Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

CovDiff-PaAno: Covariance-Aware Differential Patch Scoring with a Shared Encoder for Time-Series Anomaly Detection

Guan Yang
Zhongyuan University of Technology
Weisen Zong
Zhongyuan University of Technology
Boyuan Zhu
zby@zut.edu.cn

Zhongyuan University of Technology
Yumeng Shi
Zhongyuan University of Technology
Yongshang Li
Zhongyuan University of Technology
ChunLei Li
Zhongyuan University of Technology
Boyang Qu
Zhongyuan University of Technology
Bo Chen
Shenzhen University

Research Article

Keywords: Time-series anomaly detection, Patch-based representation learning, Mahalanobis distance, Differential scoring, Multi-view anomaly scoring, Score calibration

Posted Date: November 4th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11316661/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

CovDiff-PaAno: Covariance-Aware Differential Patch Scoring with a Shared Encoder for Time-Series Anomaly Detection

Abstract

Patch-based time-series anomaly detectors often score raw-patch embeddings by angular distance to normal memories. This assumes that angular similarity sufficiently captures abnormality, although it can underrepresent deviations along low-variance embedding directions and temporal changes in slope or curvature. We propose CovDiff-PaAno, a scoring-stage extension of PaAno that combines regularized Mahalanobis memory scoring with raw, first-order-difference, and second-order-difference views. A single frozen encoder processes all views, while each view maintains its own memory bank and covariance estimate. Fixed score-level fusion combines their point-level anomaly evidence without adding trainable backbone parameters. We evaluated the method across ten seeds on the multivariate collection of the Time-Series Benchmark for Anomaly Detection (TSB-AD-M). CovDiff-PaAno increased the area under the receiver operating characteristic curve (AUC-ROC) from 0.7612 to 0.7843 and the range-based F1 score (Range-F1) from 0.4043 to 0.4384. On the univariate collection (TSB-AD-U), the corresponding scores increased from 0.8532 to 0.8626 and from 0.4822 to 0.4924. A three-seed encoder comparison showed that view-dependent encoders improved every metric by less than 0.005, supporting the shared encoder as a parameter-efficient compromise. Conventional label-free score normalization produced only marginal changes, although empirical cumulative distribution function normalization substantially degraded performance. The method retained PaAno's encoder parameter count, but required 3.47 times the inference time and 10.01 times the peak allocated inference memory. These findings show that inference-stage scoring geometry and temporal views can strengthen patch-based detection without enlarging the backbone, while introducing a measurable deployment trade-off.

Keywords: Time-series anomaly detection, Patch-based representation learning, Mahalanobis distance, Differential scoring, Multi-view anomaly scoring, Score calibration

1 Introduction

Time-series anomaly detection (TSAD) identifies observations or intervals that depart from normal temporal behavior. It supports industrial monitoring, cyber-physical systems, DevOps, and sensor analytics, where delayed detection can obstruct diagnosis and intervention (Blázquez-García et al. 2021; Chandola et al. 2009; Farag et al. 2022; Fawzy et al. 2023; Saheed et al. 2023). Contemporary detectors model normal behavior through reconstruction, forecasting, density estimation, or learned representations (Ma et al. 2025; Malhotra et al. 2015; Sakurada and Yairi 2014; Su et al. 2019; Tuli et al. 2022; Xu et al. 2022). However, large-scale evaluations show that architectural complexity alone does not ensure reliable performance under consistent protocols (Liu and Paparrizos 2024; Paparrizos et al. 2022; Sarfraz et al. 2024; Schmidl et al. 2022). The rule that converts a representation into an anomaly score therefore remains central to the detection problem.

Patch-based detectors encode overlapping subsequences and compare query embeddings with normal references. PaAno follows this paradigm using a lightweight convolutional encoder, a normal patch memory, and cosine nearest-neighbor matching (Park and Kang 2026). This scoring rule assumes that raw-view angular distance captures the deviations relevant to anomaly detection. Yet anomaly evidence depends jointly on the learned representation, the geometry of the distance function, and the temporal view presented to the scorer.

Two mechanisms motivate a broader scoring formulation. First, normal patch embeddings may form correlated and anisotropic distributions. Cosine distance ignores scale and covariance, whereas Mahalanobis distance weights deviations according to variability in the normal reference distribution (Mahalanobis 1936; Defard et al. 2021). Second, some anomalies preserve plausible values while altering local slope, acceleration, or turning behavior. Temporal differencing exposes these changes, consistent with evidence that transformed representations can reveal patterns obscured in raw time series (Jastrzebska 2022). Patch-memory scoring can therefore be both distribution-dependent and view-dependent.

Multi-view scoring also introduces risks. Differential patches may not match an encoder trained on raw patches, branch scores may have incompatible scales, and repeated scoring increases inference cost. The central question is whether one trained patch encoder can support useful view-specific deviation models without the gains depending on fragile calibration or disproportionate resource consumption.

We address this question with CovDiff-PaAno, a scoring-stage extension of PaAno. It replaces cosine matching with regularized Mahalanobis memory scoring and evaluates raw, first-order-difference, and second-order-difference views. One frozen encoder is shared across the three views, while each view maintains its own memory bank and covariance estimate. Fixed score-level fusion combines the resulting point-level evidence without adding trainable backbone parameters.

The evaluation separates detection gains from the assumptions introduced by this design. Ten-seed experiments compare CovDiff-PaAno with the reproduced PaAno baseline on TSB-AD-M and TSB-AD-U. Additional controls examine shared and view-adapted encoders, label-free score calibration, fusion-weight sensitivity, and

computational cost. The results support covariance-aware differential scoring while showing that parameter efficiency does not imply inference efficiency.

The main contributions are as follows:

- We identify a scoring bottleneck in patch-memory detection: raw-view angular matching can underrepresent covariance-structured deviations and anomalies expressed through temporal change.
- We develop CovDiff-PaAno, which combines regularized Mahalanobis matching with raw and differential evidence using view-specific memories and one shared frozen encoder.
- We evaluate the method under repeated benchmark runs and targeted controls that distinguish detection effectiveness, encoder-sharing validity, fusion robustness, and inference cost.

