M^3-DD: DETECTING AND LOCALIZING MULTIPLE CONCURRENT PERFORMANCE DEGRADATIONS IN CLOUD SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Performance degradation is one of the most pressing threats to the reliability of modern cloud systems, yet its root causes are latent, entangled, and rarely confined to a single metric. Many available diagnostic benchmarks reduce degradation to binary or single-fault labels rather than multiple time-bounded events that may overlap. In this paper, we present the Multi-modal Concurrent Degradation Dataset (MCDD), a VM-level benchmark assembled from production observations, with 22 metric channels, a vocabulary of 122 low-level anomaly-event types, and interval annotations for 25 degradation categories. These annotations preserve multiple overlapping events. We further propose M^3-DD (Multi-Modal Multi-instance Degradation Detector), which uses object queries to jointly classify and localize overlapping degradation intervals. Its interval-aware attention fuses metric time series with low-level anomaly events, allowing the decoder to represent their temporal relationships before predicting event sets. We evaluate native concurrent event sets using Coverage–Localization Balance (CLB), a composite of event-level alert coverage and localization precision, alongside confidence-ranked average mAP. On MCDD, M^3-DD attains the highest CLB (0.4180) and average mAP (0.1967) among the evaluated native-view methods. These results highlight M^3-DD's ability to detect and localize concurrent degradations.

1 INTRODUCTION

The growing complexity of large-scale, microservice-oriented cloud systems has made performance degradation one of the most pressing threats to service reliability (Huang et al., 2017; Xu et al., 2020). It is commonly defined as a significant deviation of key performance indicators (KPIs) from their service-level agreement (SLA) baselines, with root causes that are latent and entangled—spanning low-level resource contention, such as the "noisy neighbor" effect (Pons et al., 2023), and high-level cascading failures across microservices (Zhang et al., 2025). Such degradation is rarely isolated: a minor latency spike can propagate along dependency chains, leading to cascading outages and economic loss.

To address this challenge, a large body of observability-driven diagnostics has been built on metrics, logs, and traces (Gu et al., 2023; Soldani & Brogi, 2022), spanning statistical process control, time-series anomaly detection (TSAD), and machine learning. Representative examples include reconstruction-based models (Chouliaras & Sotiriadis, 2021; Xu et al., 2018), association-discrepancy Transformers (Xu et al., 2022), vision-style detectors (Wang et al., 2025), correlation-mining systems (Gan et al., 2021a), and causal-inference frameworks (Tunde-Onadele et al., 2024); see (Choi et al., 2021; Soldani & Brogi, 2022) for reviews. More recent multimodal, LLM-enhanced, and visual-analytics diagnostics improve interpretability (Zhong et al., 2026; Xu et al., 2020), yet these methods perform best in controlled, single-fault settings.

Despite this progress, many commonly used benchmarks and detectors reduce performance degradation to binary decisions, point labels, or co-occurring labels attached to a whole window, rather than *multiple time-bounded events that overlap and interleave along the time axis* (Figure 1a–d). A window containing retransmission degradation, context-switch degradation, and connection timeouts illustrates why their individual intervals matter (Figure 1). Single-label and window-level eval-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

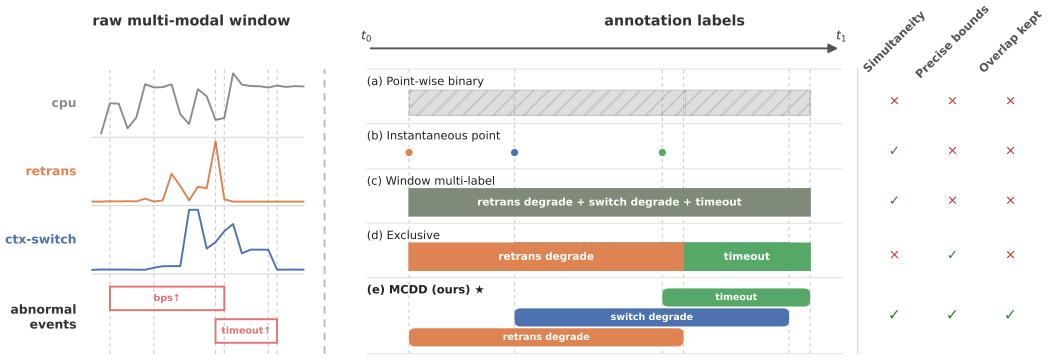

Figure 1: The same raw KPI window under five annotation paradigms. *Left*: multivariate metric curves comprising CPU, retransmission, and context-switch time series, alongside abnormal events characterized by rising bps and timeout. *Right*: how each paradigm annotates this window—(a) point-wise binary, (b) instantaneous points, (c) window-level multi-label set, (d) exclusive non-overlapping segments, and (e) MCDD's recorded multi-concurrent intervals. The latter preserves event categories, recorded start/end times, and overlap structure; boundaries are refined through human curation to ensure annotation quality (details in Appendix A).

uation also cannot distinguish a missed concurrent event from an incorrectly classified or poorly localized one, motivating native event-set evaluation.

For operational triage, an existing alert may prompt preliminary inspection before a problem is reported, while a category-correct, time-aligned prediction can narrow subsequent investigation. We therefore distinguish alert coverage from localization quality. Our evaluation uses a combination of degradation events cleaned by domain experts and customer feedback records.

To fill this gap, we construct the Multi-modal Concurrent Degradation Dataset (MCDD) from production VM observations. Its 22 metric channels and 122-type vocabulary of low-level anomaly events accompany interval annotations for 25 degradation categories; the 122 types are input features derived from production monitoring. In the benchmarks compared here, MCDD differs from unlabeled traces, binary KPI sets, and single-root-cause cases (Lin et al., 2018; Thill et al., 2017; Gao et al., 2020; Weng et al., 2022; Li et al., 2022b; Allam et al., 2024; Phamquiluan, 2024) by representing multiple overlapping, time-bounded degradation events for the same VM (Figure 1e).

However, MCDD's annotation structure calls for models that emit sets of typed intervals rather than only bag-level predictions. Bag-level multiple-instance learning (MIL) aggregates window-level evidence without emitting typed event boundaries; temporal action detection (TAD) can emit multiple bounded events but was developed for a different input domain. The question here is how to combine VM metric sequences with low-level anomaly events for concurrent degradation localization.

To this end, we propose M^3-DD (Multi-Modal Multi-instance Degradation Detector), a framework designed to separate concurrent performance degradations. Building on TAAT-style multi-modal feature extraction (Deng et al., 2024), M^3-DD introduces an Object-Query multi-instance mechanism inspired by DETR-style set prediction (Carion et al., 2020; Zhang, 2026) and TAD (Zhang et al., 2022; Liu et al., 2022; Shi et al., 2023). Each potential degradation event becomes an independent query, and an interval-aware self-attention mechanism separates overlapping feature subspaces over time, enabling joint pattern classification and interval localization. In effect, M^3-DD combines MIL's multi-instance modeling with TAD's boundary regression: it separates multiple time-bounded anomalies rather than searching for a single point.

We evaluate native concurrent intervals with Coverage–Localization Balance (CLB) and average mAP as primary measures. CLB balances event-level alert coverage against type-and-interval localization precision. On MCDD, M^3-DD attains the highest CLB and average mAP among all evaluated systems.

Our contributions:

- We present **MCDD**, a production-derived VM-level dataset pairing metrics and low-level anomaly events with fine-grained *multi-concurrent* interval annotations.
- We propose **M^3-DD**, an Object-Query-based multi-instance detector that combines multi-instance modeling with temporal-boundary regression to identify the categories and recorded intervals of overlapping faults.
- We benchmark concurrent event-set detection on MCDD and show that M^3-DD achieves the highest CLB and average mAP among all evaluated systems, while separately reporting alert coverage and localization precision.

2 Related Work

2.1 Methods for Performance Degradation Detection

Traditional approaches detect metric deviations through anomaly detection. For instance, Gu et al. Gu et al. (2023) model metric correlations and temporal evolution, while LSTMAutoencoders Chouliaras & Sotiriadis (2021) use reconstruction errors for temporal anomaly detection. However, these approaches often simplify the task to binary classification or assume a single fault source, making it difficult to distinguish features in concurrent multi-fault scenarios.

