

BEACON: Bridging Multimodal Semantics and Asynchronous Propagation for Cloud System Anomaly Detection

Zhe Li^{1,2}, Tengxue Zhang^{1,2}, Yang Shu¹, Tung Kieu³, Lunting Fan², Yingying Zhang^{2*}, Bin Yang¹

¹East China Normal University, ²Alibaba Cloud Computing, ³Aalborg University

{zheli,txzhang}@stu.ecnu.edu.cn, {yshu,byang}@dase.ecnu.edu.cn, {tungkvt}@cs.aau.dk,
{lunting.fan}@taobao.com, {congrong.zyy}@alibaba-inc.com

Abstract—Modern large-scale cloud systems continuously generate massive and multimodal observability data, such as time-series metrics, log records, and system dependency graphs, which reflect the health state of the system. To ensure reliable cloud system operation, anomaly detection models identify abnormal states by analyzing these multimodal observations. However, existing approaches rely on simple fusion strategies and graph learning, ignoring the semantic gap among modalities and the asynchronous spatio-temporal dependencies across system instances over time. To this end, we propose BEACON, a framework to Bridge multimodal sEmantics and Asynchronous propagation for Cloud system anOmaly detectionN. Specifically, we introduce a Dynamic Interpretable Multimodal Aggregator (DIMA) that leverages an information-bottleneck mechanism to compress information from the semantics and temporal dynamics of metrics, and aligns them with logs in a shared prototype space. Furthermore, we propose an Asynchronous Spatio-Temporal Propagator (ASTP) that captures temporal dependencies with multi-scale temporal attention and models spatial relations with topology-aware aggregation. This enables asynchronous propagation of anomalies across system instances. Extensive experiments on three real-world cloud system datasets demonstrate that BEACON consistently outperforms state-of-the-art baselines in anomaly detection.

Index Terms—Cloud Systems Anomaly Detection, Multimodal Fusion, Spatio-temporal Modeling

I. INTRODUCTION

Modern large-scale cloud systems (e.g., Alibaba Cloud, AWS) host vast numbers of computing machines that support diverse services for businesses and users [1], [2]. The operational states of these systems are continuously monitored and recorded through multimodal observations, including time-series metrics (e.g., CPU usage, memory utilization), unstructured or semi-structured logs (e.g., textual descriptions of error events), and system dependency graphs (e.g., relationships among system instances). System anomalies, such as software bugs, misconfigurations, or sudden traffic spikes, can severely disrupt service availability, propagate across dependent instances, and even trigger large-scale service outages. Therefore, accurate and timely anomaly detection is essential in modern large-scale cloud systems.

Traditional anomaly detection methods primarily rely on pre-defined rules, expert experience, or basic statistical analysis [3]–[5]. Although these approaches are computationally efficient, they are inherently static and struggle to adapt to the highly dynamic and complex runtime behaviors exhibited by modern large-scale cloud systems. With the advancement of machine learning, single-modal methods emerge to model the temporal dynamics in metrics [6]–[10] or parse anomalous events from logs [11]–[13]. However, these approaches miss anomalies that manifest in other modalities, such as slow jobs caused by task backlog reflected in metrics without corresponding log errors. To address this, recent studies explore jointly modeling logs and metrics by leveraging complementary cross-modal information [14]–[16], and further incorporate structural dependencies among system instances through invocation trace analysis or graph learning techniques [17], [18]. Despite these efforts, these methods suffer from significant limitations: they often adopt simple fusion strategies and assume synchronous dependencies, which fall short in real-world large-scale cloud systems. Therefore, building models that can effectively align cross-modal information and capture asynchronous spatio-temporal dependencies presents two challenges:

Lack of fine-grained semantic alignment across modalities during multimodal fusion. In large-scale cloud systems, massive volumes of multimodal observations are continuously generated. While these observations provide a comprehensive view of system health, anomaly-relevant information is often buried deep within a “sea” of voluminous operational data. For example, a “memory leak” anomaly may simultaneously cause a spike in “CPU I/O wait time” and a drop in “cache memory reclamation”, along with a burst of “WARNING” and “SEVERE” log events reporting “probable memory leak”, while most other metrics (i.e., “network throughput”) remain normal (as illustrated in Fig. 1). However, existing approaches typically align multimodal data by timestamps or apply attention to fuse modality-specific representations at the sequence level, leading to coarse-grained multimodal integration [14]. Such modality fusion lacks dynamic temporal interaction within each modality. While some methods attempt fine-grained alignment by extracting event templates [19] or adopting gradient modu-

*Corresponding author.

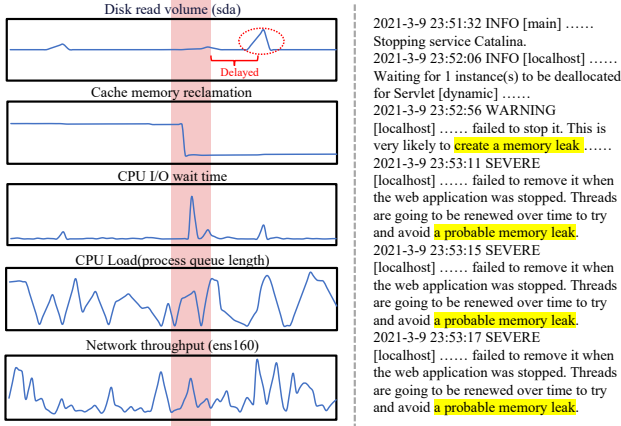


Fig. 1: An example of multimodal observations during an anomaly event on the AIOps2021 dataset. The left side presents several system metrics, where the red region indicates the anomalous time period. The right side shows corresponding logs, and the yellow highlights mark anomalous messages.

lation to balance each modality’s contribution [16], they treat all observations as equally important and cannot distinguish anomaly-relevant information from noise. This underscores the need for a semantically guided fusion mechanism that can adaptively identify anomaly-relevant information across modalities while suppressing irrelevant noise.

Lack of asynchronous spatio-temporal dependency modeling in cloud systems. Instances (e.g., containers or virtual machines) in large-scale cloud systems are interconnected through complex dependencies. In such systems, a fault at one component frequently triggers a ripple effect, propagating along the call chains. Consequently, anomalies may propagate across connected instances in the system graph with varying delays (e.g., an upstream database anomaly causing a downstream API timeout minutes later). Existing approaches primarily characterize these spatio-temporal dependencies using graph learning methods [18] or dynamic service call traces [20]. Moreover, some methods apply aggregation algorithms over system dependency graphs to the evolution of system states, such as random walks [21]. Despite their effectiveness in modeling spatio-temporal dependencies, these methods do not explicitly capture the temporal delays in anomaly propagation across the system. As illustrated in Fig. 1, increased “disk read volume” caused by a “memory leak” may only appear some time after the anomaly begins. Spatio-temporal modeling that aligns observations at the same timestamp may introduce misleading correlations. Consequently, modeling asynchronous spatio-temporal dependencies across system instances is important for improving anomaly detection in cloud systems.

To address these challenges, we propose **BEACON**, a framework to Bridge multimodal semantics and Aynchronous propagation for Cloud system anOomaly detection via semantic multimodal fusion and asynchronous spatio-temporal modeling.

To address the first challenge, we propose a Dynamic

Interpretable Multimodal Aggregator (DIMA). Rather than indiscriminately projecting multimodal observations into a shared latent space, DIMA explicitly leverages fine-grained dynamic semantics to guide the fusion process. Specifically, we first introduce an Information Bottleneck (IB) mechanism that jointly evaluates the intrinsic semantics and the patch-level temporal dynamics of each metric, adaptively filtering out redundant noise while retaining anomaly-relevant information. After selection, highly informative metric embeddings are then fused with log embeddings to form unified multimodal embeddings. Furthermore, we design a semantic prototype layer that projects the fused embeddings into an interpretable semantic space spanned by a set of learnable prototypes, endowing each representation with explicit semantic meaning. Through this process, DIMA effectively prevents sparse anomaly information from being overwhelmed by abundant normal noise, yielding highly distinguishable multimodal representations.

To address the second challenge, we design an Asynchronous Spatio-Temporal Propagator (ASTP) that models asynchronous spatio-temporal dependencies across system instances. ASTP first constructs a heterogeneous spatio-temporal graph based on temporal continuity and system dependency graphs. To capture asynchronous delays, ASTP introduces a multi-scale temporal attention mechanism. It injects a learnable mixture of time-decay kernels as a distance-aware bias into the attention scores, enabling the model to emphasize relevant historical states across different temporal ranges, modeling asynchronous temporal dependencies. To capture spatial dependencies, ASTP introduces a topology-aware spatial aggregation module that leverages spatial attention to account for non-uniform influences among neighboring instances. Through this process, ASTP accurately captures asynchronous spatio-temporal dependencies, improving the reliability of anomaly detection for modern large-scale cloud systems.

In summary, our contributions are as follows:

- We propose **BEACON**, a multimodal anomaly detection framework for cloud systems that detects anomalies by capturing the correlations among multimodal observations, including metrics, logs, and system dependency graphs.
- We employ Dynamic Interpretable Multimodal Aggregator (DIMA), which leverages fine-grained semantics to guide multimodal fusion, suppressing redundant noise while yielding interpretable and unified representations.
- We design an Asynchronous Spatio-Temporal Propagator (ASTP) to model multi-scale temporal dependencies along the system dependency graphs, enhancing anomaly detection reliability in modern large-scale cloud systems.
- Extensive experiments on three real-world datasets demonstrate that **BEACON** outperforms the evaluated baselines across the three datasets.

II. PRELIMINARIES

We formalize the multimodal observations collected from cloud systems and define the anomaly detection task.

