang, and M. R. Lyu, "Heterogeneous anomaly detection for software systems via semi-supervised cross-modal attention," in *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE, 2023, pp. 1724–1736.

[14] Z. Li, N. Zhao, M. Li, X. Lu, L. Wang, D. Chang, X. Nie, L. Cao, W. Zhang, K. Sui *et al.*, "Actionable and interpretable fault localization for recurring failures in online service systems," in *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2022, pp. 996–1008.

[15] Y. Sun, B. Shi, M. Mao, M. Ma, S. Xia, S. Zhang, and D. Pei, "Art: A unified unsupervised framework for incident management in microservice systems," in *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, 2024, pp. 1183–1194.

[16] C. Lee, T. Yang, Z. Chen, Y. Su, and M. R. Lyu, "Eadro: An end-to-end troubleshooting framework for microservices on multi-source data," in *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE, 2023, pp. 1750–1762.

[17] H. Si, J. Li, C. Pei, H. Cui, J. Yang, Y. Sun, S. Zhang, J. Li, H. Zhang, J. Han *et al.*, "Timeseriesbench: An industrial-grade benchmark for time series anomaly detection models," in *2024 IEEE 35th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2024, pp. 61–72.

[18] S. Tuli, G. Casale, and N. R. Jennings, "Tranad: Deep transformer networks for anomaly detection in multivariate time series data," *arXiv preprint arXiv:2201.07284*, 2022.

[19] Z. Wang, C. Pei, M. Ma, X. Wang, Z. Li, D. Pei, S. Rajmohan, D. Zhang, Q. Lin, H. Zhang *et al.*, "Revisiting vae for unsupervised time series anomaly detection: A frequency perspective," in *Proceedings of the ACM Web Conference 2024*, 2024, pp. 3096–3105.

[20] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," *arXiv preprint arXiv:2210.02186*, 2022.

[21] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[22] Z. Gong, G. Wang, Y. Sun, Q. Liu, Y. Ning, H. Xiong, and J. Peng, "Beyond homophily: Robust graph anomaly detection via neural sparsification." in *IJCAI*, 2023, pp. 2104–2113.

[23] W. Zhang, C. Zhang, and F. Tsung, "Grelen: Multivariate time series anomaly detection from the perspective of graph relational learning." in *IJCAI*, 2022, pp. 2390–2397.

[24] Z. Chai, S. You, Y. Yang, S. Pu, J. Xu, H. Cai, and W. Jiang, "Can abnormality be detected by graph neural networks?" in *IJCAI*, 2022, pp. 1945–1951.

[25] M. Ma, S. Zhang, J. Chen, J. Xu, H. Li, Y. Lin, X. Nie, B. Zhou, Y. Wang, and D. Pei, "{Jump-Starting} multivariate time series anomaly detection for online service systems," in *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, 2021, pp. 413–426.

[26] C. Zhao, M. Ma, Z. Zhong, S. Zhang, Z. Tan, X. Xiong, L. Yu, J. Feng, Y. Sun, Y. Zhang *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.

[27] J. Huang, Y. Yang, H. Yu, J. Li, and X. Zheng, "Twin graph-based anomaly detection via attentive multi-modal learning for microservice system," in *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2023, pp. 66–78.

[28] W. Meng, Y. Liu, Y. Zhu, S. Zhang, D. Pei, Y. Liu, Y. Chen, R. Zhang, S. Tao, P. Sun *et al.*, "Loganomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs." in *IJCAI*, vol. 19, no. 7, 2019, pp. 4739–4745.

[29] A. Ng *et al.*, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, no. 2011, pp. 1–19, 2011.

[30] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng *et al.*, "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," in *Proceedings of the 2018 world wide web conference*, 2018, pp. 187–196.

[31] P. Malhotra, L. Vig, G. Shroff, P. Agarwal *et al.*, "Long short term memory networks for anomaly detection in time series," in *Proceedings*, vol. 89, no. 9, 2015, p. 94.

[32] H. Ren, B. Xu, Y. Wang, C. Yi, C. Huang, X. Kou, T. Xing, M. Yang, J. Tong, and Q. Zhang, "Time-series anomaly detection service at microsoft," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 3009–3017.

[33] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.

[34] P. J. Rousseeuw and A. M. Leroy, *Robust regression and outlier detection*. John wiley & sons, 2003.

[35] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104.

[36] J. Xu, H. Wu, J. Wang, and M. Long, "Anomaly transformer: Time series anomaly detection with association discrepancy," *arXiv preprint arXiv:2110.02642*, 2021.