Recent efforts have integrated optimization methods to improve detection efficiency. DeLag Traini & Cortellessa (2023) employs multi-objective optimization to isolate latency-inducing requests, and POBO Guo et al. (2024) combines Bayesian optimization with primal-dual theory for resource tuning. Nevertheless, both rely on simplified metrics and lack the fine-grained information needed to detect complex, multi-dimensional faults in highly concurrent environments.

With the widespread adoption of microservice architectures, automated root cause analysis (RCA) has become important. Systems such as Sage Gan et al. (2021a;b; 2022) use unsupervised learning for cross-service localization, while ClearCausal Tunde-Onadele et al. (2024) and TopoRCA Huang et al. (2024) improve precision through causal analysis and topology awareness. Although BARO Pham et al. (2024) improves robustness to noise, such methods assume single root causes or linear causal chains; recent real-time extensions Li et al. (2022a); Xin et al. (2024) still struggle to resolve concurrent impairments.

2.2 Datasets for Cloud Performance

Current benchmarks rely on synthetic generation Allam et al. (2024) or public production traces. Notable examples include the Azure Public Dataset Lin et al. (2018); Chen et al. (2019); Li et al. (2024) for QoS degradation, Yahoo Webscope S5 Thill et al. (2017) for service KPIs, and cluster traces from Google Gao et al. (2020); Khalil et al. (2016) and Alibaba Weng et al. (2022); Deng et al. (2024); Yong et al. (2025). The AIOPS Challenge Dataset Li et al. (2022b) further integrates cross-layer metrics to support cascading fault analysis.

Despite improvements in scale and diversity, existing datasets have a critical gap in representing concurrent fault patterns. Most assume a single dominant fault per time window or provide only discrete anomaly labels, lacking fine-grained annotations for multiple concurrent degradation events.

2.3 Multiple-Instance and Temporal-Interval Localization

Bag-level multiple-instance learning (MIL) predicts from collections of instances; a bag classifier does not natively return a set of typed time intervals. Temporal action detection (TAD), including ActionFormer Zhang et al. (2022), TadTR Liu et al. (2022), and TriDet Shi et al. (2023), does localize intervals; DETR-style set prediction Carion et al. (2020) also motivates query-based matching of multiple outputs to labels. These methods serve as localization baselines; M^3-DD extends beyond them by jointly modeling concurrent, typed degradation intervals from multi-modal inputs. Our setting introduces a distinct challenge: jointly detecting multiple overlapping degradation categories from heterogeneous VM metric sequences and a low-level anomaly-event stream, which existing methods are not designed to address.

Table 1: MCDD sample and recording counts. The temporal train/test split mirrors production conditions where existing VMs continue operating alongside newly provisioned instances.

Quantity	Train	Test	Total
Recording sequences	1,999	393	2,392
Unique VMs within split	1,924	389	2,303
Window samples	26,839	8,392	35,231
Annotated intervals (recording level)	1,711	1,033	2,744

Each sequence contains 22 metrics; the separate anomaly-event vocabulary has 122 types.

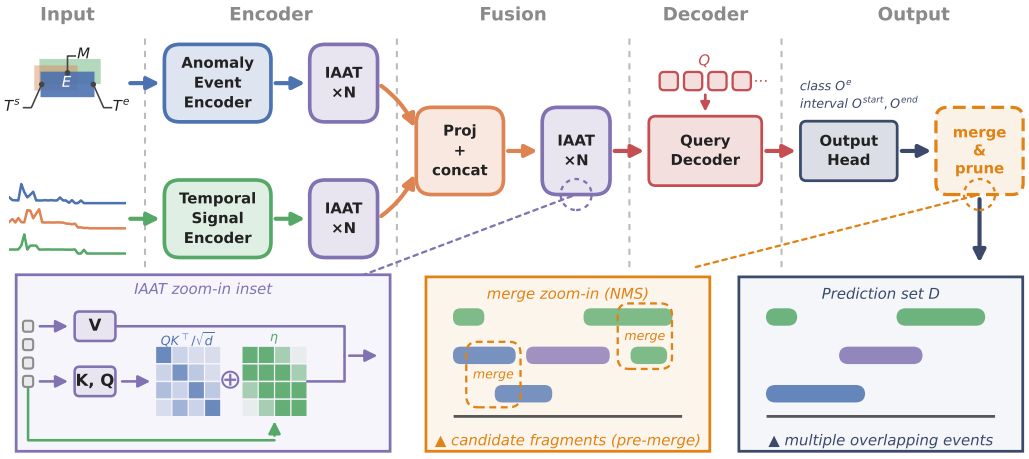

Figure 2: Overview of M^3-DD. Two encoders process event sequence $\boldsymbol{A}$ and signals $\boldsymbol{S}$; their IAAT blocks encode and fuse features using overlap correction $\boldsymbol{\eta}$. An object-query decoder predicts the degradation set $\boldsymbol{D}$.

3 Dataset Construction

To study concurrent performance degradation, we assemble the **Multi-modal Concurrent Degradation Dataset (MCDD)** from production virtual-machine (VM) observations. It contains **22 metric channels**, a vocabulary of **122 low-level anomaly-event types** extracted from monitored signals, and **25 degradation categories** annotated with time intervals. The 122 types are input event categories derived from production monitoring records. Table 1 summarizes the experimental sample organization.

Every degradation label is represented as $(E, T^{\text{start}}, T^{\text{end}})$, where E is one of 25 categories and the bounds define its recorded interval. Multiple labels may overlap. The auxiliary input instead consists of low-level, metric-derived anomaly events $(E, T^{\text{start}}, T^{\text{end}}, M)$, where M is the event magnitude. The input-generation and annotation pipelines are described separately in Appendix A.

The temporal partition, filtering criteria, event extraction, recorded label provenance, and sliding-window organization are outlined in Appendix A. Exact filtering thresholds and full category definitions are provided in Appendix B.1. A comparison with representative benchmark annotation formats appears in Appendix A.5.

4 Methodology

The proposed M^3-DD framework comprises an anomaly-event encoder, a temporal-signal encoder, interval-aware attention (IAAT) for each branch and their fusion, and an object-query decoder (Figure 2). Given low-level metric-derived anomaly events $\boldsymbol{A}$ and multivariate signals $\boldsymbol{S}$ observed up to the reference time, it predicts a set $\boldsymbol{D}$ of degradation categories and intervals over the observed window. Training uses Hungarian matching; inference applies merge-and-prune post-processing.

4.1 ANOMALY EVENT ENCODER

Each element $\boldsymbol{a}_i$ in the anomaly event sequence $\boldsymbol{A}$ consists of four features: the anomaly type E, the start time T^{start}, the end time T^{end}, and the anomaly magnitude M. The anomaly event sequence is defined as:

$$\boldsymbol{A} = \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{L^{\text{A}}}\} = \{(E_i, T_i^{\text{start}}, T_i^{\text{end}}, M_i)\}_{i=1}^{L^{\text{A}}}, \tag{1}$$

where L^{A} denotes the length of the anomaly event sequence.

Inspired by the embedding strategy employed in BERT Devlin et al. (2019), we learn an embedding function $v_E(\cdot)$ that maps each anomaly category index to a real-valued vector of dimension d^{e}. For the anomaly magnitude M, we extract its raw value, sign, and log-absolute value, then map them through a Linear+ReLU network $\mathcal{F}_m$ to output a d^{m}-dimensional magnitude feature.

Given the anomaly type sequence $\boldsymbol{E} = [E_1, \ldots, E_{L^{\text{A}}}] \in \mathbb{Z}^{L^{\text{A}}}$ and the anomaly magnitude sequence $\boldsymbol{M} = [M_1, \ldots, M_{L^{\text{A}}}] \in \mathbb{R}^{L^{\text{A}}}$, the anomaly embedding output $\boldsymbol{Z}^{\text{A},1}$ is:

$$\boldsymbol{Z}^{\text{A},1} = \text{concat}\left(v_E(\boldsymbol{E}),\ \mathcal{F}_m\left(\begin{bmatrix} \boldsymbol{M} \\ \text{sign}(\boldsymbol{M}) \\ \log(|\boldsymbol{M}| + \epsilon) \end{bmatrix}\right)\right) \in \mathbb{R}^{L^{\text{A}} \times d}, \tag{2}$$

where $d = d^{\text{e}} + d^{\text{m}}$ denotes the total feature dimension after concatenation, and ϵ is a small constant for numerical stability. $\boldsymbol{Z}^{\text{A},1}$ serves as the input to the first layer of the IAAT encoder stack.