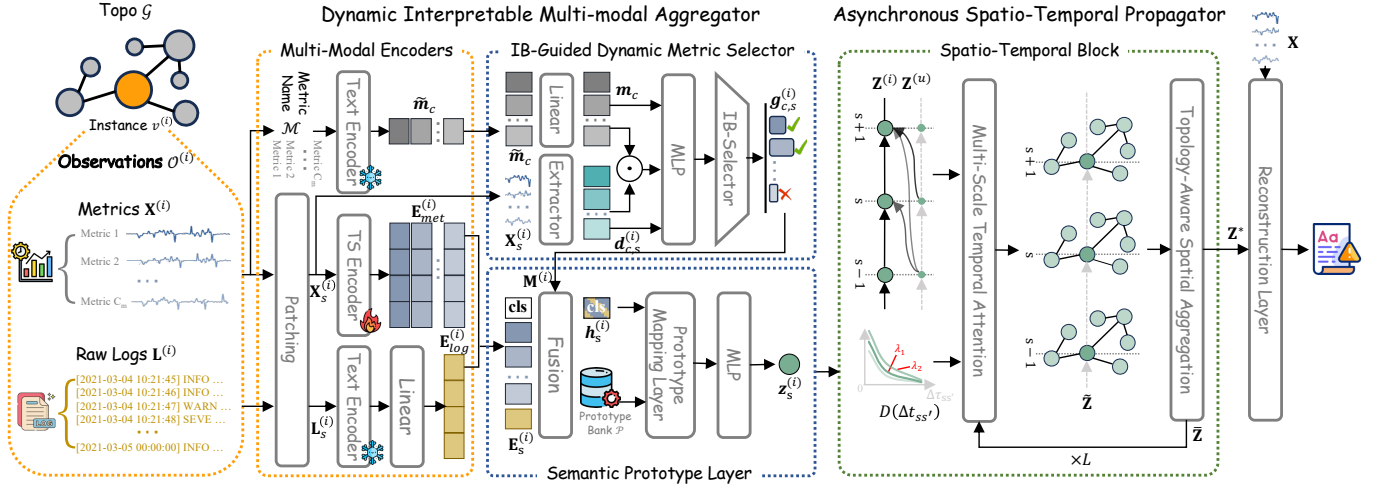


Fig. 2: The overall framework of **BEACON**. On the left, multimodal observations (i.e., metrics, logs, and system dependency graphs) are first encoded into high-dimensional embeddings through modality-specific encoders. DIMA then dynamically distills essential information from massive metrics and aligns metric and log representations in a shared prototype space. Finally, ASTP models asynchronous temporal dependencies and topology-aware spatial interactions across instances to produce representations for anomaly detection.

A. Multimodal Monitoring Data

Definition 1: Metric Sequence. Metrics are numerical measurements (e.g., CPU utilization, memory usage) sampled at regular intervals, which quantify the continuous resource consumption and performance states of the system. Let $\mathcal{M} = \{m_c\}_{c=1}^C$ denote the set of C metric names. Given a time window of length T , the metric sequence is defined as $\mathbf{X} \in \mathbb{R}^{T \times C}$.

Definition 2: Log Sequence. Logs are unstructured or semi-structured textual records generated by system applications in an event-driven manner, capturing discrete execution events and exceptions. Over the time window T , the raw log stream is first aggregated within each time step and then represented as a sequence $\mathbf{L} = \langle \ell_1, \ell_2, \dots, \ell_T \rangle$.

Definition 3: System Dependency Graph. The cloud system structure is modeled as a stable directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Here, $\mathcal{V} = \{v^{(1)}, v^{(2)}, \dots, v^{(N)}\}$ represents the set of N instances deployed in a cloud system, such as containers or virtual machines. $\mathcal{E}$ denotes the dependency relationships among these instances.

B. Problem Formulation

Given multimodal observations collected from a cloud system, including metrics, logs, and a dependency graph, the anomaly detection task aims to identify which time intervals contain anomalous behaviors. Specifically, for each instance in the system, the model takes the historical multimodal observations within a time window as input and learns representations that capture both cross-modal semantics and spatio-temporal dependencies. Based on these representations, the model detects whether each time interval is normal or anomalous.

To formalize this task, given the dependency graph $\mathcal{G}$ and a time window of length T , let $\mathcal{O} = \{\mathcal{O}^{(1)}, \mathcal{O}^{(2)}, \dots, \mathcal{O}^{(N)}\}$

denote the multimodal observations of N instances, where $\mathcal{O}^{(i)} = (\mathbf{X}^{(i)}, \mathbf{L}^{(i)})$ consists of the metric and log sequences of instance $v^{(i)}$. The objective is to learn a mapping function $\mathcal{F}$ parameterized by θ to detect the health state of the cloud system:

$$\mathcal{F}(\mathcal{G}, \mathcal{O}; \theta) \rightarrow \hat{\mathcal{Y}}, \quad (1)$$

where $\hat{\mathcal{Y}} = \{(t_s^{(i)}, t_e^{(i)}, y^{(i)})\}_{i=1}^N$ denotes the set of anomalous events of N instances. In each tuple, $[t_s^{(i)}, t_e^{(i)}]$ represents the temporal interval of the anomalous segment for instance i , and $y^{(i)} \in \mathcal{A}$ denotes its specific anomaly type label from the predefined category set $\mathcal{A}$ (e.g., network latency, memory leak).

III. METHODOLOGY

A. Overview

We propose a multimodal anomaly detection framework, **BEACON**, to accurately identify anomalies in cloud systems by leveraging multimodal observations (i.e., metrics, logs, and system dependency graphs). The overall architecture of **BEACON** is illustrated in Fig. 2. It consists of two key components: a Dynamic Interpretable Multimodal Aggregator (DIMA) and an Asynchronous Spatio-Temporal Propagator (ASTP).

The Dynamic Interpretable Multimodal Aggregator (DIMA) first employs modality-specific encoders to map the raw multimodal observations into high-dimensional embeddings to enrich information. It then applies an Information Bottleneck (IB) mechanism to dynamically compress massive observations into compact features, which reflect the health state of the cloud system. Subsequently, DIMA leverages learnable prototypes to project the compressed metric representations and log embeddings into a shared prototype space, bridging the semantic gap across modalities.

The Asynchronous Spatio-Temporal Propagator (ASTP) models complex asynchronous correlations by constructing a spatio-temporal graph. Specifically, to capture temporal dynamics, ASTP introduces a learnable mixture of multi-scale time-delay kernels to model patterns across diverse temporal ranges. To capture spatial dependencies, it leverages a graph attention network to strengthen relational interactions among connected instances. Finally, the resulting spatio-temporally enriched multimodal representations are effectively utilized for downstream anomaly detection.

B. Dynamic Interpretable Multimodal Aggregator

Cloud systems generate massive volumes of multimodal observations, making it challenging to identify key information related to anomalies. In addition, the heterogeneity between metrics and logs introduces an inherent modality gap, making it difficult to model them in a unified manner. To address these issues, we propose a Dynamic Interpretable Multimodal Aggregator (DIMA), which distills essential information from the raw observations and projects them into a shared semantic space at the patch level.

1) *Multimodal Encoders*: To effectively leverage information from multiple modalities, we use modality-specific encoders to extract distinctive features from each modality. Specifically, for a given instance $v^{(i)}$, we divide its observation sequence over a time window T into non-overlapping patches $\langle O_1^{(i)}, O_2^{(i)}, \dots, O_S^{(i)} \rangle$ following the standard patching mechanism in Triformer [22], where $S = \lceil \frac{T}{W} \rceil$ and W is the predefined patch size. Each patch $O_s^{(i)} = (\mathbf{X}_s^{(i)}, \mathbf{L}_s^{(i)})$ contains metric segment $\mathbf{X}_s^{(i)} \in \mathbb{R}^{C \times W}$ and log segment $\mathbf{L}_s^{(i)} = \langle \ell_{(s-1)W+1}^{(i)}, \dots, \ell_{sW}^{(i)} \rangle$ spanning W time steps.

For a given patch $O_s^{(i)}$, its metrics $\mathbf{X}_s^{(i)}$ are projected into D -dimensional embeddings via a learnable time series encoder:

$$\mathbf{E}_{\text{met},s}^{(i)} = \text{TSEncoder}(\mathbf{X}_s^{(i)}) = \text{MLP}_{\text{met}}(\mathbf{X}_s^{(i)}) \in \mathbb{R}^{C \times D}. \quad (2)$$

To derive compact semantic descriptors from unstructured or semi-structured and voluminous raw logs, the log segment $\mathbf{L}_s^{(i)}$ is first parsed into event templates by Drain [23] with a fixed-depth tree:

$$\mathcal{T}_s^{(i)} = \text{Drain}(\mathbf{L}_s^{(i)}) = \{(\tau_r, f_{s,r}^{(i)})\}_{r=1}^R, \quad (3)$$

where τ_r is the r -th event template, and $f_{s,r}^{(i)}$ denotes the occurrence frequency of event r . The template set is then converted into a single textual description (e.g., “[NullPointerException] ×12; [Connection timeout] ×4; ...”), containing both the semantic content and the relative intensity of log events. This description is mapped to a semantic embedding through a frozen pre-trained text encoder:

$$\tilde{\mathbf{E}}_{\text{log},s}^{(i)} = \text{TextEncoder}(\mathcal{T}_s^{(i)}) \in \mathbb{R}^{D_{\text{text}}}. \quad (4)$$

Since the text encoder output dimension D_{text} differs from the metric embedding dimension D , a lightweight linear layer is applied to align them:

$$\mathbf{E}_{\text{log},s}^{(i)} = \text{Linear}(\tilde{\mathbf{E}}_{\text{log},s}^{(i)}) \in \mathbb{R}^D. \quad (5)$$

2) *IB-Guided Dynamic Metric Selector*: Due to the complex design of cloud systems, massive metrics are often noisy and contain redundant information, making it difficult to accurately model the true state of the cloud system. To address this issue, we introduce an Information Bottleneck (IB) mechanism [24] that dynamically distills essential information by jointly considering the intrinsic semantics and the patch-level temporal dynamics of each metric, producing compact representations that better capture the system state.

Metrics reflect system states that evolve over time, requiring joint modeling of both their intrinsic semantics and temporal dynamics. Let $y_s^{(i)} \in \mathcal{A} \cup \{\text{“Normal”}\}$ denote the state of the instance $v^{(i)}$ during the s -th patch, the objective is to learn a compressed representation $\mathbf{z}_s^{(i)}$ that maximally preserves information of $y_s^{(i)}$, while discarding redundant noise from the raw metric space:

$$\max_{\mathbf{z}_s^{(i)}} I(\mathbf{z}_s^{(i)}; y_s^{(i)}) - \beta I(\mathbf{z}_s^{(i)}; \mathbf{X}_s^{(i)}, \mathcal{M}), \quad (6)$$

the $\beta > 0$ controls the compression rate.