2 Related Work

2.1 Time-Series Anomaly Detection and Evaluation

Time-series anomaly detection (TSAD) identifies observations or intervals that deviate from expected temporal behavior. Classical anomaly detectors characterize abnormality through local density, recursive isolation, or a learned boundary around normal samples (Breunig et al. 2000; Liu et al. 2008; Tax and Duin 2004). These methods remain important because they expose the geometric assumptions underlying an anomaly score. Recent surveys classify time-series and general anomaly detectors according to their representations, objectives, and assumptions rather than architectural depth alone (Choi et al. 2021; Pang et al. 2021; Ruff et al. 2021; Zamanzadeh Darban et al. 2024).

Deep TSAD methods commonly estimate abnormality through reconstruction, forecasting, density estimation, or temporal association. Representative examples include autoencoders (Sakurada and Yairi 2014), recurrent predictors (Hundman et al. 2018; Malhotra et al. 2015), latent mixture models (Zong et al. 2018), adversarial reconstruction (Audibert et al. 2020), and Transformer-based detectors (Tuli et al. 2022; Xu et al. 2022). For multivariate sequences, MSCRED models inter-sensor signature matrices (Zhang et al. 2019), whereas MTAD-GAT and GDN learn graph-structured dependencies among variables (Deng and Hooi 2021; Zhao et al. 2020). These approaches mainly improve the learned representation or prediction mechanism.

Architecture-level gains, however, do not guarantee reliable detection under controlled evaluation. Large-scale studies report substantial sensitivity to datasets, metrics, thresholding rules, and evaluation protocols (Liu and Paparrizos 2024; Paparrizos et al. 2022; Sarfraz et al. 2024; Schmidl et al. 2022). Some commonly used benchmarks may also contain structural flaws that exaggerate apparent progress (Wu and Keogh 2023). Moreover, point-wise metrics can misrepresent performance when anomalies occupy temporal ranges (Tatbul et al. 2018). These findings motivate investigation of anomaly-score construction independently of the representation backbone.

2.2 Subsequence Representations and Memory-Based Scoring

Temporal anomalies often appear as localized subsequences rather than isolated observations. Shapelet learning identifies representative short patterns (Beggel et al. 2019), while the matrix profile detects unusual subsequences through similarity joins (Yeh et al. 2016). Graph-based and normal-model approaches further support recurrent or variable-length anomaly discovery (Boniol and Palpanas 2020; Boniol et al. 2021; Wang et al. 2018). These methods demonstrate that local comparison is effective when abnormality is defined relative to recurring normal patterns.

Patch-based neural models extend this principle through learned subsequence representations. PatchTST treats local subseries as tokens for long-range modeling (Nie et al. 2023), while CATCH constructs frequency-domain patches for multivariate anomaly detection (Wu et al. 2025). Memory-based visual detectors provide another related direction. PaDiM fits distributions to normal patch embeddings (Defard et al. 2021), whereas PatchCore retrieves representative normal features from a compact memory bank (Roth et al. 2022).

PaAno transfers patch-memory scoring to TSAD (Park and Kang 2026). It trains a one-dimensional convolutional encoder, stores normal patch embeddings, and evaluates queries using nearest-neighbor cosine distance. CovDiff-PaAno retains this representation-learning pipeline. Its research question is whether inference-stage scoring can describe normal embedding geometry and temporal changes more effectively, without redesigning the encoder or its training objectives.

2.3 Covariance-Aware Multi-View Anomaly Evidence

The scoring metric determines which representation-space deviations are considered abnormal. Cosine distance measures angular disagreement but ignores covariance among embedding dimensions. Mahalanobis distance instead scales deviations according to normal covariance (Mahalanobis 1936). Covariance-aware scores have been used for out-of-distribution detection and deep-feature anomaly modeling (Defard et al. 2021; Lee et al. 2018; Rippel et al. 2021), showing that low-variance directions may provide discriminative anomaly evidence.

Raw patches primarily describe local levels and shapes. First-order differences emphasize changes in slope, whereas second-order differences expose changes in curvature and acceleration. Complementary transformations have likewise been used to reveal temporal structures that are weak in a single representation (Jastrzebska 2022). CovDiff-PaAno therefore constructs raw, first-order-difference, and second-order-difference views. Each view uses its own normal memory and covariance estimate while retaining a shared encoder.

This design is related to multi-view learning, although its views are deterministic transformations rather than independent modalities (Wang et al. 2015). The resulting branch scores may have different numerical distributions, so their fusion must also account for score scale (Kriegel et al. 2011). Finally, additional views increase inference work even when encoder parameters are shared. This distinction is relevant because efficient streaming detection requires explicit consideration of computation and memory (Bäßler et al. 2022). Accordingly, CovDiff-PaAno is positioned as a scoring-stage

extension whose benefits, calibration robustness, encoder-sharing assumption, and deployment cost are evaluated separately.

3 Methodology

CovDiff-PaAno is formulated as a controlled scoring-stage extension of PaAno rather than as a new representation backbone. The central question is whether anomaly evidence discarded by raw-view cosine matching can be recovered through covariance-aware comparison and temporally differentiated views, while retaining a single trained patch encoder. To this end, the method uses a shared frozen encoder, constructs a separate normal reference distribution for each view, converts patch-level deviations into aligned point-level scores, and combines the resulting evidence through fixed score-level fusion.

3.1 Problem Setting and Framework Overview

Let

$$X^{\mathrm{tr}} = \left\{x_t^{\mathrm{tr}}\right\}_{t=1}^{T_{\mathrm{tr}}}, \qquad x_t^{\mathrm{tr}} \in \mathbb{R}^C, \tag{1}$$

denote the label-free benchmark training prefix used to construct the reference representation and memory bank, where C is the number of channels. Given an evaluation sequence

$$X = \{x_t\}_{t=1}^{T}, \tag{2}$$

the objective is to produce a continuous anomaly score S_t for every time point t. No anomaly labels are used for encoder training, memory construction, branch-score transformation, or score fusion.

As illustrated in Fig. 1, CovDiff-PaAno represents the input through three complementary views:

$$\mathcal{D}_0(X) = X, \tag{3}$$

$$\mathcal{D}_1(X) = \Delta X, \qquad \Delta x_t = x_{t+1} - x_t, \tag{4}$$

and

$$\mathcal{D}_2(X) = \Delta^2 X, \qquad \Delta^2 x_t = x_{t+2} - 2x_{t+1} + x_t. \tag{5}$$

The raw view describes local value patterns, the first-order view emphasizes slope and trend changes, and the second-order view emphasizes curvature and local acceleration.

All three views are processed by the same frozen patch encoder. However, they do not share their memory statistics: each view constructs a separate normal memory bank and covariance matrix. This design separates representation sharing from distribution modeling. It controls the number of stored and trained encoder parameters while allowing raw, slope-change, and curvature-change embeddings to be evaluated against their respective normal reference distributions.