4.2 TEMPORAL SIGNAL ENCODER

The temporal data are represented as $\boldsymbol{S} \in \mathbb{R}^{C \times L^{\text{S}}}$, comprising C metrics over L^{S} time steps. A convolutional-pooling encoder compresses and transposes this to token-major $\boldsymbol{Z}^{\text{S},1} \in \mathbb{R}^{(L^{\text{S}}/r) \times d}$, where d is the hidden dimension and r is the downsampling rate.

For the IAAT layers' time-correlated attention, we construct interval tuples $\boldsymbol{I}$ delineating the time range covered by each token in $\boldsymbol{Z}^{\text{S},1}$ according to the downsampling rate r. The intervals are defined as:

$$\boldsymbol{I} = \big((T_i^{\text{start}}, T_i^{\text{end}})\big)_{i=1}^{L^{\text{S}}/r} = \big(\big((i-1) \cdot r,\ i \cdot r\big)\big)_{i=1}^{L^{\text{S}}/r}, \tag{3}$$

where these are relative step indices measured from the beginning of the metric sequence, not absolute Unix timestamps. Event timestamps must be converted to the same reference frame before interval-overlap attention is applied. $\boldsymbol{I}$ provides temporal information for the IAAT layers.

4.3 TRANSFORMER AND IAAT LAYER

Standard Transformers struggle with non-uniformly distributed timestamps; TAAT (Deng et al., 2024) addresses this limitation but targets only point events. Our data require modeling relationships between intervals. We therefore propose the Interval-Aware Attention Transformer (IAAT).

4.3.1 MULTI-HEAD TRANSFORMER LAYER

For input $\boldsymbol{Z} \in \mathbb{R}^{L \times d}$, the standard layer (Vaswani et al., 2017) computes $\boldsymbol{Z}^{\text{out}} = \text{MultiHead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V})$ with $\boldsymbol{Q}{=}\boldsymbol{Z}\boldsymbol{W}^Q$, $\boldsymbol{K}{=}\boldsymbol{Z}\boldsymbol{W}^K$, $\boldsymbol{V}{=}\boldsymbol{Z}\boldsymbol{W}^V$, followed by a position-wise FFN with residual connections and layer normalization. However, attention weights derive solely from token semantics, ignoring the non-uniform timestamp distribution relevant to cloud failure prediction—the limitation motivating our IAAT design.

4.3.2 INTERVAL-AWARE ATTENTION TRANSFORMER (IAAT)

The IAAT layer extends the standard Multi-Head Transformer by integrating the Time Relation Estimation (TRE) concept from TAAT and incorporating an estimated temporal correlation matrix into the attention mechanism.

Unlike TAAT, which models individual timestamps, the IAAT explicitly models a sequence of time intervals, where each token corresponds to an interval $I_i = [T_i^{\text{start}}, T_i^{\text{end}}]$. Consistent with the time definition in the Temporal Signal Encoder, we use these timestamps directly. Start and end times

are embedded via a function $v_T(\cdot)$ to form the time embedding sequence $\boldsymbol{Z}^I$. An independent multi-head self-attention layer then computes the time relation matrix $\boldsymbol{R}$:

$$\begin{aligned} \boldsymbol{Z}^I &= \text{concat}\left(v_T(\boldsymbol{T}^{\text{start}}), v_T(\boldsymbol{T}^{\text{end}})\right), \\ \boldsymbol{Q}^I &= \boldsymbol{Z}^I\boldsymbol{W}^{I,Q}, \quad \boldsymbol{K}^I = \boldsymbol{Z}^I\boldsymbol{W}^{I,K}, \\ \boldsymbol{R} &= \text{softmax}\left(\frac{\boldsymbol{Q}^I(\boldsymbol{K}^I)^\top}{\sqrt{d_I}} + \boldsymbol{\eta}\right), \end{aligned} \tag{4}$$

where $\boldsymbol{W}^{I,Q}, \boldsymbol{W}^{I,K} \in \mathbb{R}^{2d_T \times d_I}$ are learnable projection matrices for time features, with $2d_T$ representing the concatenated dimension of start and end time embeddings, and d_I denoting the attention dimension for the temporal branch. Here, $\boldsymbol{\eta} \in \mathbb{R}^{L\times L}$ is a correction matrix derived from the degree of overlap between time intervals, designed to guide the attention network toward more accurate temporal relations. The element $\eta_{i,j}$ is defined as:

$$\eta_{i,j} = \lambda \frac{|I_i \cap I_j|}{|I_i|}, \tag{5}$$

where $|I_i \cap I_j|$ denotes the duration of the overlap between the i-th and j-th intervals, $|I_i|$ is the duration of the i-th interval, and λ is a scaling factor controlling the magnitude of this correction.

Subsequently, the attention score calculation combines the semantic correlation (computed via $\boldsymbol{Q}\boldsymbol{K}^\top$) and the temporal correlation ($\boldsymbol{R}$). Let d' denote the per-head dimension of the semantic query and key. For each attention head, the IAAT formulation is:

$$\text{IAAT}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}, \boldsymbol{R}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d'}} + \boldsymbol{R}\right)\boldsymbol{V}. \tag{6}$$

In this formulation, the term $\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d'}}$ captures semantic co-occurrence patterns of anomaly types, while $\boldsymbol{R}$ injects constraints based on temporal interval overlaps. This unified attention mechanism serves as the foundational building block for both single-modality encoding and subsequent cross-modal fusion.

4.4 Multi-Modal Feature Fusion

To integrate event semantics with temporal dynamics, we introduce a fusion module. The outputs from the event-specific IAAT ($\boldsymbol{Z}^{\text{A,out}}$) and the time-series-specific IAAT ($\boldsymbol{Z}^{\text{S,out}}$) are projected for dimensional consistency and concatenated into $\boldsymbol{Z}^{\text{fuse}}$:

$$\boldsymbol{Z}^{\text{fuse}} = \text{concat}\left(\boldsymbol{Z}^{\text{A,out}}, \boldsymbol{Z}^{\text{S,out}}\right) \in \mathbb{R}^{(L^{\text{A}}+L^{\text{S}}/r)\times d}, \tag{7}$$

where L^{A} and L^{S} represent the sequence lengths of the event and time-series branches, respectively, and r denotes the temporal reduction factor applied to align the resolutions. The resulting fused sequence is then processed by a final *Fusion IAAT* layer. This Fusion IAAT layer (Section 4.3.2) models cross-modal dependencies between anomaly semantics and metric fluctuations in overlapping intervals. The output of this fusion stage, $\boldsymbol{Z}^{\text{final}}$, serves as the memory representation for the subsequent object query decoder.

4.5 Query-based Decoder

To separate concurrent events, we construct an Object-Query decoder. This design models each potential degradation event as an independent query. The framework learns the interactions between these queries and the *fused multi-modal representation* $\boldsymbol{Z}^{\text{final}}$ (generated in Section 4.4) to predict the set of events directly.

First, we initialize a set of L^{Q} learnable vectors, denoted as the query matrix $\mathbf{Q} \in \mathbb{R}^{L^{\text{Q}}\times d}$. Each query retrieves information for one impairment pattern from the fused features. Through a

Multi-Head Cross-Attention mechanism, $\mathbf{Q}$ interacts with $\boldsymbol{Z}^{\text{final}}$ to generate decoded representations $\boldsymbol{Z}^{\text{decode}}$ enriched with semantic context:

$$\boldsymbol{Z}^{\text{decode}} = \text{MultiHead}(\mathbf{Q}, \boldsymbol{Z}^{\text{final}}, \boldsymbol{Z}^{\text{final}}), \tag{8}$$

where $\mathbf{Q}$ serves as the query, while $\boldsymbol{Z}^{\text{final}}$ serves as both the key and the value.

A feed-forward network $\mathcal{F}_o$ (Linear+ReLU) maps $\boldsymbol{Z}^{\text{decode}}$ to category logits and temporal boundaries:

$$\boldsymbol{O} = \begin{bmatrix} \boldsymbol{o}^{\text{e}} \\ \boldsymbol{o}^{\text{start}} \\ \boldsymbol{o}^{\text{end}} \end{bmatrix} = \mathcal{F}_o(\boldsymbol{Z}^{\text{decode}}), \tag{9}$$

where $\boldsymbol{o}^{\text{e}} \in \mathbb{R}^{L^{\text{Q}} \times N_{\text{class}}}$ denotes the logits for impairment categories, and $\boldsymbol{o}^{\text{start}}, \boldsymbol{o}^{\text{end}} \in \mathbb{R}^{L^{\text{Q}}}$ represent the raw predictions for the start and end times, respectively.