For the semantic prior, the textual metric name m_c is encoded by the TextEncoder($\cdot$) to obtain the metric name embedding $\mathbf{m}_c$:

$$\mathbf{m}_c = \text{Linear}(\text{TextEncoder}(m_c)) \in \mathbb{R}^D. \quad (7)$$

Then, we extract standard statistical features to capture temporal dynamics, including mean μ , variance σ , skewness γ_1 , kurtosis γ_2 , maximum first-order difference $\max |\Delta|$, and coefficient of variation $\sigma_\Delta / (|\mu| + \epsilon)$. These dynamic features are processed by an MLP to obtain the temporal embedding:

$$\mathbf{d}_{c,s}^{(i)} = \text{MLP}_{\text{dyn}}[\mu, \sigma, \gamma_1, \gamma_2, \max |\Delta|, \sigma_\Delta / (|\mu| + \epsilon)]. \quad (8)$$

These two embeddings are fused to produce compact representations that focus on informative metrics. Specifically, we first compute a scalar logit:

$$l_{c,s}^{(i)} = \text{MLP}_{\text{fus}}([\mathbf{m}_c; \mathbf{d}_{c,s}^{(i)}; \mathbf{m}_c \odot \mathbf{d}_{c,s}^{(i)}]) \in \mathbb{R}. \quad (9)$$

where the $\odot$ denotes Hadamard product. The retention probability of the c -th metric is computed as $p_{c,s}^{(i)} = \sigma(l_{c,s}^{(i)})$, where $\sigma(\cdot)$ denotes the standard sigmoid activation function that maps the logit into (0,1). A binary gating variable $g_{c,s}^{(i)} \sim \text{Bernoulli}(p_{c,s}^{(i)})$ is sampled to control whether the metric is retained.

The raw metric embeddings $\mathbf{E}_{\text{met},s}^{(i)}$ are filtered by the learned gates $\mathbf{G}_s^{(i)} = \{g_{c,s}^{(i)}\}_{c=1}^C$ to produce compact metric embeddings that focus on informative information:

$$\hat{\mathbf{E}}_{\text{met},s}^{(i)} = \mathbf{G}_s^{(i)} \cdot \mathbf{E}_{\text{met},s}^{(i)} \in \mathbb{R}^{C \times D}. \quad (10)$$

To realize the compression term in (6), we define the prior distribution $\text{Bernoulli}(\pi)$ following the previous work [25], where $\pi \in [0, 1]$ is a prior retention rate. For each instance i at the s -th patch, we calculate Kullback-Leibler (KL) divergence as the tractable loss to constrain the retention of redundant metrics:

$$\mathcal{L}_{\text{kl},s}^{(i)} = \frac{1}{C} \sum_{c=1}^C D_{\text{KL}}(\text{Bernoulli}(p_{c,s}^{(i)}) \parallel \text{Bernoulli}(\pi)). \quad (11)$$

3) *Semantic Prototype Layer*: To bridge the semantic gap between the compact metric and log embeddings, we introduce learnable prototypes as semantic anchors, mapping heterogeneous modalities into a unified, interpretable semantic space. Specifically, we employ a self-attention to capture interactions between metrics and logs:

$$\text{Self-Attn}(\mathbf{E}_s^{(i)}) = \mathbf{M}_s^{(i)} \cdot \text{Softmax}\left(\frac{\mathbf{Q}_s^{(i)} \mathbf{K}_s^{(i)\top}}{\sqrt{D}}\right) \mathbf{V}_s^{(i)}, \quad (12)$$

where the query, key, and value matrices $\mathbf{Q}_s^{(i)}$, $\mathbf{K}_s^{(i)}$, and $\mathbf{V}_s^{(i)}$ are projected from the multimodal sequence $\mathbf{E}_s^{(i)} = [\langle \text{cls} \rangle; \hat{\mathbf{E}}_{\text{met},s}^{(i)}; \mathbf{E}_{\text{log},s}^{(i)}]$, in which $\langle \text{cls} \rangle$ is a learnable token used to aggregate the multimodal information. The binary vector $\mathbf{M}_s^{(i)} = \{1\} \cup \mathbf{G}_s^{(i)} \cup \{1\} \in \mathbb{R}^{2+C}$ serves as the attention mask to disable interactions for suppressed metrics. We take the hidden state corresponding to the $\langle \text{cls} \rangle$ token as the fused multimodal representation $\mathbf{h}_s^{(i)} \in \mathbb{R}^D$.

Let $\mathcal{P} = \{\mathbf{p}_k\}_{k=1}^K \in \mathbb{R}^{K \times D}$ denotes a bank of learnable prototypes that act as semantic anchors in the shared representation space. Each prototype represents a typical joint pattern of metrics and logs (e.g., a surge in CPU utilization accompanied by program startup logs). The prototype bank is initialized through an LLM-guided procedure (detailed in Section III-D2).

Given the fused representation $\mathbf{h}_s^{(i)}$, we compute its similarity with each prototype to obtain the activation scores:

$$\alpha_{s,k}^{(i)} = \frac{\exp(\cos(\mathbf{h}_s^{(i)}, \mathbf{p}_k) / \tau_{\text{proto}})}{\sum_{k=1}^K \exp(\cos(\mathbf{h}_s^{(i)}, \mathbf{p}_k) / \tau_{\text{proto}})}, \quad (13)$$

where τ_{proto} is a temperature parameter, controlling the sharpness of the similarity distribution. We then project the fused representation into the prototype semantic space by aggregating the prototypes according to these activation scores:

$$\mathbf{h}_{\text{proto},s}^{(i)} = \sum_{k=1}^K \alpha_{s,k}^{(i)} \cdot \mathbf{p}_k \in \mathbb{R}^D. \quad (14)$$

By expressing multimodal observations as combinations of shared prototypes, heterogeneous metric and log features are effectively mapped into a unified semantic space. Finally, we concatenate the two representations and feed the result into an MLP to obtain the final multimodal fusion representation:

$$\mathbf{z}_s^{(i)} = \text{MLP}_{\text{proto}}([\mathbf{h}_s^{(i)}; \mathbf{h}_{\text{proto},s}^{(i)}]) \in \mathbb{R}^D. \quad (15)$$

The resulting representation $\mathbf{z}_s^{(i)}$ is served as the initial node feature for the asynchronous spatio-temporal propagator and used to compute the classification loss:

$$\mathcal{L}_{\text{cls}} = \mathbb{E}(\mathbf{z}_s^{(i)}, y_s) \left[\log q_\theta(y_s^{(i)} | \mathbf{z}_s^{(i)}) \right]. \quad (16)$$

The overall selection loss is formulated by combining the KL-divergence loss and the classification loss as $\mathcal{L}_{\text{select}} = \mathcal{L}_{\text{kl}} + \mathcal{L}_{\text{cls}}$. Furthermore, to maintain the semantic integrity of the prototype bank, we regularize $\mathcal{P}$ with three complementary objectives:

sparsity, diversity, and semantic alignment. The regularization terms are defined as:

$$\begin{aligned} \mathcal{L}_{\text{sparse}} &= \frac{1}{NS} \sum_{i=1}^N \sum_{s=1}^S (1 - \max_{1 \leq k \leq K} \alpha_{k,s}^{(i)}), \\ \mathcal{L}_{\text{div}} &= \frac{2}{K(K-1)} \sum_{1 \leq k < k' \leq K} \max(0, \cos(\mathbf{p}_k, \mathbf{p}_{k'}) - \delta_{\text{proto}}), \\ \mathcal{L}_{\text{align}} &= -\frac{1}{K} \sum_{k=1}^K \cos(\mathbf{p}_k, \mathbf{p}_k^{(\text{text})}). \end{aligned} \quad (17)$$

Here, $\mathcal{L}_{\text{sparse}}$ encourages each patch representation to concentrate on a small number of prototypes, $\mathcal{L}_{\text{div}}$ promotes diversity among prototypes by penalizing highly similar prototype pairs, where δ_{proto} is the similarity threshold, and $\mathcal{L}_{\text{align}}$ preserves the initial semantic meaning of prototypes by aligning them with their textual embeddings $\mathbf{p}_k^{(\text{text})}$ obtained during initialization. The overall prototype regularizer is given by $\mathcal{L}_{\text{proto}} = \mathcal{L}_{\text{sparse}} + \mathcal{L}_{\text{div}} + \mathcal{L}_{\text{align}}$.

C. Asynchronous Spatio-Temporal Propagator

In cloud systems, anomalies rarely remain confined to a single instance. Anomalies in one instance can propagate to related instances with a time lag. To capture such asynchronous dependencies across instances, we propose an Asynchronous Spatio-Temporal Propagator that combines topology-aware message passing with multi-scale temporal modeling.

1) *Spatio-Temporal Graph Construction*: Given the system dependencies graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, we construct a heterogeneous spatio-temporal graph $\mathcal{G}_{\text{st}} = (\mathcal{V}_{\text{st}}, \mathcal{E}_{\text{st}})$ over patch windows. The node set $\mathcal{V}_{\text{st}} = \{v_s^{(i)} \mid i \in [1, N], s \in [1, S]\}$ expands each instance $v^{(i)} \in \mathcal{V}$ across patch windows, where $v_s^{(i)}$ denotes instance $v^{(i)}$ at the s -th patch. Each node is initialized with a multimodal representation $\mathbf{z}_s^{(i)}$ of the corresponding patch. The edge set $\mathcal{E}_{\text{st}}$ includes temporal and spatial edges to capture dependencies between nodes: *Temporal edges* $\mathcal{E}_{\text{tem}} = \{(v_{s'}^{(u)}, v_s^{(i)}) \mid (u, i) \in \mathcal{E} \cup \{(i, i)\}, s - s^* < s' < s\}$ connect each node to the previous-patch nodes of itself and its neighbors $\mathcal{N}(v^{(i)})$ in $\mathcal{G}$, where s^* denotes the maximum temporal range used to construct temporal edges. *Spatial edges* $\mathcal{E}_{\text{spa}} = \{(v_s^{(u)}, v_s^{(i)}) \mid (u, i) \in \mathcal{E}\}$ are based on the system dependencies graph $\mathcal{G}$, capturing the spatial interactions among instances at the current patch.

2) *Multi-Scale Temporal Attention*: Anomalies in distributed cloud systems often propagate across instances with varying time delays, caused by factors such as deferred project scheduling and multi-hop dependency chains. To capture such asynchronous temporal dependencies, we introduce a Multi-Scale Temporal Attention mechanism that models delayed interactions across different time horizons using learnable time-decay kernels.