3.2 Shared Patch Encoder and View-Specific Memory Banks

For a view $b \in \mathcal{B} = \{0, 1, 2\}$, let

$$X^{(b)} = \mathcal{D}_b(X) \tag{6}$$

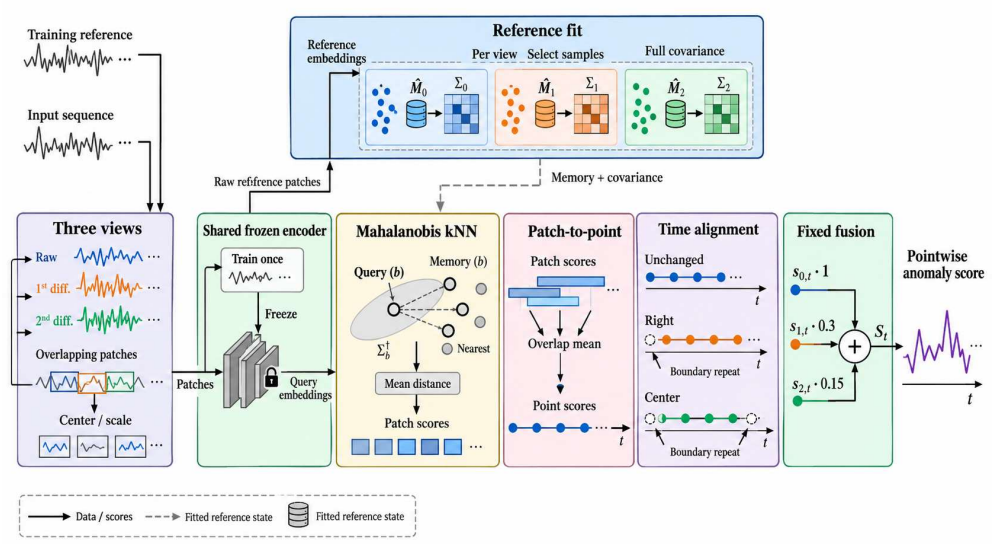


Fig. 1 Overview of CovDiff-PaAno. A single PaAno encoder is trained on normal raw patches and then frozen. Raw, first-order-difference, and second-order-difference patches are processed by the shared encoder, while each view constructs its own normal memory bank and covariance estimate. The aligned branch scores are combined by fixed score-level fusion without introducing an additional trainable fusion module.

denote the corresponding sequence, whose length is $T_b = T - b$. Overlapping patches of length w and stride one are extracted as

$$p_i^{(b)} = \left[x_i^{(b)}, x_{i+1}^{(b)}, \ldots, x_{i+w-1}^{(b)}\right]^\top, \tag{7}$$

where $i = 1, \ldots, T_b - w + 1$ and $p_i^{(b)} \in \mathbb{R}^{C \times w}$.

We retain the one-dimensional convolutional patch encoder f_θ and the representation-learning objective of PaAno (Park and Kang 2026). The encoder is trained only once, using patches extracted from the raw normal training sequence. After training, its parameters are frozen. A patch from any of the three views is embedded as

$$h_i^{(b)} = f_\theta\left(p_i^{(b)}\right), \qquad h_i^{(b)} \in \mathbb{R}^d, \tag{8}$$

where $d = 64$ in the reported implementation.

Before convolutional encoding, each channel of a patch is centered and scaled using its patch-specific statistics, following the normalization setting used by the PaAno implementation. This operation reduces trivial location and scale discrepancies between views, but it is not assumed to eliminate their structural distribution differences. Those differences are instead handled through view-specific memory banks and covariance estimates.

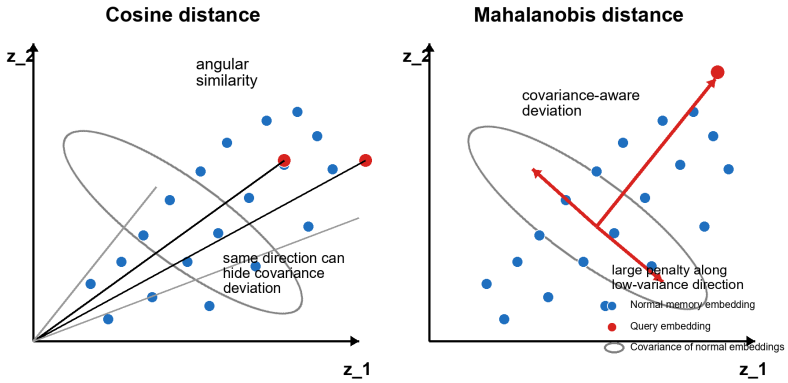


Fig. 2 Conceptual comparison of cosine and Mahalanobis memory scoring. Cosine distance primarily measures angular separation, whereas Mahalanobis distance accounts for the covariance geometry of normal embeddings and assigns greater importance to deviations along low-variance directions.

For each view, the normal training patches are encoded to form a candidate memory set

$$\mathcal{M}_b = \left\{ h_{j,\mathrm{tr}}^{(b)} \right\}_{j=1}^{N_b}. \tag{9}$$

To reduce redundant references, a coreset is constructed by applying mini-batch k-means to ℓ_2-normalized candidate embeddings and retaining the observed embedding nearest to each cluster center. For $N_b > 1$, the effective coreset size is

$$K_b = \min\left(N_b - 1,\, \max\left(500, \mathrm{round}(0.1 N_b)\right)\right). \tag{10}$$

The resulting view-specific coreset is denoted by

$$\widehat{\mathcal{M}}_b = \{m_{b,j}\}_{j=1}^{K_b}. \tag{11}$$

The encoder and its training objective are therefore shared, whereas patch generation, coreset construction, and normal-distribution estimation are performed separately for the raw and differential views.

3.3 Covariance-Aware Memory Scoring

PaAno scores query embeddings using cosine distance to nearby normal memory embeddings. Cosine distance measures angular dissimilarity but does not explicitly account for anisotropy or correlation among embedding dimensions. Consequently, deviations along a tightly concentrated normal direction may be underestimated when their angular direction remains similar to that of normal samples.

CovDiff-PaAno replaces cosine matching with a regularized Mahalanobis metric (Mahalanobis 1936). As conceptually illustrated in Fig. 2, the metric scales a deviation according to the covariance structure of the corresponding normal embedding distribution.