Finally, post-processing derives the predictions. The impairment category $\hat{E}$ is determined by applying the $\arg\max$ operation to the category logits, while the predicted start time $\hat{T}^{\text{start}}$ and end time $\hat{T}^{\text{end}}$ are mapped to normalized time coordinates via the sigmoid function $\sigma(\cdot)$:

$$\hat{E} = \arg\max(\boldsymbol{o}^{\text{e}}), \quad \hat{T}^{\text{start}} = \sigma(\boldsymbol{o}^{\text{start}}), \quad \hat{T}^{\text{end}} = \sigma(\boldsymbol{o}^{\text{end}}). \tag{10}$$

Here, $\hat{E}$ indicates the predicted impairment pattern, and the pair $(\hat{T}^{\text{start}}, \hat{T}^{\text{end}})$ defines the normalized impaired time interval. During inference, a merge-and-prune strategy is applied to the predicted event set $\boldsymbol{D} = \{(\hat{E}_i, \hat{T}_i^{\text{start}}, \hat{T}_i^{\text{end}})\}_{i=1}^{L^{\text{Q}}}$ to generate the final consolidated detection results.

4.6 Set Matching and Training Objective

Let $\hat{p}_q(c)$ be the class probability for query q, including background class $\varnothing$, and let $\hat{I}_q = [\hat{s}_q, \hat{e}_q]$ be its normalized interval. For ground-truth event j with class c_j and interval $I_j = [s_j, e_j]$, we assign predictions one-to-one by minimizing the Hungarian cost

$$\mathcal{C}(q, j) = \alpha[-\log \hat{p}_q(c_j)] + \beta \frac{|\hat{s}_q - s_j| + |\hat{e}_q - e_j|}{2} + \gamma[1 - \text{GIoU}(\hat{I}_q, I_j)]. \tag{11}$$

Here GIoU is the generalized intersection-over-union of one-dimensional intervals. Matched queries incur classification cross-entropy, mean absolute start/end error, and $1 - \text{GIoU}$ with positive-loss weights 1:1:1; unmatched queries incur background cross-entropy. If P and U are the matched and unmatched query sets, respectively, the objective used by the implementation is

$$\mathcal{L} = \frac{\sum_{q \in P} \left[-\log \hat{p}_q(c_q) + \frac{|\hat{s}_q - s_q| + |\hat{e}_q - e_q|}{2} + 1 - \text{GIoU}(\hat{I}_q, I_q) \right] + \delta \sum_{q \in U} [-\log \hat{p}_q(\varnothing)]}{|P| + \delta |U|}. \tag{12}$$

The matching weights are $(\alpha, \beta, \gamma) = (1, 0.5, 0.5)$, and the background weight is $\delta = 0.7$ (Appendix B.3). The decoder has ten queries and 26 output classes (25 degradation types plus background).

5 Experiment

5.1 Experimental Setting

We evaluate native concurrent event sets on MCDD (Section 3) under a temporal train/test partition: 1,924 training and 389 test VM identifiers. The partition combines temporal ordering with VM-level diversity: training VMs contribute earlier windows while test windows draw from both held-out dates and previously unseen VM identifiers, mirroring the production setting where existing VMs continue operating and new VMs are provisioned daily. Dataset counts and protocols are in Appendix B.1; implementation settings are in Appendix B.3. Appendix B.2 additionally reports a binary-label comparison that provides a common evaluation ground for conventional detectors.

5.2 Experiments on the Native Multi-Concurrent Scenario

We use MCDD's native labels, allowing multiple degradation types and intervals within one window.

5.2.1 Evaluation Metric

The two primary native-view measures are **Coverage–Localization Balance (CLB)** and **Avg mAP**. CLB is the harmonic mean $2\,\mathrm{LP}\,\mathrm{R}/(\mathrm{LP}+\mathrm{R})$ of localization precision (LP: correct type and tIoU > 0.5) and event-level alert coverage (R: category-agnostic matching without a required interval intersection). Avg mAP measures confidence-ranked localization over tIoU thresholds 0.3–0.7. Secondary metrics DP, CP, LP, and R expose coarse alerts, type identification, localization, and coverage separately. Complete metric definitions are provided in Appendix B.4.

5.2.2 Comparison with Native Set-Output Baselines

We compare native event sets against an adapted DETR-style predictor Carion et al. (2020) and temporal action detectors (TadTR Liu et al. (2022), ActionFormer Zhang et al. (2022), TriDet Shi et al. (2023)). The TAD baselines are trained on MCDD's 25-class interval labels with (T, C) metric arrays (C=22). Table 2 reports the comparison. The within-model ablations in Table 3 further isolate each component's contribution.

Table 2: Native MCDD event-set comparison. Primary measures: CLB and Avg mAP. M^3-DD is designed to leverage heterogeneous data sources including anomaly events when available. Outputs are capped at top-K=50 per recording. The window-level baseline has no learned boundaries (LP/CLB N/A); its mAP uses whole-window pseudo-intervals. Definitions: Appendix B.4.

Family	Method	CLB	Avg mAP	LP	R	DP	CP
Set / TAD	DETR-style (adapted)	0.2389	0.0256	0.1470	0.6370	0.3182	0.2123
	TadTR	0.0486	0.0533	0.0252	0.6805	0.0470	0.0409
	ActionFormer	0.0526	0.0566	0.0272	0.7851	0.0413	0.0349
	TriDet	0.0528	0.0590	0.0273	0.7851	0.0413	0.0350
Multi-label	window-level multi-label	N/A	0.0614	N/A	0.7948	0.6718	0.5295
Ours	**M^3-DD**	**0.4180**	**0.1967**	0.2787	0.8354	0.3712	0.3299

On native event sets, M^3-DD has the highest CLB (0.4180) and Avg mAP (0.1967) among the evaluated systems. Its LP (0.2787) and event-level alert coverage (0.8354) explain the composite score: it balances reports of degradation events with predictions that identify both type and approximate interval. The temporal-action baselines have LP of at most 0.0273, while the adapted DETR-style model attains 0.1470 LP. Section 5.2.3 examines coverage at different concurrency levels.

5.2.3 Concurrency-Degree Analysis

When test windows are grouped by concurrency $(1, 2, 3, 4, \geq 5)$, M^3-DD's recall rises from 0.59 on single-event windows to 0.86 at four concurrent faults and 0.87 at five or more. Its advantage is most pronounced at higher concurrency ($\geq$4 events), where it leads all reported baselines. Its LP is highest in each group. The full breakdown is in Appendix B.5 (Table 8).

5.3 Ablation Study

Table 3 examines the effects of the input modality (time-series only, event-log only, multi-modal), the attention mechanism (IAAT vs. point-event-oriented TAAT), the cross-modal fusion layer (Section 4.4), and the interval-overlap correction $\boldsymbol{\eta}$ (Section 4.3.2).

Table 3: Native-view ablation (top-K=50, up to 80 epochs with early stopping). CLB and Avg mAP are primary. LP/R and DP/CP diagnose localization, alert coverage, and coarser reporting behavior.

Group	Variant	CLB	Avg mAP	LP	R	DP	CP
Modality / Attn.	Time Series Only + IAAT	0.2170	0.0637	0.1258	0.7880	0.2722	0.1977
	Event Log Only + IAAT	0.3663	0.1931	0.2399	0.7742	0.3130	0.2853
	Multi-Modal + TAAT	0.3601	0.1857	0.2303	0.8254	0.3018	0.2773
	Multi-Modal + IAAT (Ours)	**0.4180**	**0.1967**	0.2787	0.8354	0.3712	0.3299
Component	w/o fusion layer	0.3728	0.1868	0.2413	0.8190	0.3207	0.2844
	w/o overlap correction $\boldsymbol{\eta}$	0.3498	0.1849	0.2245	0.7916	0.3027	0.2747

The event-only variant has higher CLB (0.3663) and Avg mAP (0.1931) than time series alone (0.2170 and 0.0637). Combining both modalities raises CLB to 0.4180 while improving Avg mAP to 0.1967, yielding consistent gains over either input alone and validating the complementarity of event and time-series signals. Replacing IAAT with TAAT lowers CLB to 0.3601 and Avg mAP to 0.1857. Removing fusion or the overlap correction likewise reduces both primary measures, with the larger CLB drop under removal of overlap correction (0.3498).