Specifically, let $\Delta t_{ss'} = s - s'$ denote the time interval between $v_s^{(i)}$ and its temporal neighbor $v_{s'}^{(u)} \in \mathcal{N}_{\text{tem}}(v_s^{(i)})$. We

model a multi-scale temporal factor $D(\Delta t_{ss'})$ as a mixture of M exponential decays with different learnable decay rates λ_m :

$$D(\Delta t_{ss'}) = \sum_{m=1}^M w_m \exp(-\lambda_m \cdot \Delta t_{ss'}), \quad (18)$$

where w_m denotes the learnable scalar weight for the m -th scale. We incorporate the factor as a temporal bias in attention scores to modulate the influence of historical states with different delays:

$$s_{ss'}^{(ui)} = \left(\frac{(\mathbf{W}_Q \mathbf{z}_s^{(i)})^\top (\mathbf{W}_K \mathbf{z}_{s'}^{(u)})}{\sqrt{D}} \right) + D(\Delta t_{ss'}), \quad (19)$$

Here, $\mathbf{W}_Q, \mathbf{W}_K \in \mathbb{R}^{D \times D}$ are the query and key projection matrices. We normalize the scores over the temporal neighborhood using softmax to obtain attention weights $a_{ss'}^{(ui)} = \text{Softmax}(s_{ss'}^{(ui)})$. The temporal representation is then updated as:

$$\tilde{\mathbf{z}}_s^{(i)} = \sum_{v_{s'}^{(u)} \in \mathcal{N}_{\text{tem}}(v_s^{(i)})} a_{ss'}^{(ui)} \mathbf{z}_{s'}^{(u)}. \quad (20)$$

3) *Topology-Aware Spatial Aggregation*: Building upon the temporally updated representations, we further model spatial dependencies among adjacent instances within the same patch using a spatial attention mechanism. Specifically, in the s -th patch, we compute the spatial attention score between an instance $v_s^{(i)}$ and its spatial neighbor $v_{s'}^{(u)} \in \mathcal{N}(v_s^{(i)})$ as:

$$s_s^{(ui)} = \mathbf{a}_{\text{spa}}^\top \text{LeakyReLU}(\mathbf{W}_{\text{spa}} [\tilde{\mathbf{z}}_s^{(u)} \parallel \tilde{\mathbf{z}}_s^{(i)}]), \quad (21)$$

where $\mathbf{W}_{\text{spa}} \in \mathbb{R}^{D \times 2D}$ is the feature transformation matrix, $\mathbf{a}_{\text{spa}} \in \mathbb{R}^D$ is a learnable weight vector, and $\parallel$ denotes concatenation operation. The scores $s_s^{(ui)}$ are normalized with a softmax function over all spatial neighbors to get the attention weight $a_s^{(ui)}$:

$$a_s^{(ui)} = \frac{\exp(s_s^{(ui)})}{\sum_{v_{s'}^{(v)} \in \mathcal{N}(v_s^{(i)})} \exp(s_s^{(vi)})}. \quad (22)$$

Finally, the spatial representation is updated as:

$$\tilde{\mathbf{z}}_s^{(i)} = \sigma \left(\sum_{v_{s'}^{(u)} \in \mathcal{N}(v_s^{(i)})} a_s^{(ui)} \mathbf{W}_V \tilde{\mathbf{z}}_{s'}^{(u)} \right), \quad (23)$$

where $\mathbf{W}_V \in \mathbb{R}^{D \times D}$ is the spatial feature projection matrix, σ denotes a non-linear activation function, and $\tilde{\mathbf{z}}_s^{(i)}$ represents the final output of the spatio-temporal block for instance $v_s^{(i)}$ at the s -th patch.

To model asynchronous spatio-temporal dependencies across the system, we implement the ASTP as a stack of L Spatio-Temporal Blocks. Each block combines multi-scale temporal attention with topology-aware spatial aggregation, updating node representations by integrating information from historical states and neighbors. The output of the l -th block is used as the input to the $(l+1)$ -th block. We omit the layer index (l) in the preceding formulations for notational simplicity. After L layers, we obtain the final multimodal spatio-temporal representations of all instances, denoted as $\mathbf{Z}^* \in \mathbb{R}^{N \times S \times D}$.

Algorithm 1 Pseudo-code of BEACON

Input: Multimodal observations $\mathcal{O}$ and a system dependency graph $\mathcal{G}$.

Warm-up and Representation Collection:

```

1: for  $i = 1$  to  $N$  do
2:   Divide  $\mathcal{O}^{(i)}$  into patches  $\langle \mathcal{O}_s^{(i)} \rangle_{s=1}^S$ ;
3:   for  $s = 1$  to  $S$  do
4:     Encode  $\mathcal{O}_s^{(i)}$  into  $\mathbf{E}_{\text{met},s}^{(i)}$  and  $\mathbf{E}_{\text{log},s}^{(i)}$  using (2)–(5);
5:     Compute the binary gates  $\mathbf{G}_s^{(i)}$  using (7)–(9);
6:     Filter metrics  $\hat{\mathbf{E}}_{\text{met},s}^{(i)}$  by (10);
7:     Fusion multimodal to get  $\mathbf{h}_s^{(i)}$  using (12);
8:   end for
9: end for
10: Update BEACON based on (11) and (17);
11: Apply K-Means to  $\mathbf{h}_s^{(i)}$  to obtain the typical representation set  $\tilde{\mathcal{P}}$ ;
12: return  $\tilde{\mathcal{P}}$ ;

```

Prototype Initialization and Joint Optimization:

```

13: Annotate  $\tilde{\mathcal{P}}$  with  $\{\mathbf{p}_k^{(\text{text})}\}_{k=1}^K$  using a LLM to construct the
    prototype bank  $\mathcal{P}$ ;
14: for  $i = 1$  to  $N$  do
15:   Execute Line 1 of the algorithm;
16:   for  $s = 1$  to  $S$  do
17:     Execute Lines 4–7 to get the fusion representations  $\mathbf{h}_s^{(i)}$ ;
18:     Map  $\mathbf{h}_s^{(i)}$  into prototype space  $\mathbf{z}_s^{(i)}$  on (13)–(15);
19:     Construct spatio-temporal graph  $\mathcal{G}_{\text{st}}$ ;
20:     Compute the multi-scale temporal bias  $D(\Delta t_{ss'})$  on (18);
21:     Model temporal dependencies  $\tilde{\mathbf{z}}_s^{(i)}$  using (19)–(20);
22:     Model spatial dependencies  $\tilde{\mathbf{z}}_s^{(i)}$  using (21)–(23);
23:   end for
24: end for
25: Reconstruct metric sequences  $\hat{\mathbf{X}}$  from (24)–(25);
26: Update BEACON based on (28);
27: return;

```

D. Detection and Optimization

1) *Reconstruction-based Anomaly Detection*: The spatio-temporal representations $\mathbf{Z}^*$ encode fused multimodal information in a shared semantic space. To detect anomalies, we reconstruct the original metric observations from these representations and measure the deviation between the reconstructed and observed values.

Specifically, for each instance $v^{(i)}$, we first expand the fused representations $\mathbf{Z}^* \in \mathbb{R}^{N \times S \times D}$ into C metric-wise representations using an MLP:

$$\mathbf{H} = \text{MLP}_{\text{expand}}(\mathbf{Z}^*) \in \mathbb{R}^{N \times C \times S \times D}. \quad (24)$$

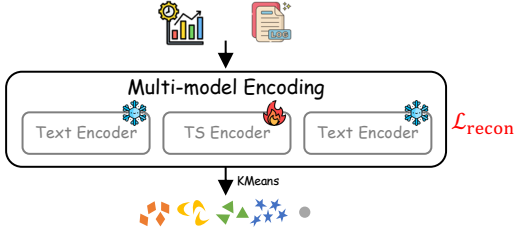
Then, another MLP maps the metric-wise representations back to the original temporal resolution:

$$\hat{\mathbf{X}} = \text{MLP}_{\text{recon}}(\mathbf{H}) \in \mathbb{R}^{N \times C \times T}. \quad (25)$$

During the training phase, the primary objective is to minimize the reconstruction deviation over normal data. The reconstruction loss $\mathcal{L}_{\text{recon}}$ is formally defined as the average Mean Squared Error (MSE) over all instances:

$$\mathcal{L}_{\text{recon}} = \|\mathbf{X} - \hat{\mathbf{X}}\|_2^2. \quad (26)$$

Phase 1 Warm-up and Representation Collection



Phase 2 Prototype Initialization and Joint Optimization

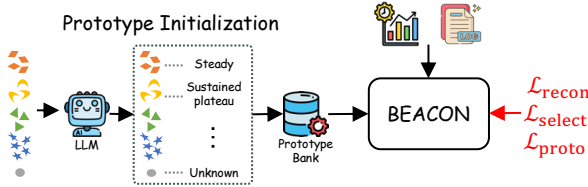


Fig. 3: Overview of the two-phase training pipeline.

During the inference phase, the metric-wise anomaly score at the s -th patch is defined as the reconstructed error $\epsilon_{c,s}^{(i)}$ compared with the original values. For the instance-level anomaly score, we aggregate the metric-wise scores weighted by their dynamic selection probabilities $p_{c,s}^{(i)}$ derived in Section III-B2:

$$\epsilon_s^{(i)} = \sum_{c=1}^C p_{c,s}^{(i)} \cdot \epsilon_{c,s}^{(i)}. \quad (27)$$

We compute an anomaly threshold η on the validation set to identify time intervals when an instance becomes anomalous, i.e., $\epsilon_s^{(i)} > \eta$. The duration, severity, and scope of instance-level anomalies can then be used to assess the overall health status of the cloud system.

2) *Two-Phase Training and Prototype Initialization*: Standard end-to-end optimization of prototypes often suffers from mode collapse and semantic ambiguity. To ensure the prototypes bank captures meaningful, human-readable multimodal information, we design a two-phase training pipeline with an LLM-guided initialization strategy. The overall procedure is summarized in Algorithm 1 and illustrated in Fig. 3.

Phase 1: Warm-up and Representation Collection. We first warm up the network using the reconstruction loss $\mathcal{L}_{\text{recon}}$ to stabilize the multimodal fusion space. We then sample a diverse set of fused patch-level representations $\mathbf{h}_s^{(i)}$ from the training set. K-Means [26] clustering is applied to identify $K - 1$ typical joint patterns of metrics and logs, and the cluster centers are used as the patch-level representation set $\tilde{\mathcal{P}}$. To account for unseen patterns, the K -th representation is reserved as an unknown state and initialized using the mean representation of samples located in low-density regions.

Phase 2: Prototype Initialization and Joint Optimization. For each of the $K - 1$ discovered clusters, we sample several nearest neighbors and feed their raw metric sequences and textual logs into an LLM. Guided by a diagnostic prompt, the LLM

TABLE I: The statistics of evaluation datasets.