For each view b, the empirical mean of its coreset memory embeddings is

$$\mu_b = \frac{1}{K_b} \sum_{j=1}^{K_b} m_{b,j}. \tag{12}$$

The regularized covariance matrix is estimated as

$$\Sigma_b = \frac{1}{K_b - 1} \sum_{j=1}^{K_b} (m_{b,j} - \mu_b)(m_{b,j} - \mu_b)^\top + \epsilon I_d, \tag{13}$$

where I_d is the d-dimensional identity matrix and $\epsilon = 10^{-5}$. The Moore–Penrose pseudo-inverse $\Sigma_b^\dagger$ is used to retain numerical stability when the empirical covariance is ill-conditioned.

Given a query embedding $h_i^{(b)}$ and a memory embedding $m_{b,j}$, their covariance-aware distance is

$$d_b\left(h_i^{(b)}, m_{b,j}\right) = \sqrt{\left(h_i^{(b)} - m_{b,j}\right)^\top \Sigma_b^\dagger \left(h_i^{(b)} - m_{b,j}\right) + \delta}, \tag{14}$$

where $\delta = 10^{-12}$ prevents numerical instability in the square-root operation.

Let

$$\mathcal{N}_{b,k_b}\left(h_i^{(b)}\right) \tag{15}$$

be the set of the $k_b = \min(k, K_b)$ nearest memory embeddings under this metric. The patch-level anomaly score is

$$a_i^{(b)} = \frac{1}{k_b} \sum_{m \in \mathcal{N}_{b,k_b}(h_i^{(b)})} d_b\left(h_i^{(b)}, m\right). \tag{16}$$

The reported experiments use $k = 3$. Averaging several nearest distances retains the local memory-matching principle of PaAno while reducing dependence on a single reference embedding.

The patch-level scores are converted to point-level scores by averaging all patches covering a given position. Let

$$\mathcal{I}_\tau^{(b)} = \{i \mid i \le \tau \le i + w - 1\} \tag{17}$$

denote the set of patches from view b that contain position τ. Its point-level score is

$$\bar{s}_\tau^{(b)} = \frac{1}{\left|\mathcal{I}_\tau^{(b)}\right|} \sum_{i \in \mathcal{I}_\tau^{(b)}} a_i^{(b)}. \tag{18}$$

A larger value indicates that the local pattern is farther from its view-specific normal patch memory.

3.4 Differential Evidence and Temporal Alignment

The raw branch evaluates local value patterns, but some anomalies are more directly expressed as unusual changes between consecutive observations. The first-order sequence ΔX emphasizes changes in local slope, whereas $\Delta^2 X$ emphasizes curvature and turning behavior. Separate patch memories are therefore constructed as

$$\widehat{\mathcal{M}}_0, \qquad \widehat{\mathcal{M}}_1, \qquad \widehat{\mathcal{M}}_2, \tag{19}$$

with corresponding covariance matrices

$$\Sigma_0, \qquad \Sigma_1, \qquad \Sigma_2. \tag{20}$$

The same frozen encoder f_θ is applied to all three branches. This shared encoder is not assumed to be theoretically optimal for every differential view. Instead, it constitutes a capacity-controlled design choice: all branches use an identical nonlinear mapping, but their anomaly scores are measured relative to different view-specific normal distributions. The adequacy of this choice is evaluated experimentally against fine-tuned and independently trained view encoders.

Because differencing shortens a sequence, the differential scores must be returned to the original time axis before fusion. Let

$$\left\{\bar{s}_\tau^{(1)}\right\}_{\tau=1}^{T-1} \tag{21}$$

and

$$\left\{\bar{s}_\tau^{(2)}\right\}_{\tau=1}^{T-2} \tag{22}$$

denote the first- and second-order point-score sequences before alignment.

The first-order score is right-aligned with the original sequence, and its initial boundary value is replicated:

$$s_{1,t} = \begin{cases} \bar{s}_1^{(1)}, & t = 1, \\ \bar{s}_{t-1}^{(1)}, & 2 \le t \le T. \end{cases} \tag{23}$$

The second-order score is aligned with the center of its corresponding three-point difference, with the two missing boundary positions filled by replication:

$$s_{2,t} = \begin{cases} \bar{s}_1^{(2)}, & t = 1, \\ \bar{s}_{t-1}^{(2)}, & 2 \le t \le T-1, \\ \bar{s}_{T-2}^{(2)}, & t = T. \end{cases} \tag{24}$$

For the raw branch,

$$s_{0,t} = \bar{s}_t^{(0)}. \tag{25}$$

This deterministic alignment uses no labels and changes only the boundary positions introduced by differencing.

3.5 Training-Reference Score Transformation and Fusion

Scores produced by different views may have different empirical scales. We therefore express the fusion rule using a branch-wise transformation

$$\widetilde{s}_{b,t} = g_b\left(s_{b,t}; \mathcal{R}_b\right), \tag{26}$$

where

$$\mathcal{R}_b = \{s_{b,t} \mid t \in \mathcal{T}_{\mathrm{tr}}\} \tag{27}$$

contains only scores from the unlabeled label-free training-reference prefix. Test scores and anomaly labels are not used to estimate the transformation parameters.

The default transformation is the identity mapping:

$$g_b^{\mathrm{none}}(u) = u. \tag{28}$$

To examine whether explicit calibration changes the conclusions, we additionally consider the following training-reference transformations:

$$g_b^{\mathrm{z}}(u) = \frac{u - \mu_{\mathcal{R}_b}}{\sigma_{\mathcal{R}_b}}, \tag{29}$$

$$g_b^{\mathrm{robust}}(u) = \frac{u - \mathrm{median}(\mathcal{R}_b)}{1.4826\ \mathrm{MAD}(\mathcal{R}_b)}, \tag{30}$$

$$g_b^{\mathrm{minmax}}(u) = \frac{u - \min(\mathcal{R}_b)}{\max(\mathcal{R}_b) - \min(\mathcal{R}_b)}, \tag{31}$$

and the smoothed empirical cumulative distribution transform

$$g_b^{\mathrm{ECDF}}(u) = \frac{|\{v \in \mathcal{R}_b : v \le u\}| + 0.5}{|\mathcal{R}_b| + 1}. \tag{32}$$

Near-zero scale terms are handled using finite fallback scales in the implementation. These transformations are evaluated as robustness variants rather than selected separately for each dataset. The main CovDiff-PaAno configuration uses g_b^{none}.