5.4 Cross-Dataset Generalization

To assess adaptability beyond MCDD, we apply M^3-DD and feasible baselines to four public benchmarks—SMD Su et al. (2019), RCAEval Phamquiluan (2024) (RE1-OB and RE2-OB), and Exathlon Jacob et al. (2021)—using their native annotations. We adopt class-agnostic event-level detection F1 at tIoU > 0.3 under one-to-one matching (P/R/F1) as the primary metric, with point-adjustment F1 (AdjF1) as secondary. Checkpoint provenance and modality handling follow each dataset's standard protocol; complete implementation details are provided in Appendix B.5. Table 4 reports results across available model–dataset combinations.

Table 4: Cross-dataset generalization on four public benchmarks under a unified event-level protocol for feasible model–dataset combinations. Primary metric: event-level class-agnostic detection F1 at tIoU > 0.3 (P/R/F1); secondary metric: point-adjustment F1 (AdjF1). RE1/RE2 denote RCAEval RE1-OB / RE2-OB. Bold marks the best value in each column.

Method	SMD				RE1				Exathlon				RE2			
	P	R	F1	AdjF1	P	R	F1	AdjF1	P	R	F1	AdjF1	P	R	F1	AdjF1
TimesNet	0.0573	**1.0000**	0.1084	0.4233	0.0145	0.8000	0.0284	0.9343	0.0459	0.1923	0.0741	0.3523	0.1543	0.9000	0.2634	0.9470
PatchTST	0.0814	0.9649	0.1501	**0.5424**	0.0097	0.7600	0.0191	0.9303	0.0270	0.1923	0.0474	0.3135	0.2188	0.9333	0.3544	0.9499
iTransformer	0.0801	0.9474	0.1477	0.5182	0.0036	0.6400	0.0072	0.8706	0.0159	0.0769	0.0263	0.1187	0.0518	0.9000	0.0980	0.9451
ViLBERT	0.3046	0.3860	0.3405	0.3177	0.8621	**1.0000**	0.9259	0.9883	0.0000	0.0000	0.0000	0.2357	0.8571	**1.0000**	0.9231	0.9798
ActionFormer	0.2192	0.1404	0.1711	0.3258	0.0107	0.6000	0.0211	0.8941	0.2692	0.2692	0.2692	0.3766	0.0316	0.7000	0.0604	0.7850
TriDet	0.0661	0.1491	0.0916	0.5028	0.0350	0.8000	0.0670	0.7975	0.4000	0.1538	0.2222	0.2679	0.9286	0.8667	0.8966	0.8835
TadTR	0.0620	0.3596	0.1058	0.2980	0.9600	0.9600	0.9600	0.6536	**0.8333**	0.3846	**0.5263**	**0.5213**	**1.0000**	**1.0000**	**1.0000**	0.6670
M^3-DD (Ours)	**0.4246**	0.8596	**0.5684**	0.1328	**0.9615**	**1.0000**	**0.9804**	**0.9972**	0.1029	**0.8077**	0.1826	0.1798	**1.0000**	**1.0000**	**1.0000**	**0.9909**

M^3-DD attains the best event-level detection F1 on SMD (0.5684) and RE1 (0.9804), and achieves perfect event F1 on RE2 (1.0000), spanning both metric-window benchmarks and service-level traces. On Exathlon, M^3-DD attains the highest recall (0.8077) among all evaluated methods. These results demonstrate that M^3-DD adapts consistently across heterogeneous monitoring scenarios without task-specific redesign.

6 Conclusion and Limitations

MCDD advances degradation diagnosis beyond binary and window-level benchmarks by preserving multiple overlapping, time-bounded events per VM, and M^3-DD achieves the best CLB and Avg mAP on MCDD, demonstrating effective multi-modal concurrent degradation detection. Ablations confirm that fusing low-level anomaly events with metric sequences is a key driver of this performance, and we propose CLB as a unified operational metric.

CLB jointly captures alert coverage and localization quality, providing an operational view of detection effectiveness. Cross-dataset results demonstrate adaptability to heterogeneous monitoring scenarios.

Several limitations remain. First, although M^3-DD generalizes across datasets, its event-level F1 on Exathlon still trails dedicated temporal action detection methods, indicating that domain adaptation remains necessary when transferring to substantially different monitoring environments. Second, the current evaluation is grounded in MCDD, which reflects a specific production cloud-infrastructure monitoring scenario; broader industrial IoT settings with different metric semantics and degradation patterns have not yet been covered. These observations motivate future work on matched-input ablations that isolate modality contributions under identical baselines, and on evaluation in more diverse temporal generalization settings.

REPRODUCIBILITY STATEMENT

We ensure reproducibility by detailing the model architecture in Section 4, the experimental setup in Section 5, and the hyperparameters in Appendix B.3. To facilitate review and future verification, the complete source code and evaluation scripts are currently available for anonymous access at `https://anonymous.4open.science/r/M3DD-0C96`, and the trained models will be released upon acceptance.

AI USE STATEMENT

During the preparation of this work, the authors utilized generative AI tools to assist with editing the text for clarity and organizing bibliographic data. Following the use of these tools, the authors thoroughly reviewed and edited the content to ensure accuracy. All references were verified against original sources. All figures in this paper were produced with matplotlib; no AI-generated visual content is included. The authors accept full responsibility for the final content of this manuscript.

ETHICS STATEMENT

The MCDD dataset is derived from production cloud-infrastructure monitoring signals. No direct user-identifiable information is present: customer feedback records are used solely as auxiliary signals for data-cleaning decisions (e.g., confirming whether a degradation episode is genuine), and no personal content from those records is retained in the released dataset. Anonymization removes all VM names and specific timestamps prior to release. The dataset will be publicly available upon paper acceptance under an appropriate license.

REFERENCES

Mohamed Allam, Noureddine Boujnah, Noel E O'Connor, and Mingming Liu. Synthetic time series for anomaly detection in cloud microservices. In *International Conference on Machine Learning, Optimization, and Data Science*, pp. 419–433. Springer, 2024.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pp. 213–229. Springer, 2020.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016. doi: 10.1145/2939672.2939785.

Yujun Chen, Xian Yang, Qingwei Lin, Hongyu Zhang, Feng Gao, Zhangwei Xu, Yingnong Dang, Dongmei Zhang, Hang Dong, Yong Xu, et al. Outage prediction and diagnosis for cloud service systems. In *The world wide web conference*, pp. 2659–2665, 2019.

Kukjin Choi, Jihun Yi, Changhwa Park, and Sungroh Yoon. Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines. *IEEE Access*, 9:119447–119465, 2021.

Spyridon Chouliaras and Stelios Sotiriadis. Detecting performance degradation in cloud systems using lstm autoencoders. In *International Conference on Advanced Information Networking and Applications*, pp. 472–481. Springer, 2021.

Lingfei Deng, Yunong Wang, Haoran Wang, Xuhua Ma, Xiaoming Du, Xudong Zheng, and Dongrui Wu. Time-aware attention-based transformer (taat) for cloud computing system failure prediction. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4906–4917, 2024.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long*

and Short Papers), pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

Yu Gan, Mingyu Liang, Sundar Dev, David Lo, and Christina Delimitrou. Sage: practical and scalable ml-driven performance debugging in microservices. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 135–151, 2021a.

Yu Gan, Mingyu Liang, Sundar Dev, David Lo, and Christina Delimitrou. Sage: Leveraging ml to diagnose unpredictable performance in cloud microservices. *arXiv preprint arXiv:2112.06263*, 2021b.

Yu Gan, Mingyu Liang, Sundar Dev, David Lo, and Christina Delimitrou. Enabling practical cloud performance debugging with unsupervised learning. *ACM SIGOPS Operating Systems Review*, 56(1):34–41, 2022.

Jiechao Gao, Haoyu Wang, and Haiying Shen. Task failure prediction in cloud data centers using deep learning. *IEEE transactions on services computing*, 15(3):1411–1422, 2020.

Wenwei Gu, Jinyang Liu, Zhuangbin Chen, Jianping Zhang, Yuxin Su, Jiazhen Gu, Cong Feng, Zengyin Yang, and Michael Lyu. Performance issue identification in cloud systems with relational-temporal anomaly detection. *arXiv preprint arXiv:2307.10869*, 2023.