Dataset	AIOPS2021	Train-Ticket	D1
Observation	14400	11165	24689
Instance	18	94	14
Frequency	1 min	5 seconds	1 min
Feature	6-90	38	5-36
Anomaly types	6	12	4
Split	4:2:3	2:1:1	5:2:3

generates concise semantic labels, which are then projected into the embedding space as textual anchors $\{\mathbf{p}_k^{(\text{text})}\}_{k=1}^K$. These textual anchors together with $\tilde{\mathcal{P}}$ constitute the prototype bank $\mathcal{P}$. After the prototype initialization, the model is optimized under the following objective using gradient descent with backpropagation:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{select}} + \mathcal{L}_{\text{proto}}. \quad (28)$$

IV. EXPERIMENTS

A. Experimental Setup

1) *Datasets*: To evaluate the performance of **BEACON**, we conduct extensive experiments on three cloud system datasets: AIOPS2021, Train-Ticket, and D1 (collected from an anonymized production environment of a large-scale cloud system). Table I presents detailed statistics of the three datasets.

- **AIOPS2021**¹ is collected from the management system of a large commercial bank and contains monitoring data from 18 instances (i.e., microservices, servers, databases, and Docker containers). The dataset covers the period from January 2021 to June 2021. Failure records are annotated with several fault types, including Memory, Network Fault, JVM-CPU, JVM-Memory, Disk, and CPU. The labeling process is conducted by two experienced operators independently and cross-checked to ensure consistency.
- **Train-Ticket**² is collected from the Train-Ticket system for railway ticketing services, supporting operations such as ticket booking, payment, and rescheduling. FAMOS [14] uses Locust to inject traffic to simulate user requests and employs ChaosBlade to inject anomalies into the system. The data is collected during the last week of June 2024 and includes monitoring data at the host, JVM, and pod levels. In total, 12 types of anomalies are covered, such as CPU Overload, Network Packet Loss, and Service Latency.
- **D1** is collected from an anonymized production environment of a large-scale cloud system under approved internal research procedures and fully anonymized to protect proprietary and sensitive information. To capture realistic failure scenarios while ensuring system stability, the anomalies are accumulated through routine operational telemetry combined with controlled fault-injection exercises and chaos engineering drills. The dataset contains monitoring data from 14 instances and includes four types of anomalies: Failed Requests, CPU Usage Anomaly,

¹<https://aiops-challenge.com>

²<https://modelscope.cn/datasets/dcm552/FAMOS-dataset>

Request Backlog, and Service Unavailability Events. These anomaly categories are labeling classes for benchmarking detection methods and do not reflect the service quality or availability of any production system.

2) *Baselines*: We compare **BEACON** with several state-of-the-art baselines for cloud system anomaly detection, including both single-modal and multimodal methods. Table II summarizes the modalities used by each method.

- ContraAD [27] is a contrastive learning-based anomaly detection method designed for Flink systems. It constructs positive and negative sample pairs from metric time series and applies cross-contrastive learning to learn discriminative representations, enabling the model to distinguish between normal and anomalous patterns without labels.
- TranAD [8] is a transformer-based anomaly detection method for time series. It employs a deep transformer encoder-decoder architecture combined with an adversarial training procedure and a self-conditioning mechanism to amplify reconstruction errors for anomalous points.
- Hades [11] is a semi-supervised anomaly detection method for software systems that handles massive logs. It designs a cross-modal attention mechanism to capture the semantic relationships between different logs, allowing the model to detect anomalies by learning from both labeled and unlabeled log sequences.
- MultiLog [12] is an anomaly detection method specifically designed for distributed database systems. It extracts multivariate features from logs, including log event counts, semantic embeddings, and temporal patterns, and combines them to identify anomalous behaviors in complex distributed environments.
- DiagFusion [19] is a multimodal method that fuses all modalities during data preparation by consolidating them into an event sequence. These events are then encoded with two topology-adaptive graphs for anomaly detection.
- Medicine [16] is a modality-independent anomaly detection framework that identifies anomaly types based on multimodal observations. To preserve the unique characteristics of each modality, it encodes them separately and further adopts multimodal adaptive optimization to balance the learning progress.
- FAMOS [14] is an anomaly detection method for cloud systems that achieves effective multimodal data fusion. It designs modality-specific encoders to extract features from different observations and employs a fusion module to combine these features.
- Eadro [18] presents a multimodal failure diagnosis framework. It designs separate feature extractors for three modalities and then fuses the multimodal features. Eadro learns the dependency-aware status of the system using graph attention networks.
- ART [17] is a unified unsupervised framework for incident management in cloud systems. It integrates multimodal observations and applies representation learning techniques to perform anomaly detection without requiring labeled

anomaly samples.

3) *Evaluation Metrics*: We adopt Precision, Recall, and F1 score as evaluation metrics. Since anomalies in cloud systems typically manifest as continuous segments rather than isolated time points, we perform evaluation at the segment level. Specifically, if any point within an anomalous segment is correctly detected, the entire segment is considered a true positive. Conversely, if no point within an anomalous segment is detected, the entire segment is counted as a false negative. For normal segments, if any point is incorrectly flagged as anomalous, it is treated as a false positive. Based on these segment-level criteria, Precision measures the fraction of detected anomalous segments that are truly anomalous, Recall measures the fraction of actual anomalous segments that are successfully detected, and F1 score is the harmonic mean of Precision and Recall, providing a balanced assessment of overall detection performance.

4) *Implementation Details*: All experiments are conducted using PyTorch [28] in Python 3.10 and executed on an NVIDIA Tesla-A800 GPU. The model is optimized using Adam [29] with an initial learning rate of 0.001 and a batch size of 32. The random seed is set to 2026. The warm-up stage is trained for 3 epochs using the reconstruction loss (26) and the IB-select loss (11) to stabilize the multimodal fusion space. After prototype initialization, the model enters the joint optimization stage and is further trained for 50 epochs. During training, the learning rate is decayed using the StepLR scheduler. The textual embedding model is the pre-trained model “text-embedding-v4” [30]. The time window length is set to 60, the number of prototypes is set to 15, and the number of scales is set to 3.

B. Main Results

Table II presents the overall performance of all methods on the three datasets. **BEACON** achieves the strongest results among the compared methods on these datasets. To verify whether the improvements over the baselines are statistically significant, we conduct t-tests comparing the anomaly scores of our method and the baselines. The resulting p-values are smaller than 0.001, indicating that the improvements are statistically significant.

1) *Compared to single-modal methods*: For single-modal methods, ContraAD and TranAD rely on metric data, while Hades and MultiLog focus on log information. Overall, metric-based methods perform better than log-based approaches on all datasets. For example, ContraAD achieves an F1 score of 0.6271 on AIOps2021, which is higher than Hades and MultiLog. This suggests that system metrics provide relatively stable information for anomaly detection. However, the overall performance of single-modal methods remains limited. Even the best metric-based baseline, ContraAD, achieves only moderate F1 scores on the three datasets. Log-based methods show even larger performance degradation, especially on the D1 dataset, indicating that relying solely on log information is insufficient in complex cloud environments.

In comparison, **BEACON** consistently outperforms all single-modal baselines. Specifically, it achieves a relative average improvement of 28.7% in F1 score over the strongest single-modal method when evaluated on the three datasets.

TABLE II: Overall anomaly detection results. The best-performing results are highlighted in bold, and the second-best results are highlighted in underline.

Method	Modality			AIOps2021			Train-Ticket			D1		
	Metric	Log	Topology	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
ContraAD [27]	✓			0.6527	0.5792	0.6138	0.5146	0.5282	0.5210	0.5424	0.5731	0.5573
TranAD [8]	✓			0.6281	0.5549	0.5892	0.4917	0.5378	0.5137	0.5179	0.5553	0.5359
Hades [11]		✓		0.5138	0.4693	0.4905	0.4615	0.5051	0.4823	0.3671	0.4002	0.3829
MultiLog [12]		✓		0.4786	0.4439	0.4606	0.4937	0.4829	0.4882	0.2576	0.3357	0.2915
DiagFusion [19]	✓	✓		0.2341	0.1655	0.1969	0.1123	0.1408	0.1249	0.3446	0.2891	0.3144
Medicine [16]	✓	✓		0.6829	0.1761	0.2800	0.5012	0.2198	0.3056	0.6194	0.4576	0.5263
FAMOS [14]	✓	✓		0.5235	0.5811	0.5508	0.5732	0.5813	0.5772	0.7654	0.8552	0.8078
Eadro [18]	✓	✓	✓	0.6778	<u>0.5912</u>	<u>0.6315</u>	<u>0.5893</u>	<u>0.5554</u>	<u>0.5718</u>	<u>0.7835</u>	<u>0.7529</u>	<u>0.7679</u>
ART [17]	✓	✓	✓	0.6883	0.3333	0.4491	0.5677	0.4207	0.4833	0.7092	0.4011	0.5124
BEACON	✓	✓	✓	0.7368	0.6123	0.6688	0.6311	0.6274	0.6292	0.8562	0.8890	0.8723

The results indicate that integrating multiple modalities helps capture complementary anomaly information that is difficult to observe from a single data source.

2) *Compared to multimodal methods*: Multimodal methods generally achieve better performance than single-modal approaches, showing that combining metrics, logs, and topology provides richer information for anomaly detection. However, their effectiveness largely depends on the representation and integration of heterogeneous modalities.

Some methods exhibit clear limitations in modeling multimodal observations. For example, DiagFusion transforms multimodal observations into textual event sequences and applies language modeling techniques. This representation simplifies diverse modalities into discrete events and may discard important temporal and numerical characteristics, leading to significant performance degradation. Medicine attempts to balance the influence of different modalities and achieves relatively high precision on some datasets, but its recall remains very low. This suggests that overly constraining modality contributions may make the model conservative and cause many anomalies to be missed. In addition, early fusion strategies such as ART directly concatenate multimodal representations without explicitly addressing modality heterogeneity, which limits the effectiveness of the fused representation. More recent approaches attempt to model interactions across modalities. Methods such as Eadro and FAMOS employ graph attention or cross-attention mechanisms and achieve relatively strong results across datasets. Their performance indicates that properly modeling dependencies among multimodal representations is beneficial for anomaly detection.

As shown in Table II, **BEACON** consistently achieves the best performance across all datasets. Compared with the strongest baselines, **BEACON** improves the F1 score by 5.9% on AIOps2021 and 9.0% on Train-Ticket, and achieves a 7.9% higher F1 score than the best baseline on the D1 dataset. These results suggest that fine-grained semantic fusion, achieved by aligning multimodal representations in a shared space, helps improve the model’s anomaly detection capability.