The final anomaly score is

$$S_t = \widetilde{s}_{0,t} + \alpha\widetilde{s}_{1,t} + \beta\widetilde{s}_{2,t}, \tag{33}$$

where the raw-branch coefficient is fixed to one. We use

$$\alpha = 0.3, \qquad \beta = 0.15 \tag{34}$$

for all datasets and random seeds unless otherwise specified. Once fixed, this fusion introduces neither a trainable module nor an additional optimization objective. The coefficients should be interpreted as fixed evidence-combination factors rather than probabilistic branch importances.

Table 1 TSB-AD collections processed in the reproduced experiments.

Dataset	Type	Series	Collections	Channels
TSB-AD-U	Univariate	870	23	1
TSB-AD-M	Multivariate	200	17	Multiple

3.6 Training and Inference Characteristics

CovDiff-PaAno stores and trains only one encoder. If the PaAno encoder contains P parameters, then

$$P_{\text{CovDiff}} = P_{\text{PaAno}} = P. \tag{35}$$

The differential branches therefore introduce no additional trainable encoder parameters and require no additional encoder-training run in the shared configuration.

This parameter efficiency does not imply inference-cost equivalence. CovDiff-PaAno performs three encoder passes, constructs three view-specific memory banks, estimates three covariance matrices, and carries out three nearest-memory searches. Its embedding-reference storage is proportional to

$$d \sum_{b=0}^{2} K_b, \tag{36}$$

in addition to the three $d \times d$ covariance matrices. By contrast, the PaAno baseline uses only the raw-view memory bank.

Accordingly, we describe CovDiff-PaAno as a single-encoder and training-efficient extension, rather than as a latency-neutral detector. The practical trade-off between unchanged trainable capacity and increased inference time and memory consumption is quantified in the experimental evaluation.

4 Experiments

4.1 Experimental Setup

Datasets and evaluation scope.

We evaluated CovDiff-PaAno on the univariate and multivariate TSB-AD collections, denoted by TSB-AD-U and TSB-AD-M, respectively (Liu and Paparrizos 2024). All available benchmark files were processed, comprising 870 univariate series from 23 source collections and 200 multivariate series from 17 source collections. Table 1 summarizes this evaluation scope.

For each series, observations preceding the benchmark-defined training boundary formed the label-free training-reference prefix, whereas the complete sequence was scored. The encoder, memory banks, covariance estimates, and score transformations were constructed without anomaly labels. Labels were accessed only during metric calculation.

Table 2 Default configuration for the controlled PaAno and CovDiff-PaAno experiments.

Item	Setting
Encoder optimization iterations	200
Batch size	512
Learning rate	10^{-4}
Patch stride	1
Patch length	64 (TSB-AD-U); 96 (TSB-AD-M)
Patch-wise normalization	Enabled
Nearest memory neighbors	$k = 3$
Coreset target ratio	0.1
Encoder assignment	Shared and frozen
Score transformation	Identity (`none`)
First-order fusion weight	$\alpha = 0.3$
Second-order fusion weight	$\beta = 0.15$
Main comparison repetitions	10 independent seeds

Removing labeled anomalies from a training prefix would introduce supervision. We therefore retained all prefix observations. An audit showed that 22 of the 200 TSB-AD-M prefixes contained 1,558 labeled points, which remained unlabeled reference data during training.

Comparison protocol.

The primary controlled baseline was our reproduction of PaAno using raw-view cosine memory matching (Park and Kang 2026). Both methods used identical data, training boundaries, encoders, optimization settings, random seeds, and evaluation procedures. Only the investigated scoring components differed.

The main tables also reproduce published results for PatchTST, DLinear, NLinear, DCdetector, iTransformer, CATCH, KAN-AD, DADA, MOMENT, TimesFM, and PaAno. These methods were not rerun locally and serve only as contextual references. Claims of improvement are therefore based on the controlled comparison between reproduced PaAno and CovDiff-PaAno.

Implementation details.

Table 2 summarizes the default configuration used in the controlled experiments.

Unless stated otherwise, all three views used one encoder trained on raw training-reference patches and then frozen. Differential views required no additional encoder optimization, and anomaly scoring used full covariance matrices.

The PyTorch encoder comprised four one-dimensional convolutional blocks with output widths of 128, 256, 128, and 64. Global average pooling produced a 64-dimensional embedding for each patch.

Evaluation metrics and threshold selection.

Following the PaAno and TSB-AD settings (Park and Kang 2026; Paparrizos et al. 2022; Liu and Paparrizos 2024; Sarfraz et al. 2024), we report AUC-ROC, AUC-PR, Point-F1, VUS-ROC, VUS-PR, and Range-F1. AUC-ROC and AUC-PR measure point-wise ranking, whereas the VUS metrics integrate performance across score thresholds and temporal-tolerance windows.

VUS computation used 250 score thresholds and window sizes from zero to an autocorrelation-derived maximum. The estimator examined at most the first 20,000 observations of the first channel. It selected a local autocorrelation peak between lags 3 and 300, with 125 used as the fallback.

Point-F1 was maximized over the precision–recall thresholds of each sequence. Range-F1 used range-based precision and recall with an existence weight of 0.2 and was maximized over 100 uniformly spaced thresholds. Both are oracle summaries that use evaluation labels for retrospective threshold selection. Conventional point adjustment was neither reported nor used.

Each metric was computed independently for every series and then macro-averaged across the corresponding benchmark collection. Consequently, each series contributed equally regardless of its length or anomaly count.

Repetitions and statistical reporting.

The controlled PaAno–CovDiff-PaAno comparison used 10 independent random seeds. The main tables report the mean and standard deviation of the seed-level benchmark averages.

The original component ablations used one fixed seed and are interpreted as diagnostic comparisons. The shared-encoder and score-calibration studies used all 200 TSB-AD-M series with seeds 2000, 2021, and 2022.

Each calibration transformation was fitted exclusively to the corresponding branch's training-reference scores. Paired differences against identity transformation were averaged across seeds within each sequence. We then obtained 95% confidence intervals from 10,000 bootstrap resamples of the 200 unique sequences. This sequence-level procedure avoids treating repeated runs as independent observations. Wins, ties, losses, and rank-biserial effect sizes provide complementary descriptive evidence.

Efficiency measurement.

Efficiency was measured on all 200 TSB-AD-M series using seed 2000. Both methods ran in separate processes under Windows, Python 3.11.16, PyTorch 2.11.0 with CUDA 12.8, and an NVIDIA GeForce RTX 4090 GPU.