Hengquan Guo, Hongchen Cao, Jingzhu He, Xin Liu, and Yuanming Shi. Pobo: safe and optimal resource management for cloud microservices. *ACM SIGMETRICS Performance Evaluation Review*, 51(4):20–21, 2024.

HelenGuohx. Logbert: Log anomaly detection via bert. GitHub: https://github.com/HelenGuohx/logbert, 2020.

Chengqiang Huang, Geyong Min, Yulei Wu, Yiming Ying, Ke Pei, and Zuochang Xiang. Time series anomaly detection for trustworthy services in cloud computing systems. *IEEE Transactions on Big Data*, 2017.

Youliang Huang, Jue Zhang, Xiaolin Chai, and Yan Sun. Toporca: A lightweight root cause analysis system based on application topology. In *2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 2943–2948. IEEE, 2024.

Vincent Jacob, Feiyang Song, Arnaud Stiegler, Bijan Rad, Yanlei Diao, et al. Exathlon: A benchmark for explainable anomaly detection over time series. *Proceedings of the VLDB Endowment*, 14(11), 2021.

Moataz H Khalil, Walaa M Sheta, and Adel S Elmaghraby. Categorizing hardware failure in large scale cloud computing environment. In *2016 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pp. 327–332. IEEE, 2016.

Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. *arXiv preprint arXiv:2102.03334*, 2021.

Haozhe Li, Minghua Ma, Yudong Liu, Pu Zhao, Lingling Zheng, Ze Li, Yingnong Dang, Murali Chintalapati, Saravan Rajmohan, Qingwei Lin, et al. Why does prediction accuracy decrease over time? uncertain positive learning for cloud failure prediction. *arXiv preprint arXiv:2402.00034*, 2024.

Richard Li, Min Du, Zheng Wang, Hyunseok Chang, Sarit Mukherjee, and Eric Eide. Longtale: Toward automatic performance anomaly explanation in microservices. In *Proceedings of the 2022 ACM/SPEC on International Conference on Performance Engineering*, pp. 5–16, 2022a.

Zeyan Li, Nengwen Zhao, Shenglin Zhang, Yongqian Sun, Pengfei Chen, Xidao Wen, Minghua Ma, and Dan Pei. Constructing large-scale real-world benchmark datasets for aiops. *arXiv preprint arXiv:2208.03938*, 2022b.

Qingwei Lin, Ken Hsieh, Yingnong Dang, Hongyu Zhang, Kaixin Sui, Yong Xu, Jian-Guang Lou, Chenggang Li, Youjiang Wu, Randolph Yao, et al. Predicting node failure in cloud service systems. In *Proceedings of the 2018 26th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering*, pp. 480–490, 2018.

Xiaolong Liu, Qimeng Wang, Yao Hu, Xu Tang, Shiwei Zhang, Song Bai, and Xiang Bai. End-to-end temporal action detection with transformer. *IEEE Transactions on Image Processing*, 31: 6277–6291, 2022.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations (ICLR)*, 2024.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*, 2019.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations (ICLR)*, 2023.

Luan Pham, Huong Ha, and Hongyu Zhang. Baro: Robust root cause analysis for microservices via multivariate bayesian online change point detection. *Proceedings of the ACM on Software Engineering*, 1(FSE):2214–2237, 2024.

Phamquiluan. Rcaeval: A benchmark for root cause analysis. GitHub: `https://github.com/phamquiluan/RCAEval`, 2024.

Lucía Pons, Josué Feliu, Julio Sahuquillo, María E Gómez, Salvador Petit, Julio Pons, and Chaoyi Huang. Cloud white: Detecting and estimating qos degradation of latency-critical workloads in the public cloud. *Future Generation Computer Systems*, 138:13–25, 2023.

Dingfeng Shi, Yujie Zhong, Qiong Cao, Lin Ma, Jia Li, and Dacheng Tao. Tridet: Temporal action detection with relative boundary modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Jacopo Soldani and Antonio Brogi. Anomaly detection and failure root cause analysis in (micro) service-based cloud applications: A survey. *ACM Computing Surveys (CSUR)*, 55(3):1–39, 2022.

Ya Su, Youjian Zhao, Chenyang Niu, Rong Liu, Dan Pei, and Yi Liu. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

Markus Thill, Wolfgang Konen, and Thomas Bäck. Online anomaly detection on the webscope s5 dataset: A comparative study. In *2017 Evolving and Adaptive Intelligent Systems (EAIS)*, pp. 1–8. IEEE, 2017.

Luca Traini and Vittorio Cortellessa. Delag: Using multi-objective optimization to enhance the detection of latency degradation patterns in service-based systems. *IEEE Transactions on Software Engineering*, 49(6):3554–3580, 2023.

Olufogorehan Tunde-Onadele, Feiran Qin, Xiaohui Gu, and Yuhang Lin. Clearcausal: Cross layer causal analysis for automatic microservice performance debugging. In *2024 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)*, pp. 175–180. IEEE, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Zexin Wang, Changhua Pei, Yang Liu, Hengyue Jiang, Quan Zhou, Haotian Si, Hang Cui, Jianhui Li, Gaogang Xie, Jingjing Li, and Dan Pei. Vits: Teaching machines to see time series anomalies like human experts. *arXiv preprint arXiv:2510.04710*, 2025.

Qizhen Weng, Wencong Xiao, Yinghao Yu, Wei Wang, Cheng Wang, Jian He, Yong Li, Liping Zhang, Wei Lin, and Yu Ding. MLaaS in the wild: Workload analysis and scheduling in Large-Scale heterogeneous GPU clusters. In *19th USENIX symposium on networked systems design and implementation (NSDI 22)*, pp. 945–960, 2022.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *International Conference on Learning Representations (ICLR)*, 2023.

Ruyue Xin, Peng Chen, Paola Grosso, and Zhiming Zhao. A fine-grained robust performance diagnosis framework for run-time cloud applications. *Future Generation Computer Systems*, 155: 300–311, 2024.

Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang 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*, pp. 187–196, 2018.

Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly transformer: Time series anomaly detection with association discrepancy. In *International Conference on Learning Representations (ICLR)*, 2022.

Ke Xu, Yun Wang, Leni Yang, Yifang Wang, Bo Qiao, Si Qin, Yong Xu, Haidong Zhang, and Huamin Qu. Clouddet: Interactive visual analysis of anomalous performances in cloud computing systems. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):1107–1121, 2020.

Yaoguang Yong, Xiaoming Du, Xuhua Ma, Yuxiang Wang, Bin Yao, Xudong Zheng, and Huite Yi. Predicting dram-caused risky vms in large-scale clouds. In *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 1231–1245. IEEE, 2025.

Chen-Lin Zhang, Jianxin Wu, and Yin Li. Actionformer: Localizing moments of action with transformers. In *European Conference on Computer Vision (ECCV)*, 2022.

Haokun Zhang. Long-horizon set prediction for multi-object tracking via temporal instance queries. In *IEEE International Conference*. IEEE, 2026.

Shenglin Zhang, Sibo Xia, Wenzhao Fan, Binpeng Shi, Xiao Xiong, Zhenyu Zhong, Minghua Ma, Yongqian Sun, and Dan Pei. Failure diagnosis in microservice systems: A comprehensive survey and analysis. *ACM Transactions on Software Engineering and Methodology*, 2025.

Zhenyu Zhong, Ruowei Fu, Minghua Ma, Shenglin Zhang, Yongqian Sun, Chetan Bansal, and Dan Pei. Llm-enhanced failure localization in microservices: Integrating multi-modal data and expert interpretation. *IEEE Transactions on Services Computing*, 19(3):2275–2288, 2026.

A Dataset Construction Details

This appendix complements Section 3 with the full MCDD construction pipeline.

A.1 Time Series Sampling

The virtual machine (VM) is the observational unit for metrics collected in production. We sample windows associated with observed operations and metric changes, including VM live migration, instance startup/shutdown, and CPU spikes; these are production sampling contexts. The sequences contain 22 KPI channels, including CPU utilization, memory usage, and network traffic.