C. Ablation Studies

To evaluate the contribution of each component in BEACON, we design several variants and report the results in Table III.

1) *w/o intrinsic semantics*: discards the KL-divergence regularization term in (7) of the IB-guided dynamic metric selector. Intrinsic semantics provide stable descriptions of metric meanings, which help the selector distinguish informative metrics from noisy ones. Without this information, the selector relies mainly on observed signals, which makes metric selection less reliable and reduces overall performance.

2) *w/o temporal dynamics*: This variant excludes the dynamic temporal features defined in (8) from the IB-guided dynamic metric selector, retaining only static semantic representations. The results show a noticeable performance drop than removing intrinsic semantics. This suggests that static semantics overlook state-dependent metric relevance, whereas modeling temporal dynamics enables DIMA’s adaptive retrieval.

3) *w/o prototypes*: This variant removes the semantic prototype projection and directly fuses modality features through cross-attention. Prototypes serve as an intermediate semantic space for modality alignment, projecting heterogeneous features into a shared representation space. Without this step, cross-attention operates on modality representations distributed in different feature spaces, which weakens cross-modality alignment and reduces the effectiveness of feature fusion.

4) *w/o multi-scale attention*: This variant replaces the multi-scale temporal delay kernels with a standard transformer. It results in a noticeable performance drop, demonstrating that anomaly propagation in real systems often involves different

TABLE III: Effect of **BEACON** variants on F1 across all datasets.

Method	AIOps2021	Train-Ticket	D1
w/o intrinsic semantics	0.6276	0.6004	0.8577
w/o temporal dynamics	0.6154	0.5913	0.8516
w/o prototypes	0.6253	0.5998	0.8572
w/o multi-scale attention	0.6227	0.6039	0.8468
w/o topology	0.6410	0.6156	0.8614
w/o warm-up	0.6578	0.6198	0.8537
BEACON	0.6688	0.6292	0.8723

temporal delays. Modeling multiple temporal scales is important for capturing these asynchronous dependencies.

5) *w/o topology*: This variant removes topology-aware spatial aggregation and models each node independently. The results show a slight performance decrease after removing spatial modeling. A possible reason is that the multi-scale temporal attention captures cross-instance temporal dependencies over different time ranges, which reflects the spatial relationships among instances. Thus, removing topology-aware modeling has a relatively limited impact on the overall performance.

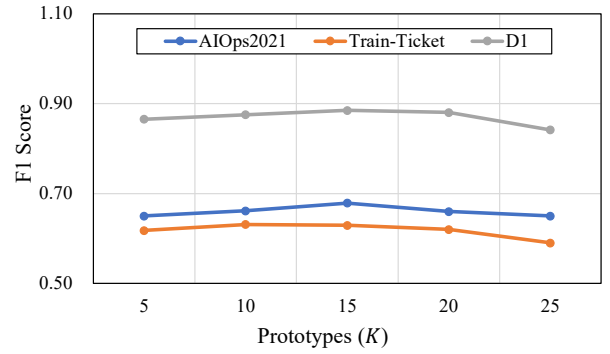
6) *w/o warm-up*: This variant removes the warm-up training stage for prototype initialization and directly trains the full model end-to-end from scratch. The warm-up stage stabilizes the initialization of semantic prototypes. Without this stage, prototypes are randomly initialized, which leads to unstable prototype representations in early training. This weakens their ability to guide modality alignment and feature fusion, leading to reduced performance.

D. Parameter Sensitivity Analysis

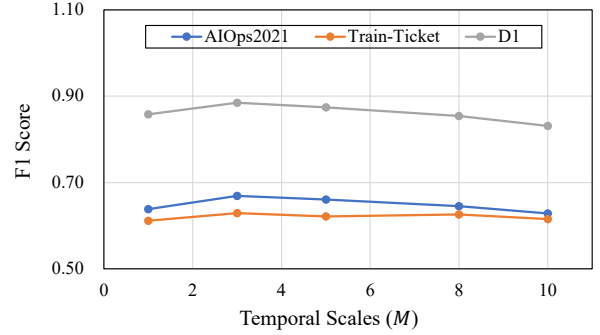
We evaluate the robustness of **BEACON** to two key hyperparameters: the number of prototypes K in DIMA and temporal scales M in ASTP. We vary one parameter at a time while fixing the other, reporting F1 scores across all datasets.

1) *Effect of the number of prototypes K* : To study the impact of the number of prototypes, we set K to $\{5, 10, 15, 20, 25\}$ and report the results in Fig. 4(a). The best performance on AIOps2021 and D1 is achieved around $K = 15$, while Train-Ticket reaches its highest F1 score at $K = 10$. A possible reason is that Train-Ticket is collected with a relatively dense sampling interval (5 seconds), where system states change smoothly between adjacent observations. As a result, a smaller number of prototypes is sufficient to capture the main patterns. Considering that all datasets achieve strong performance around $K = 15$, we set $K = 15$ as the default configuration in the experiments. As K increases, the F1 score first improves and then slightly declines across all datasets. This trend indicates that an appropriate number of prototypes helps capture diverse patterns in multimodal observations and improves representation learning for anomaly detection, while too many prototypes may introduce redundant patterns and degrade performance. Therefore, a moderate number of prototypes provides a good balance between representation capacity and model robustness.

2) *Effect of the number of temporal scales M* : To study the impact of the number of temporal scales, we set M to $\{1, 3, 5, 8, 10\}$ and report the results in Fig. 4(b). As M increases from 1 to 3, the F1 score on all three datasets improves, indicating that modeling multiple temporal ranges helps capture asynchronous temporal dependencies. The best performance is achieved around $M = 3$, suggesting that the model can effectively capture short-, medium-, and long-term temporal dependencies with a moderate number of temporal scales. This design enables the model to adaptively aggregate temporal information at different ranges and improves anomaly detection performance. When M further increases, the performance slightly declines, likely due to redundant temporal information and increased



(a) Effect of the number of prototypes K .



(b) Effect of the number of temporal scales M .

Fig. 4: Results of **BEACON** with different parameters on F1 score across three datasets.

model complexity. In contrast, the performance on the Train-Ticket dataset changes only marginally as M increases. This is consistent with the observation in Section IV-D1, likely because the dataset is collected with a fine-grained sampling interval, where observations are densely sampled, and system states exhibit limited variation within the same temporal windows.

E. Efficiency

Fig. 6 compares different methods in terms of F1 score, training time, and parameter size. The results show clear differences in the trade-off between detection performance, computational cost, and model complexity.

BEACON achieves the best overall detection performance while maintaining moderate training time and a relatively small parameter size. Some baselines obtain competitive F1 scores but rely on significantly larger models or longer training time, which increases the computational overhead. Several lightweight approaches reduce parameter size or training time, but their detection performance is noticeably lower. This suggests that simply reducing model complexity often leads to a loss in accuracy. In contrast, **BEACON** maintains a compact model while still achieving strong performance. Overall, the results indicate that **BEACON** achieves a better balance between effectiveness and efficiency.

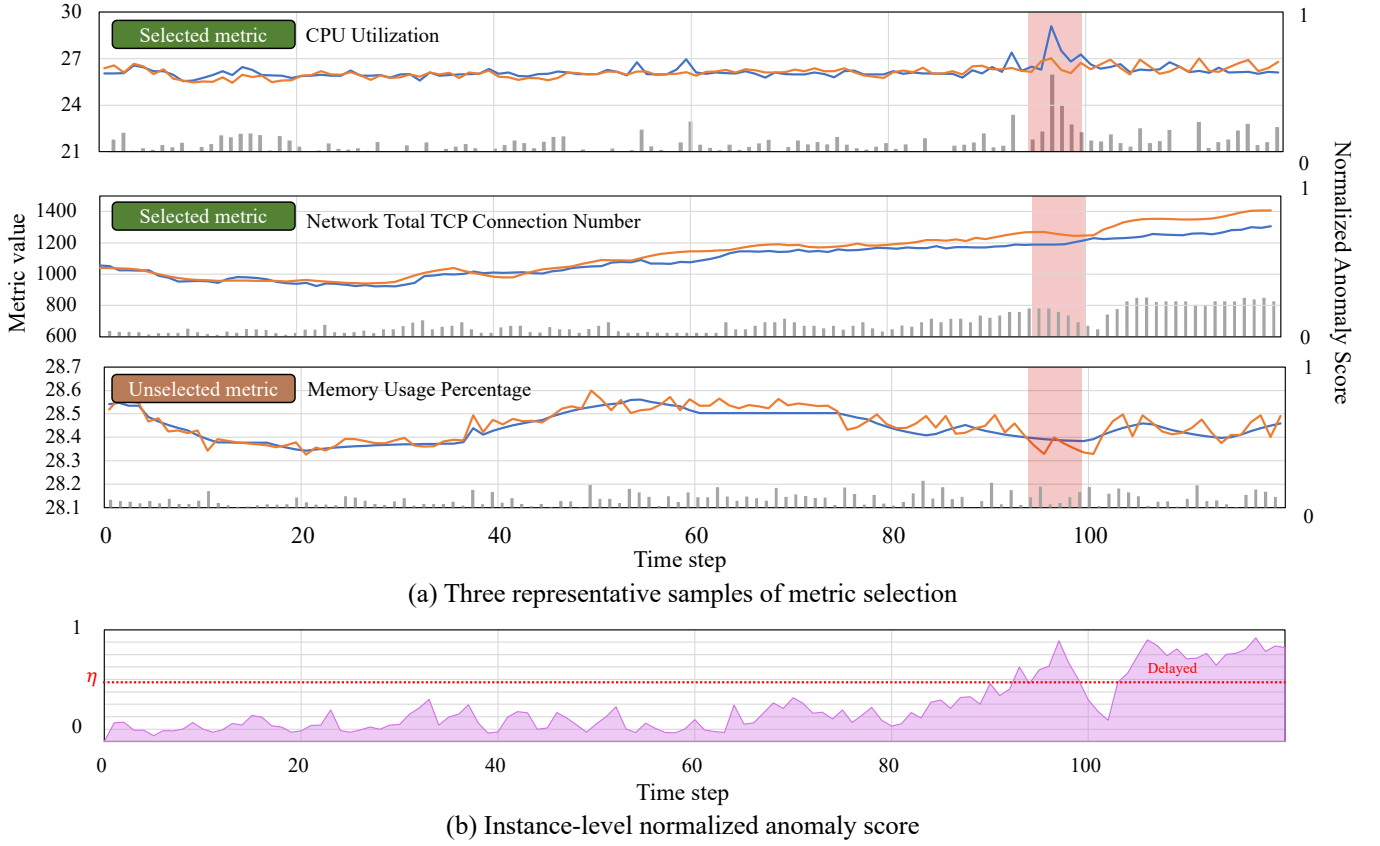


Fig. 5: Visualization of metric selection and instance-level anomaly detection. (a) Three representative metrics from the same instance. Blue lines and orange lines denote observed and reconstructed values, respectively. Gray bars represent normalized anomaly scores, and the red shaded region indicates the ground-truth anomaly interval. (b) The aggregated instance-level normalized anomaly score, η is the anomaly threshold on the validation set.