CUDA operations were synchronized around every timed stage. Training time covered encoder optimization, including branch-specific encoders when required. Inference time covered memory construction, covariance estimation, patch encoding, nearest-memory scoring, and patch-to-point aggregation. Metric computation and result serialization were excluded.

Table 3 Results on TSB-AD-M. Higher values are better. Published results are included as contextual references, whereas the final two rows constitute the controlled comparison used to support our main claim. Values in the table are rounded to two decimal places.

Method	AUC-ROC	AUC-PR	Point-F1	VUS-ROC	VUS-PR	Range-F1
Results reported in prior work						
PatchTST (Nie et al. 2023)	0.65	0.26	0.32	0.71	0.28	0.26
DLinear (Zeng et al. 2023)	0.66	0.27	0.32	0.70	0.29	0.26
NLinear (Zeng et al. 2023)	0.65	0.24	0.31	0.70	0.29	0.28
DCdetector (Yang et al. 2023)	0.50	0.06	0.10	0.56	0.10	0.10
iTransformer (Liu et al. 2024)	0.63	0.23	0.28	0.70	0.29	0.23
CATCH (Wu et al. 2025)	0.67	0.24	0.30	0.73	0.30	0.27
KAN-AD (Zhou et al. 2025)	0.73	0.38	0.42	0.75	0.41	0.41
DADA (Shentu et al. 2025)	0.69	0.31	0.35	0.73	0.31	0.25
PaAno (reported) (Park and Kang 2026)	0.76	0.38	0.43	0.79	0.43	0.41
Controlled reproduction: 10-seed means						
PaAno (reproduced)	0.76	0.37	0.42	0.80	0.43	0.40
CovDiff-PaAno	**0.78**	**0.40**	**0.45**	**0.82**	**0.45**	**0.44**

Peak GPU memory was the maximum allocated CUDA memory during training or inference. End-to-end time additionally included loading, preprocessing, and data-loader construction. Throughput was calculated from the total number of processed time points divided by inference time.

4.2 Main Results

Tables 3 and 4 report results on TSB-AD-M and TSB-AD-U, respectively. Published results provide context only because their implementations and preprocessing procedures may differ. The primary evidence comes from the controlled comparison, in which reproduced PaAno and CovDiff-PaAno shared the same data, encoder, training settings, evaluation pipeline, and ten random seeds.

On TSB-AD-M, CovDiff-PaAno improved all six metrics over the reproduced PaAno baseline. AUC-ROC increased from 0.7612 ± 0.0025 to 0.7843 ± 0.0021, while Range-F1 increased from 0.4043 ± 0.0090 to 0.4384 ± 0.0057. The relatively small between-seed deviations indicate stable performance across repeated runs. Improvements in both ranking-based and range-aware metrics support the effectiveness of covariance-aware multi-order scoring within an otherwise unchanged PaAno pipeline.

On TSB-AD-U, CovDiff-PaAno improved five metrics and matched VUS-ROC after rounding. AUC-ROC increased from 0.8532 ± 0.0032 to 0.8626 ± 0.0008, and VUS-PR increased from 0.4946 ± 0.0066 to 0.5152 ± 0.0018. Range-F1 also increased from 0.4822 ± 0.0045 to 0.4924 ± 0.0034. These gains show that the proposed formulation remains effective for univariate series, although its improvements are generally larger on TSB-AD-M. This pattern is consistent with the motivation for covariance modelling, but does not by itself establish the source of the difference.

Table 4 Results on TSB-AD-U. Higher values are better. Published results are presented for context only. The principal comparison is between the reproduced PaAno baseline and CovDiff-PaAno under the same ten-seed experimental protocol. Values are rounded to two decimal places; cells that become tied after rounding are not emphasized.

Method	AUC-ROC	AUC-PR	Point-F1	VUS-ROC	VUS-PR	Range-F1
Results reported in prior work						
PatchTST (Nie et al. 2023)	0.63	0.21	0.25	0.75	0.26	0.22
DLinear (Zeng et al. 2023)	0.62	0.21	0.26	0.74	0.25	0.22
NLinear (Zeng et al. 2023)	0.62	0.18	0.23	0.72	0.23	0.20
DCdetector (Yang et al. 2023)	0.50	0.05	0.10	0.56	0.09	0.10
iTransformer (Liu et al. 2024)	0.61	0.16	0.21	0.74	0.22	0.18
KAN-AD (Zhou et al. 2025)	0.80	0.41	0.44	0.82	0.43	0.43
DADA (Shentu et al. 2025)	0.71	0.29	0.38	0.77	0.31	0.31
MOMENT (FT) (Goswami et al. 2024)	0.69	0.30	0.35	0.76	0.39	0.35
MOMENT (ZS) (Goswami et al. 2024)	0.68	0.30	0.35	0.75	0.38	0.36
TimesFM (Das et al. 2024)	0.67	0.28	0.34	0.74	0.30	0.34
PaAno (reported) (Park and Kang 2026)	0.86	0.46	0.51	0.89	0.52	0.48
Controlled reproduction: 10-seed means						
PaAno (reproduced)	0.85	0.42	0.47	0.88	0.49	0.48
CovDiff-PaAno	**0.86**	**0.44**	**0.49**	0.88	**0.52**	**0.49**

Across both collections, the controlled comparison supports the main empirical claim: covariance-aware multi-order discrepancy scoring improves PaAno without changing its encoder or data pipeline. The published rows should not be interpreted as controlled evidence of superiority over every prior method. The following analyses instead examine component contributions, the shared-encoder design, score calibration, and computational cost.

4.3 Component Ablation

Table 5 examines three questions: whether covariance-aware scoring improves the raw-view baseline, whether differential views provide additional evidence, and whether the two differential orders are complementary. The cosine baseline retains the original PaAno scorer, whereas all remaining configurations use Mahalanobis scoring for their active views. Every configuration within a dataset used the same seed, encoder, data pipeline, and evaluation procedure. This single-seed experiment is therefore a diagnostic component analysis rather than a substitute for the ten-seed results in Tables 3 and 4.

First, replacing cosine matching with Mahalanobis scoring improved all six metrics on both datasets while retaining only the raw view. On TSB-AD-M, AUC-ROC increased from 0.7608 to 0.7782 and Range-F1 increased from 0.4152 to 0.4328.

Table 5 Single-seed diagnostic ablation of covariance-aware scoring and differential evidence. Except for the cosine baseline, all active views use Mahalanobis scoring. Differential configurations retain the raw view and add the indicated temporal views using the default fusion weights. Bold values denote the best result within each dataset.