To ensure data integrity and model robustness, we use three criteria to exclude low-quality segments. First, we enforce a **Non-Null Rate** threshold to ensure temporal completeness; specifically, any segment in which the proportion of missing (NULL) values exceeds a predefined limit is discarded to mitigate the impact of device failures or transmission interruptions. Second, we apply a **Coverage** constraint to account for the distribution of missing data. Even if a segment satisfies the overall Non-Null Rate, it is rejected if any individual metric channel consists entirely of NULL values. This distinction tolerates sparse missing values while excluding systematic data loss across specific channels to prevent bias. Finally, we use an **Information Rate** metric to filter out stationary signals. By calculating the variance for each metric and assessing the proportion of metrics exceeding a variance threshold, we exclude samples dominated by nearly constant signals. Such segments provide little information about anomalies or degradation patterns and would otherwise introduce noise into feature learning.

A.2 Anomaly Event Generation

The auxiliary event vocabulary contains 122 low-level metric–detector types (including changepoint, outlier, high, and drop0 detections), not degradation labels. At inference time, the model requests the preceding 30 minutes of events from an internal anomaly-event retrieval API; its input comprises event categories; unrecognized types are filtered during preprocessing. Both input and target cover the observed period, framing the task as retrospective interval detection.

The anomaly-event tuples $(E, T^{\text{start}}, T^{\text{end}}, M)$ contain an event type, start and end timestamps, and a magnitude. Their timestamps align them with the metric sequences; the label tuples have a separate category vocabulary.

A.3 Degradation Event Annotation

The dataset construction code consumes degradation categories and absolute start/end timestamps from aggregated production diagnostic records. Upstream labels are derived from production diagnostic workflows, combining rule-based diagnoses, customer feedback, and expert corrections. Expert SRE teams performed thorough manual cleaning to ensure high annotation quality. The reported bounds reflect recorded production timestamps from operational monitoring systems.

Unlike binary annotation schemes, MCDD accommodates concurrent impairments. Degradation events are recorded as tuples $(E, T^{\text{start}}, T^{\text{end}})$, with a category and recorded start/end timestamps. These tuples preserve the natural overlap structure of recorded production intervals.

A.4 Sample Organization

The experiments use standard windows constructed from separate VM metric sequences, anomaly-event records, and degradation-label records. The dataset will be publicly released with anonymization upon paper acceptance.

For each reference time t, the metric input comprises 100 one-minute samples ending at t. Anomaly events and degradation labels both refer to the preceding 30-minute interval $[t - 30, t]$, with no events dated after t in the examined sample file. The former is an inference-time input and the latter a training/evaluation target; the task operates within a retrospective detection framing. The standard sample set contains 35,231 windows from 2,392 recordings, with 2,744 recorded degradation intervals at the recording level. The metric sampling cadence is 60 seconds.

A.5 Comparison with Existing Public Benchmarks

Table 5 compares MCDD with representative public cloud and microservice benchmarks along dimensions relevant to concurrent-degradation research: observational unit, modality, annotation granularity (point vs. interval), label type, native support for concurrent faults, and whether the data are production-derived. No existing benchmark in the comparison provides both concurrent fault labels and precise temporal intervals; this distinction motivates the label-degradation protocols used for comparison.

Table 5: Comparison of MCDD with representative public benchmarks. "Concurrent" indicates native support for multiple overlapping, time-bounded fault labels.

Benchmark	Unit	Modality	Granularity	Label type	Concurrent	Production
Azure / Google / Alibaba traces	VM/job	metric	–	none	×	✓
Yahoo S5 / SMD / SMAP-MSL / NAB	entity	metric	point	binary	×	mixed
AIOps / RCAEval / Nezha	service	metric+log+trace	point/case	multi-class	×	✓
AnoMod / fault-propagation bench	service	multi-modal	case	multi-class	×	✓
MCDD (Ours)	**VM**	**multi-modal**	**interval**	**multi-label**	✓	✓

B Experimental Setting and Evaluation Metrics

B.1 Dataset Partitioning and Binary Label Degradation

The construction code partitions recordings strictly by date (before January 25, 2026 for training), ensuring zero temporal overlap between splits. The training set contains 1,924 distinct VM identifiers and the test set 389 (2,303 overall). Ten VMs appear in both periods by design: this evaluates the model's ability to generalize across different degradation episodes on the same infrastructure, mirroring production deployment where systems evolve over time. The evaluation uses a production-aligned temporal split:

- **Training Set**: 1,999 recordings from 1,924 VM identifiers, totaling 26,839 window samples before the 5% validation holdout.
- **Test Set**: 393 recordings from 389 VM identifiers, totaling 8,392 window samples from later dates.

MCDD's "multi-concurrent" annotation allows several degradation types in the same window. Binary label degradation accommodates detectors without native interval-set outputs. M^3-DD thresholds are selected on the training-derived validation holdout.

For comparisons with traditional binary anomaly detectors, we apply **Binary Degradation**: after merging all impairment events along the timeline, a time step t is labeled 1 (anomaly) if any impairment is present, and 0 (normal) otherwise. This discards fault categories and concurrency counts; native multi-concurrent labels are retained for the set-output evaluation.

B.2 Binary-Label Compatibility Comparison

This supplementary check tests whether MCDD's labels can also support conventional binary comparisons; this comparison focuses on binary degradation detection under a simplified label setting. Table 6 summarizes the results. TimesNet Wu et al. (2023), PatchTST Nie et al. (2023), and iTransformer Liu et al. (2024) are trained on normal training recordings using reconstruction residuals. Event-only baselines are TAAT Deng et al. (2024), LogBERT HelenGuohx (2020), and XGBoost Chen & Guestrin (2016); multimodal baselines include ViLT Kim et al. (2021) (metric-window heatmaps and event-ID text) and ViLBERT Lu et al. (2019) (projected metric features and event tokens). The adapted classifiers use binary window labels in MCDD. Several baselines require instantaneous events, for which an N-minute event is decomposed into N point events. For M^3-DD, query class probabilities yield a foreground–background margin; after thresholding, predicted intervals are projected to the time axis, overlapping windows are combined by union, and point adjustment is applied.

Table 6: Reported binary results on MCDD, where any concurrent impairment yields a positive label. Bold highlights the best result per column.

Modality	Method	P	R	F1
Time Series	TimesNet	39.32	84.73	53.72
	PatchTST	39.07	84.10	53.35
	iTransformer	39.45	85.32	53.95
Event	LogBERT	59.51	54.60	56.95
	TAAT	75.38	75.28	75.33
	XGBoost	78.87	**93.95**	85.75
Multi-Modal	ViLT†	57.85	98.05	72.28
	ViLBERT	78.86	72.22	75.39
	M^3-DD	**84.94**	92.06	**88.35**

†All methods are evaluated under the same protocol on the MCDD test split.

Table 7: Implementation details of M^3-DD and the training environment.

Item	Value
Optimizer	AdamW (β_1=0.9, β_2=0.999, ϵ=10^{-8}, weight decay 0.01)
Learning rate / schedule	5×10^{-5}, constant (no scheduler)
Batch size	256
Training epochs	At most 80; early stopping after at least 5 consecutive epochs without improvement
Hidden dim d / heads H	256 / 4 (main Transformer); 2 heads per sub-model
#Object queries L^{Q}	10
Downsampling rate r	4 (input 100 steps $\rightarrow$ 2$\times$MaxPool1d(2) $\rightarrow$ 25 tokens)
λ (overlap scaling, Sec. 4.3.2)	1.0
Loss weights	Hungarian cost: class α=1.0 / start-end β=0.5 / IoU γ=0.5; positive-sample loss 1:1:1; background δ=0.7
Hardware (GPU / memory)	NVIDIA H20, 96 GB
Framework	PyTorch 2.5.1+cu124, Python 3.9.23

B.3 Implementation Details

With L^{Q}=10 queries, a single prediction window can return at most ten impairment events.

B.4 Concurrency-Aware Evaluation Metrics

Because M^3-DD detects a set of q events, one-to-one matching is needed before computing event-level metrics. We use the **Hungarian matching algorithm** with a temporal intersection-over-union (tIoU) cost and two matching modes. Detection-level counts (TD, hence DP and R) use *category-agnostic* matching (cost $= 1 -$ tIoU), while classification- and localization-level counts (CTP and LTP) use separate *category-constrained* matching. Only LTP requires temporal overlap above a threshold: a TD or CTP match indicates that a degradation issue was reported within the evaluation context. All evaluated methods are limited to the top-K=50 highest-confidence proposals per recording under the same protocol.