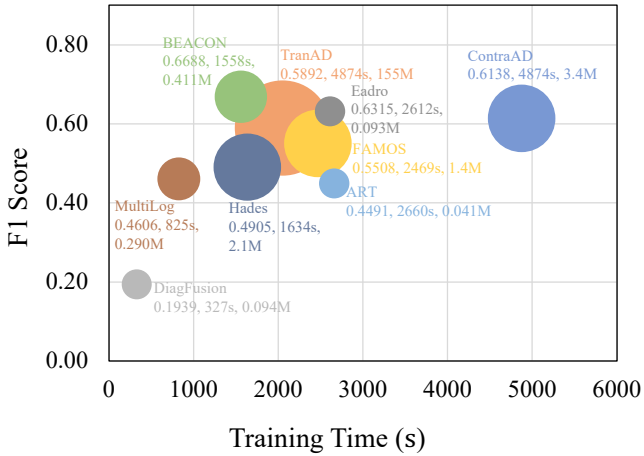


Fig. 6: Comparison of F1 Score, training time, and parameter sizes on the AIOps2021 dataset.

F. Case Studies

To demonstrate the effectiveness of the main components in **BEACON**, which identify informative metrics from massive

multimodal observations for fusion with logs and effectively model asynchronous spatio-temporal dependencies, we provide a “*Network Packet Loss*” anomaly sample in Fig. 5.

We observe that “*Network Packet Loss*” triggers request retries and connection timeouts, resulting in a significant increase in “*CPU utilization*”. By combining the temporal pattern of “*CPU Utilization*” with its semantic meaning conveyed by the metric name, the aggregator correctly identifies it as an anomaly-relevant metric. Notably, the large number of repeated requests also leads to a delayed increase in “*Network Total TCP Connection Number*”. This delayed change is also captured by the model, and the corresponding metric is correctly identified as anomaly-related. Similarly, “*Memory Usage Percentage*” shows no significant variation in its temporal pattern and is excluded. Fig. 5(b), the instance-level anomaly score remains low during normal periods and rises sharply within the ground-truth anomaly interval, and anomalies caused by delayed effects are also correctly identified. This is because the aggregator assigns high weights to the selected anomaly-relevant metrics while suppressing the influence of irrelevant ones, allowing the aggregated score to faithfully reflect the actual anomaly occurrence without being diluted by noise.

This case illustrates that the DIMA can effectively distinguish anomaly-relevant metrics from irrelevant ones. The joint use of temporal dynamics and metric semantics enables more precise metric selection than relying on either modality alone, reducing false positives caused by coincidental fluctuations and improving the interpretability of anomaly detection results. Meanwhile, the ASTP captures asynchronous spatio-temporal dependencies among system instances through explicit multi-scale temporal modeling and spatial graph learning.

V. RELATED WORK

A. Anomaly Detection

Traditional anomaly detection methods aim to identify observations that deviate from the expected data generation process in time series. Early approaches mainly rely on classical statistical and signal processing techniques. Representative methods include density- and clustering-based approaches such as LOF [3] and CBLOF [31], and histogram-based methods like HBOS [4]. Other methods use statistical thresholds or signal transformations, including z-score-based detection, matrix profile, and discrete wavelet transform [5], [32]–[34]. These approaches typically detect anomalies based on statistical deviation or pattern mismatch in time series, struggling to capture complex system behaviors and high-dimensional monitoring signals in modern cloud environments.

With the advancement of deep learning techniques, numerous methods have been proposed to model single-modal observability data. Some methods adopt autoencoders and transformers to detect deviations in multivariate time series [8], [27], [35]–[41], while others parse logs into event sequences and perform anomaly detection based on sequential patterns [11], [12], [42]–[44]. Although these single-modal approaches achieve promising results within their respective data sources, they inherently miss anomaly signals that only manifest in other modalities, limiting their overall detection capability.

Recently, integrating multimodal monitoring data has attracted increasing attention in anomaly detection [45]–[50]. Some methods encode each modality independently and perform early fusion by concatenating embeddings [17], while others apply late fusion to combine modality-specific representations [16]. Cross-modal graph structures have also been used to model correlations across modalities [18]. However, these approaches often fail to capture the heterogeneity across modalities. Although some multimodal time series models attempt to align time series with text [48], they assume the availability of paired textual descriptions and are not directly applicable to cloud system scenarios where metrics and logs exhibit distinct structures and semantics. In contrast, DIMA filters redundant information through an IB mechanism and aligns multimodal representations in a shared prototype space.

B. Spatio-Temporal Modeling

Spatio-temporal modeling aims to capture both temporal dynamics and spatial dependencies in data. Early efforts primarily modeled the temporal dynamics of time series such as RNNs, temporal convolution networks (TCNs), and Transformers,

which capture temporal correlations in system behaviors [51], [52]. Methods such as PatchTST [53] and TriFormer [22] learn temporal dependencies by modeling local temporal patterns within segmented sequences using Transformer architectures. Other approaches, including TimesNet [54] and TFMixer [55], capture temporal dynamics through frequency-domain decomposition, such as FFT or Wavelet Transform, to model multi-scale temporal patterns. While effective for temporal modeling, these approaches treat system entities independently and ignore structural relationships among instances.

To address this limitation, some studies introduce graph-based models that incorporate spatial dependencies [56]–[62]. For example, Orca [63] leverages Z-order curves to capture grid-based spatial proximity, while MOST [64] employs a multimodal guided spatial aggregation module with a mixture-of-experts architecture to model spatio-temporal relationships among neighboring sensors. In cloud systems, spatial relationships among instances are reflected through service traces and system dependency graphs. Several methods detect system anomalies by identifying sudden increases in latency from trace data [16], while others model the spatial propagation of anomalies based on classical graph algorithms such as cut trees [17] and random walks [21]. However, these methods assume that anomalies manifest simultaneously across related instances, overlooking the fact that in complex cloud systems, anomalies often propagate as cascading failures with varying temporal delays. In contrast, the proposed ASTP module models asynchronous spatio-temporal dependencies across instances through learnable multi-scale delay kernels and graph learning.

VI. CONCLUSION

We introduce BEACON, a multimodal anomaly detection framework for cloud systems that integrates time series metrics, unstructured or semi-structured logs, and system dependency graphs. **BEACON** addresses both challenges through two core components. The Dynamic Interpretable Multimodal Aggregator (DIMA) compresses metric observations via an information bottleneck mechanism and aligns the informative metric embeddings with log embeddings in a shared prototype space for effective multimodal fusion. The Asynchronous Spatio-Temporal Propagator (ASTP) further models dependencies across temporal dynamics and system structure through multi-scale temporal modeling and topology-aware aggregation, allowing anomaly signals to propagate across instances and time. Experiments on three real-world cloud datasets show that **BEACON** consistently outperforms existing methods in anomaly detection. These results highlight the effectiveness of fine-grained multimodal semantic fusion and modeling asynchronous spatio-temporal dependencies in complex cloud environments.

ACKNOWLEDGMENT

This work was partially supported by the National Natural Science Foundation of China (62406112) and Alibaba Innovative Research Program.