Dataset	Active scoring components	Point-wise metrics ↑			Range-wise metrics ↑		
		AUC-ROC	AUC-PR	Point-F1	VUS-ROC	VUS-PR	Range-F1
TSB-AD-M	Raw view, cosine (PaAno)	0.7608	0.3901	0.4333	0.7932	0.4389	0.4152
	Raw view, Mahalanobis	0.7782	0.3992	0.4458	0.8102	0.4513	0.4328
	Raw $+\Delta X$	0.7848	0.4043	0.4500	0.8165	0.4521	0.4380
	Raw $+\Delta^2 X$	**0.7852**	**0.4058**	**0.4509**	**0.8167**	**0.4534**	0.4438
	Raw $+\Delta X + \Delta^2 X$ (CovDiff-PaAno)	0.7850	0.4054	0.4508	0.8165	0.4531	**0.4461**
TSB-AD-U	Raw view, cosine (PaAno)	0.8499	0.4170	0.4674	0.8723	0.4928	0.4793
	Raw view, Mahalanobis	0.8569	0.4192	0.4704	0.8787	0.4957	0.4809
	Raw $+\Delta X$	0.8611	0.4307	0.4841	**0.8816**	0.5091	**0.4893**
	Raw $+\Delta^2 X$	0.8609	0.4340	0.4874	0.8809	0.5115	0.4879
	Raw $+\Delta X + \Delta^2 X$ (CovDiff-PaAno)	**0.8612**	**0.4347**	**0.4881**	0.8813	**0.5117**	0.4888

The improvements were smaller but directionally consistent on TSB-AD-U. Because the active view remained unchanged, this comparison isolates the contribution of covariance-aware scoring.

Second, differential evidence provided gains beyond the raw-view Mahalanobis configuration. On TSB-AD-M, adding both differential orders increased Range-F1 from 0.4328 to 0.4461. On TSB-AD-U, it increased AUC-PR from 0.4192 to 0.4347 and Point-F1 from 0.4704 to 0.4881. Each differential branch also improved every metric when added individually to the raw-view scorer. Thus, neither covariance modelling nor a single differential order alone accounts for the complete improvement.

Third, no configuration dominated every metric. The second-order branch achieved the best five metrics on TSB-AD-M, whereas combining both orders produced the highest Range-F1. On TSB-AD-U, the first-order branch produced the highest VUS-ROC and Range-F1, while the complete configuration led the remaining four metrics. We therefore retain both differential branches as the default because they provide the most balanced performance across datasets and evaluation perspectives.

These findings are consistent with the intended division of evidence: the raw branch captures static distributional deviation, whereas the differential branches emphasize temporal change. However, aggregate metrics cannot establish that the two differential orders exclusively detect specific anomaly morphologies. Moreover, this single-seed ablation explains component behavior within a controlled run; overall robustness is supported by the ten-seed main comparison.

4.4 Shared-Encoder Validation

The default CovDiff-PaAno configuration reuses a frozen encoder trained on raw patches for all three views. This design assumes that raw-view representations remain informative for the differential inputs. We tested whether adapting the encoder to each differential view materially improves performance.

Three encoder assignments were evaluated on TSB-AD-M. Shared frozen reused the raw-trained encoder without further optimization. View-finetuned initialized two encoder copies from the raw-trained model and adapted them to the corresponding differential views. View-specific independently initialized and trained one encoder for

Table 6 Comparison of encoder-assignment strategies on TSB-AD-M. Results are reported as mean ± standard deviation over three random seeds. Higher values are better, and the best mean for each metric is shown in bold.

Encoder strategy	AUC-ROC	AUC-PR	Point-F1	VUS-ROC	VUS-PR	Range-F1
Shared frozen	0.7825 ± 0.0027	0.3965 ± 0.0044	0.4436 ± 0.0031	0.8145 ± 0.0030	0.4453 ± 0.0044	0.4339 ± 0.0019
View-finetuned	0.7834 ± 0.0015	0.3988 ± 0.0049	0.4462 ± 0.0043	0.8151 ± 0.0018	0.4466 ± 0.0040	$\mathbf{0.4378 \pm 0.0065}$
View-specific	$\mathbf{0.7872 \pm 0.0016}$	$\mathbf{0.4000 \pm 0.0044}$	$\mathbf{0.4477 \pm 0.0030}$	$\mathbf{0.8190 \pm 0.0017}$	$\mathbf{0.4492 \pm 0.0039}$	0.4374 ± 0.0010

each view. Patch extraction, training objectives, memory construction, scoring, fusion weights, and evaluation procedures remained unchanged. Each strategy was evaluated with seeds 2000, 2021, and 2022.

Table 6 shows that view-specific training achieved the highest mean on five metrics, whereas view-finetuning achieved the highest Range-F1. Differential-view adaptation can therefore provide a modest improvement. However, the shared frozen encoder remained within 0.005 of the best mean on every metric. Its largest gaps were 0.0047 for AUC-ROC, 0.0039 for VUS-PR, and 0.0039 for Range-F1.

These differences were comparable in magnitude to the between-seed variation observed over the three runs. This descriptive comparison does not establish statistical equivalence, but it indicates that view-dependent training yields limited aggregate gains under the evaluated conditions.

The shared strategy avoids two additional encoder-training procedures and separate branch-specific parameters. We therefore retain it as the default accuracy–efficiency trade-off, rather than claiming that raw-trained representations are optimal for differential inputs. Section 4.6 quantifies the corresponding computational benefit.

This experiment was limited to TSB-AD-M and three seeds. Its findings support the selected multivariate configuration but do not guarantee equivalent behavior across all univariate datasets or encoder architectures.

4.5 Score Calibration and Fusion-Weight Robustness

Fixed score-level fusion is meaningful only when branch scales are comparable and performance does not depend on a narrowly selected weight configuration. We therefore examined branch-scale statistics, label-free calibration, and local fusion-weight sensitivity.

Branch-scale diagnostics.

For each sequence and seed, we calculated the standard-deviation and interquartile-range ratios of each differential score to the raw score. These statistics used only the benchmark training prefix. Across 600 seed–sequence combinations on TSB-AD-M, the median standard-deviation ratios were 0.916 and 0.882 for the first- and second-order branches, respectively. The corresponding interquartile-range ratios were 0.975 and 0.938. The branch distributions were therefore not identical, but their typical dispersions had comparable orders of magnitude.