From these matches, we count three types of true positives at different diagnostic resolutions:

- **True Detection (TD)**: under category-agnostic matching, a pair is counted if both the prediction and ground truth represent a non-background degradation ($o^e \in \mathcal{C}_{deg}$ and $l^e \in \mathcal{C}_{deg}$). This records a reported issue without requiring the correct type or interval.
- **Classification True Positive (CTP)**: under separate category-constrained matching, a matched pair has the same degradation category ($o^e == l^e$) but need not have overlapping intervals.
- **Localization True Positive (LTP)**: a category-constrained matched pair that additionally satisfies a temporal-overlap threshold (tIoU$(o^e, l^e) > 0.5$).

Formally, let $\mathcal{M}_{\text{ag}}$ be the set of category-agnostic matched pairs and $\mathcal{M}_{\text{cc}}$ the set of category-constrained (per-class) matched pairs. We define the metrics as follows:

$$\begin{aligned} \text{TD} &= \sum_{(o^e,l^e)\in\mathcal{M}_{\text{ag}}} \mathbb{I}(o^e \in \mathcal{C}_{deg} \wedge l^e \in \mathcal{C}_{deg}) \\ \text{CTP} &= \sum_{(o^e,l^e)\in\mathcal{M}_{\text{cc}}} \mathbb{I}(o^e == l^e) = |\mathcal{M}_{\text{cc}}| \\ \text{LTP} &= \sum_{(o^e,l^e)\in\mathcal{M}_{\text{cc}}} \mathbb{I}(\text{tIoU}(o^e, l^e) > 0.5) \end{aligned} \tag{13}$$

where $\mathbb{I}(\cdot)$ is the indicator function, $\mathcal{C}_{deg}$ denotes the set of all degradation categories, and tIoU is calculated as $\frac{|I_{pred} \cap I_{label}|}{|I_{pred} \cup I_{label}|}$.

Based on these definitions, we propose four evaluation metrics: **Detection Precision (DP)**, **Classification Precision (CP)**, **Localization Precision (LP)**, and **Recall (R)**:

$$\text{DP} = \frac{\text{TD}}{N_{\text{pred}}}, \quad \text{CP} = \frac{\text{CTP}}{N_{\text{pred}}}, \quad \text{LP} = \frac{\text{LTP}}{N_{\text{pred}}}, \quad \text{R} = \frac{\text{TD}}{N_{\text{gt}}} \tag{14}$$

where $N_{\text{pred}} = \sum \mathbb{I}(o^e \in \mathcal{C}_{deg})$ is the total number of predicted degradation events, and $N_{\text{gt}} = \sum \mathbb{I}(l^e \in \mathcal{C}_{deg})$ is the total number of ground truth events.

Interpretation:

- **DP (Detection Precision)** indicates how often an alarm corresponds to an actual degradation in the evaluation context. An analyst still needs to determine its type and precise time range.
- **R (Event-level Alert Coverage)** indicates what fraction of annotated degradation events are paired with an alert in the evaluation recording, regardless of predicted type or interval intersection. The assignment does not require the alert to fall inside the true event interval. R measures event-level alert coverage across all degradation categories.
- **CP (Classification Precision)** indicates how often a prediction also identifies the degradation type, though an analyst must still locate the interval. CP uses a separate matching assignment from DP.
- **LP (Localization Precision)** additionally requires sufficient interval overlap (tIoU > 0.5) and is the principal measure of the intended output: the correct type at the correct time.

Coverage–Localization Balance (CLB). We summarize the two operational objectives represented by event-level alert coverage R and localization precision LP with an equal-weight harmonic mean:

$$\text{CLB} = \frac{2\,\text{LP}\,\text{R}}{\text{LP} + \text{R}}, \tag{15}$$

with CLB set to zero if both inputs are zero and left undefined if LP is not applicable. R and LP use different matching assignments and do not share a true-positive definition: CLB adapts the harmonic-mean principle to concurrent multi-type detection, balancing alert coverage against localization precision. It rewards systems that both issue alerts for annotated degradation events and provide category-correct, sufficiently aligned predictions.

Average mAP (Avg mAP). DP/CP/LP are computed as ratios over the number of proposals, rewarding methods that balance precision with coverage. As a complementary primary, confidence-ranked measure, we also report mean average precision on the same top-K proposal sets. For a tIoU threshold α, all predictions are ranked by confidence and matched in descending order to unmatched ground-truth events of the same category with the highest tIoU; a prediction with tIoU $\geq \alpha$ is a true positive and otherwise a false positive. The per-class average precision is the area under its precision–recall curve using 101-point interpolation,

$$\text{AP}_c@\alpha = \int_0^1 P_c(r)\,dr, \qquad \text{mAP}@\alpha = \frac{1}{|\mathcal{C}_{deg}|} \sum_{c\in\mathcal{C}_{deg}} \text{AP}_c@\alpha, \tag{16}$$

and the reported **Avg mAP** averages mAP@α over $\alpha \in \{0.3, 0.4, 0.5, 0.6, 0.7\}$.

B.5 Extended Results: Cross-Dataset Generalization and Concurrency Breakdown

This appendix extends Section 5.4 and reports the concurrency breakdown from Section 5.2.3.

Cross-dataset generalization. We evaluate M^3-DD and feasible baselines on four public benchmarks—SMD Su et al. (2019), RCAEval Phamquiluan (2024) (RE1-OB and RE2-OB), and Exathlon Jacob et al. (2021)—using their native annotations. The primary metric is class-agnostic event-level detection F1 at tIoU > 0.3 under one-to-one Hungarian matching (P/R/F1); point-adjustment F1 (AdjF1) serves as the secondary metric. Checkpoint provenance and modality handling follow each dataset's standard protocol: time-series baselines (TimesNet, PatchTST, iTransformer) use reconstruction residuals with dataset-recommended thresholds; ViLBERT uses projected metric features and event tokens; temporal action detectors (ActionFormer, TriDet, TadTR) are trained on interval annotations where available. Results are presented in Table 4 in the main text (Section 5.4).

Concurrency-degree breakdown. Table 8 groups test windows by concurrent event count $(1, 2, 3, 4, \geq 5$; $37/68/94/85/45$ windows and $37/136/282/340/238$ ground-truth events) under the top-K=50 native protocol of Table 2. M^3-DD's recall rises from 0.59 to 0.87, leading the baselines at four or more events; its localization precision leads in every group. The window-level multi-label baseline leads in DP and CP but has no interval-regression output (LP: N/A).

Table 8: Native View performance by number of co-occurring events per window (top-K=50). Bold: best per column; N/A: no interval-regression output.

Method	Detection Precision (DP)					Classification Precision (CP)				
	1	2	3	4	≥ 5	1	2	3	4	≥ 5
ActionFormer	0.0505	0.0772	0.1341	0.1973	0.2475	0.0369	0.0575	0.1095	0.1731	0.2264
TriDet	0.0484	0.0764	0.1230	0.1964	0.2219	0.0363	0.0562	0.0987	0.1784	0.1908
TadTR	0.0876	0.1450	0.2504	0.3445	0.4469	0.0637	0.1088	0.2012	0.3065	0.4196
DETR-style (adapted)	0.0983	0.1736	0.2738	0.3672	0.5066	0.0809	0.0882	0.1820	0.2883	0.3783
window-level multi-label	**0.2923**	**0.5257**	**0.6474**	**0.8339**	**0.9140**	**0.1846**	**0.3371**	**0.4954**	**0.7076**	**0.7151**
M^3-DD (Ours)	0.1507	0.2725	0.3835	0.4460	0.4836	0.1370	0.2290	0.3292	0.4003	0.4416

Method	Localization Precision (LP)					Recall (R)				
	1	2	3	4	≥ 5	1	2	3	4	≥ 5
ActionFormer	0.0252	0.0369	0.0806	0.1504	0.1941	0.7027	0.6912	0.7731	0.7412	0.7395
TriDet	0.0190	0.0398	0.0717	0.1481	0.1721	**0.7568**	**0.7206**	**0.8085**	0.7059	0.7479
TadTR	0.0239	0.0611	0.1187	0.2079	0.2970	0.5946	0.5588	0.6135	0.6676	0.6891
DETR-style (adapted)	0.0405	0.0413	0.1336	0.2109	0.2862	0.4595	0.4632	0.5816	0.7118	0.6471
window-level multi-label	N/A	N/A	N/A	N/A	N/A	0.5135	0.6765	0.7553	0.7382	0.7143
M^3-DD (Ours)	**0.1027**	**0.1710**	**0.2609**	**0.3592**	**0.3972**	0.5946	0.6912	0.7766	**0.8618**	**0.8697**