REFERENCES

- [1] Z. Pan, Y. Wang, Y. Zhang, S. B. Yang, Y. Cheng, P. Chen, C. Guo, Q. Wen, X. Tian, Y. Dou, Z. Zhou, C. Yang, A. Zhou, and B. Yang, "Magicscaler: Uncertainty-aware, predictive autoscaling," *Proc. VLDB Endow.*, pp. 3808–3821, 2023.
- [2] B. Ouyang, Y. Zhang, H. Cheng, Y. Shu, C. Guo, B. Yang, Q. Wen, L. Fan, and C. S. Jensen, "Rcrank: Multimodal ranking of root causes of slow queries in cloud database systems," *Proc. VLDB Endow.*, vol. 18, no. 4, pp. 1169–1182, 2024.
- [3] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *SIGMOD*, 2000, pp. 93–104.
- [4] M. Goldstein and A. Dengel, "Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm," *KI - German Conference on Artificial Intelligence: poster and demo track*, vol. 1, 2012.
- [5] C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, D. F. Silva, A. Mueen, and E. J. Keogh, "Matrix profile i: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets," in *ICDM*, 2016, pp. 1317–1322.
- [6] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "USAD: unsupervised anomaly detection on multivariate time series," in *SIGKDD*, 2020, pp. 3395–3404.
- [7] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pai, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *SIGKDD*, 2019.
- [8] S. Tuli, G. Casale, and N. R. Jennings, "Tranad: Deep transformer networks for anomaly detection in multivariate time series data," *Proc. VLDB Endow.*, vol. 15, no. 6, pp. 1201–1214.
- [9] Z. Li, J. Tian, H. Miao, Z. Lei, C. Guo, and B. Yang, "Tiweaver: Unified temporal dynamics modeling via contextual patching," in *SIGKDD*, 2026, pp. 2933–2944.
- [10] T. Zhang, B. Ouyang, Y. Shu, X. Chen, C. Guo, and B. Yang, "Swifts: A swift selection framework for time series pre-trained models via multi-task meta-learning," in *ICLR*, 2026.
- [11] 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 *ICSE*, 2023, p. 1724–1736.
- [12] L. Zhang, T. Jia, M. Jia, Y. Li, Y. Yang, and Z. Wu, "Multivariate log-based anomaly detection for distributed database," in *SIGKDD*, 2024, pp. 4256–4267.
- [13] H. Xie, M. Zhou, Q. Yu, J. Yu, Z. Sheng, H. Xie, and D. Lian, "M²-MFP: A multi-scale and multi-level memory failure prediction framework for reliable cloud infrastructure," in *SIGKDD*, 2025, pp. 5083–5094.
- [14] C. Duan, Y. Yang, T. Jia, G. Liu, J. Liu, H. Zhang, Q. Zhou, Y. Li, and G. Huang, "Famos: Fault diagnosis for microservice systems through effective multi-modal data fusion," in *ICSE*, 2025, pp. 2613–2624.
- [15] C. Zhao, M. Ma, Z. Zhong, S. Zhang, Z. Tan, X. Xiong, L. Yu, J. Feng, Y. Sun, Y. Zhang, D. Pei, Q. Lin, and D. Zhang, "Robust multimodal failure detection for microservice systems," in *SIGKDD*, 2023, pp. 5639–5649.
- [16] L. Tao, S. Zhang, Z. Jia, J. Sun, M. Ma, Z. Li, Y. Sun, C. Yang, Y. Zhang, and D. Pei, "Giving every modality a voice in microservice failure diagnosis via multimodal adaptive optimization," in *ASE*, 2024, pp. 1107–1119.
- [17] 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 *ASE*, 2024, pp. 1183–1194.
- [18] 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 *ICSE*, 2023, pp. 1750–1762.
- [19] S. Zhang, P. Jin, Z. Lin, Y. Sun, B. Zhang, S. Xia, Z. Li, Z. Zhong, M. Ma, W. Jin, D. Zhang, Z. Zhu, and D. Pei, "Robust failure diagnosis of microservice system through multimodal data," *IEEE Trans. Serv. Comput.*, vol. 16, no. 6, pp. 3851–3864, 2023.
- [20] X. Guo, X. Peng, H. Wang, W. Li, H. Jiang, D. Ding, T. Xie, and L. Su, "Graph-based trace analysis for microservice architecture understanding and problem diagnosis," in *FSE*, 2020, pp. 1387–1397.
- [21] Y. Sun, X. Pan, X. Xiong, L. Tao, J. Wang, S. Zhang, Y. Yuan, Y. Li, and K. Jian, "Clustercca: An end-to-end approach for network fault localization and classification for HPC system," in *ISSRE*, 2025, pp. 347–358.
- [22] R. Cirstea, C. Guo, B. Yang, T. Kieu, X. Dong, and S. Pan, "Triformer: Triangular, variable-specific attentions for long sequence multivariate time series forecasting," in *IJCAI*, 2022, pp. 1994–2001.
- [23] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An online log parsing approach with fixed depth tree," in *ICWS*, 2017, pp. 33–40.
- [24] N. Tishby and N. Zaslavsky, "Deep learning and the information bottleneck principle," in *ITW*, 2015, pp. 1–5.
- [25] S. Miao, M. Liu, and P. Li, "Interpretable and generalizable graph learning via stochastic attention mechanism," in *ICML*, 2022, pp. 15 524–15 543.
- [26] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, pp. 129–136, 1982.
- [27] Z. Zhuang, Y. Zhang, K. Zhao, C. Guo, B. Yang, Q. Wen, and L. Fan, "Noise matters: Cross contrastive learning for flink anomaly detection," *Proc. VLDB Endow.*, vol. 18, no. 4, p. 1159–1168, 2024.
- [28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *NeurIPS*, vol. 32, 2019.
- [29] D. P. Kingma, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [30] OpenAI, "Openai embeddings api," <https://platform.openai.com/docs/guides/embeddings>, 2024.
- [31] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern recognition letters*, vol. 24, no. 9–10, pp. 1641–1650, 2003.
- [32] A. Bhatnagar, P. Kassianik, C. Liu, T. Lan, W. Yang, R. Cassius, D. Sahoo, D. Arpit, S. Subramanian, G. Woo, A. Saha, A. K. Jagota, G. Gopalakrishnan, M. Singh, K. C. Krithika, S. Maddineni, D. Cho, B. Zong, Y. Zhou, C. Xiong, S. Savarese, S. C. H. Hoi, and H. Wang, "Merlion: A machine learning library for time series," 2021.
- [33] M. Thill, W. Konen, and T. Bäck, "Time series anomaly detection with discrete wavelet transforms and maximum likelihood estimation," in *ITISE*, vol. 2, 2017, pp. 11–23.
- [34] P. Boniol and T. Palpanas, "Series2graph: Graph-based subsequence anomaly detection for time series," *Proc. VLDB Endow.*, vol. 13, no. 11, pp. 1821–1834, 2020.
- [35] Z. Chen, J. Liu, Y. Su, H. Zhang, X. Ling, and M. R. Lyu, "Adaptive performance anomaly detection for online service systems via pattern sketching," in *ICSE*, 2022, pp. 61–72.
- [36] 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 *SIGKDD*, 2019, pp. 3009–3017.
- [37] Z. Li, Y. Zhao, J. Han, Y. Su, R. Jiao, X. Wen, and D. Pei, "Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding," in *SIGKDD*, 2021, pp. 3220–3230.
- [38] Z. Li, X. Qiu, P. Chen, Y. Wang, H. Cheng, Y. Shu, J. Hu, C. Guo, A. Zhou, C. S. Jensen, and B. Yang, "Tsfm-bench: A comprehensive and unified benchmark of foundation models for time series forecasting," in *SIGKDD*, 2025, pp. 5595–5606.
- [39] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "USAD: unsupervised anomaly detection on multivariate time series," in *SIGKDD*, 2020, pp. 3395–3404.
- [40] Y. Chen, S. Huang, Y. Cheng, P. Chen, Z. Rao, Y. Shu, B. Yang, L. Pan, and C. Guo, "Aimts: Augmented series and image contrastive learning for time series classification," in *ICDE*, 2025, pp. 1952–1965.
- [41] B. Zhang, T. Kieu, X. Qiu, C. Guo, J. Hu, A. Zhou, C. S. Jensen, and B. Yang, "An encode-then-decompose approach to unsupervised time series anomaly detection on contaminated training data," in *ICDE*, 2026, pp. 919–933.
- [42] T. Jia, Y. Li, Y. Yang, G. Huang, and Z. Wu, "Augmenting log-based anomaly detection models to reduce false anomalies with human feedback," in *SIGKDD*, 2022, pp. 3081–3089.
- [43] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in *SIGSAC*, 2017.
- [44] R. Chen, S. Zhang, D. Li, Y. Zhang, F. Guo, W. Meng, D. Pei, Y. Zhang, X. Chen, and Y. Liu, "Logtransfer: Cross-system log anomaly detection for software systems with transfer learning," in *ISSRE*, 2020, pp. 37–47.
- [45] 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 *ICSE*, 2023, pp. 1724–1736.
- [46] C. Zhang, X. Peng, C. Sha, K. Zhang, Z. Fu, X. Wu, Q. Lin, and D. Zhang, "Deeptralog: Trace-log combined microservice anomaly detection through graph-based deep learning," in *ICSE*, 2022, pp. 623–634.
- [47] N. Zhao, J. Chen, Z. Yu, H. Wang, J. Li, B. Qiu, H. Xu, W. Zhang, K. Sui, and D. Pei, "Identifying bad software changes via multimodal anomaly detection for online service systems," in *ESEC/FSE*, 2021, p. 527–539.
- [48] S. Hu, J. Jin, Y. Shu, P. Chen, B. Yang, and C. Guo, "Towards multimodal time series anomaly detection with semantic alignment and condensed interaction," in *ICLR*, 2026.

- [49] H. Liu, S. Xu, Z. Zhao, L. Kong, H. Kamarthi, A. B. Sasanur, M. Sharma, J. Cui, Q. Wen, C. Zhang, and B. A. Prakash, "Time-mmd: multi-domain multimodal dataset for time series analysis," in *NeurIPS*, 2024.
- [50] P. Liu, H. Guo, T. Dai, N. Li, J. Bao, X. Ren, Y. Jiang, and S. Xia, "CALF: aligning llms for time series forecasting via cross-modal fine-tuning," in *AAAI*, 2025, pp. 18 915–18 923.
- [51] X. Wu, D. Zhang, M. Zhang, C. Guo, B. Yang, and C. S. Jensen, "Autocts+: Joint neural architecture and hyperparameter search for correlated time series forecasting," *Proc. ACM Manag. Data*, vol. 1, no. 1, 2023.
- [52] H. Gao, W. Shen, X. Qiu, R. Xu, B. Yang, and J. Hu, "Ssd-ts: Exploring the potential of linear state space models for diffusion models in time series imputation," in *SIGKDD*, 2025, p. 649–660.
- [53] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," in *ICLR*, 2023.
- [54] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," in *ICLR*, 2022.
- [55] X. Qiu, K. Yan, X. Liu, X. Wu, and J. Hu, "Bridging time and frequency: A joint modeling framework for irregular multivariate time series forecasting," in *ICML*, 2026.
- [56] R.-G. Cirstea, C. Guo, and B. Yang, "Graph attention recurrent neural networks for correlated time series forecasting," in *SIGKDD*, 2019.
- [57] J. Tian, Y. Liang, R. Xu, P. Chen, C. Guo, A. Zhou, L. Pan, Z. Rao, and B. Yang, "Air quality prediction with physics-guided dual neural odes in open systems," in *ICLR*, 2025.
- [58] Y. Fang, Y. Liang, B. Hui, Z. Shao, L. Deng, X. Liu, X. Jiang, and K. Zheng, "Efficient large-scale traffic forecasting with transformers: A spatial data management perspective," in *SIGKDD*, 2023.
- [59] J. Tian, Y. Ding, R. Xu, H. Miao, C. Guo, and B. Yang, "Arrow: An adaptive rollout and routing method for global weather forecasting," in *ICLR*, 2026.
- [60] R. Xu, H. Cheng, C. Guo, H. Gao, J. Hu, S. B. Yang, and B. Yang, "Mm-path: Multi-modal, multi-granularity path representation learning," in *SIGKDD*, 2025, pp. 1703–1714.
- [61] J. Mei, J. Tian, R. Xu, H. Wei, C. Guo, and B. Yang, "Find3: A dual 3d state space model with dynamic hypergraph for financial stock prediction," in *CIKM*, 2025, pp. 2084–2094.
- [62] X. Zhao, J. Tian, and H. Miao, "Pivot-centric trajectory prediction: Bridging long horizons via dynamical guidance," in *IJCAI*, 2026, pp. 3276–3284.
- [63] Z. Li, R. Xu, J. Hu, Z. Peng, X. Lu, C. Guo, and B. Yang, "Ocean significant wave height estimation with spatio-temporally aware large language models," in *CIKM*, 2024.
- [64] R. Xu, J. Chen, J. Tian, C. Guo, and B. Yang, "Most: A foundation model for multi-modality spatio-temporal traffic prediction," in *SIGKDD*, 2026, pp. 1716–1727.