Label-free score calibration.

We compared the untransformed scores with standard z-score, robust z-score, min–max, and empirical-CDF calibration. Each transformation was fitted separately for

Table 7 Effect of label-free branch-score calibration on TSB-AD-M. Results are reported as mean ± standard deviation over three random seeds. Calibration parameters were estimated separately for each branch using only its benchmark training-prefix scores.

Calibration method	AUC-ROC	VUS-PR	Range-F1
None (default)	0.7825 ± 0.0027	0.4453 ± 0.0044	0.4339 ± 0.0019
Z-score	0.7828 ± 0.0025	0.4454 ± 0.0041	0.4349 ± 0.0019
Robust z-score	0.7824 ± 0.0032	0.4452 ± 0.0044	0.4363 ± 0.0002
Min–max	0.7823 ± 0.0024	0.4447 ± 0.0041	0.4347 ± 0.0031
Empirical CDF	0.6861 ± 0.0067	0.2201 ± 0.0085	0.1216 ± 0.0026

every sequence and branch using only its training-reference scores. Neither anomaly labels nor evaluation-set statistics were used. The calibrated branches retained the default weights $\alpha = 0.3$ and $\beta = 0.15$.

Table 7 reports representative metrics over seeds 2000, 2021, and 2022. Standard z-score, robust z-score, and min–max calibration closely matched the untransformed fusion. Across all six computed metrics, their largest absolute change from the default was 0.0024. The corresponding sequence-level paired bootstrap intervals included zero, providing no evidence of a consistent improvement.

Empirical-CDF calibration, in contrast, substantially reduced every reported metric. This transformation replaces branch magnitudes with bounded within-branch ranks before fusion. Its poor performance is consistent with relative score magnitude carrying useful anomaly evidence, although this result does not establish that monotonic calibration is generally harmful.

Because the conventional label-free transformations produced no consistent gain, we retain the untransformed fusion. This choice avoids sequence-specific calibration parameters and preserves a simpler inference procedure. The result does not imply identical branch distributions. It shows only that their observed scale differences do not explain the effectiveness of the default fusion on TSB-AD-M.

Fusion-weight robustness.

We evaluated $(\alpha, \beta) \in \{(0.2, 0.1), (0.3, 0.15), (0.4, 0.2)\}$ on TSB-AD-M and TSB-AD-U using a fixed seed. All other components remained unchanged. Because every pair preserves a two-to-one ratio, this experiment tests the overall strength of differential evidence rather than independent optimization of the two weights.

Figure 3 shows limited variation across the tested neighborhood. On TSB-AD-M, AUC-ROC ranged from 0.7846 to 0.7860 and Range-F1 ranged from 0.4441 to 0.4461. On TSB-AD-U, the corresponding ranges were 0.8605–0.8612 and 0.4888–0.4895. No weight pair performed best on every metric. We therefore retain $(\alpha, \beta) = (0.3, 0.15)$ as a balanced default rather than a universally optimal setting.

Together, the scale, calibration, and weight analyses show that the reported behavior does not rely on severe branch-scale imbalance or one isolated weight pair. This conclusion remains local to the evaluated configurations. Calibration was tested on TSB-AD-M with three seeds, whereas weight sensitivity used one seed and varied both

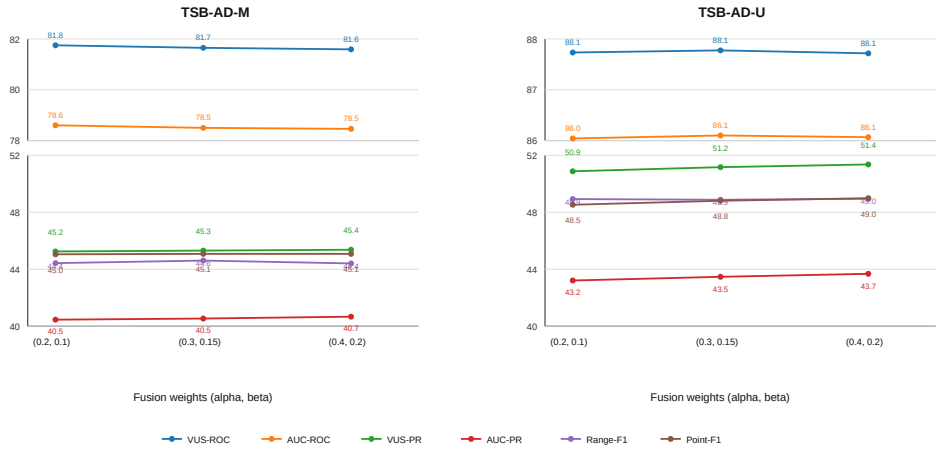


Fig. 3 Sensitivity of CovDiff-PaAno to the differential fusion weights on TSB-AD-M and TSB-AD-U. Values are percentages from a single-seed diagnostic experiment. Separate vertical ranges are used for the ROC-based metrics and the remaining metrics to make small changes visible. The tested weight pairs are $(0.2, 0.1)$, $(0.3, 0.15)$, and $(0.4, 0.2)$ for (α, β), respectively.

weights at a fixed ratio. These diagnostics are not treated as independent evidence for the main performance claim.

4.6 Efficiency and Complexity

We profiled all 200 TSB-AD-M sequences, comprising 21,552,097 time points. Both methods used seed 2000, a batch size of 512, and 200 training iterations. Measurements were obtained using PyTorch 2.11.0 with CUDA 12.8 on an NVIDIA GeForce RTX 4090. Training time covers encoder optimization, whereas inference includes memory construction, covariance estimation, scoring, and patch-to-point aggregation. End-to-end time excludes metric computation and result serialization. Because each configuration was profiled once, the measurements characterize computational cost rather than timing uncertainty.

Let B be the number of active views, Q_b and K_b the numbers of query and memory embeddings for view b, and d the embedding dimension. Covariance estimation and inversion require

$$\mathcal{O}\left(\sum_{b=1}^{B} K_b d^2 + B d^3\right).$$

The current implementation explicitly applies the inverse covariance matrix to every query–memory difference vector. Its dominant scoring complexity is therefore

$$\mathcal{O}\left(\sum_{b=1}^{B} Q_b K_b d^2\right).$$

Table 8 Efficiency comparison on the complete TSB-AD-M benchmark. Times are mean values per sequence over 200 sequences. The stored parameter count is averaged across sequences because the encoder input dimensionality depends on the number of variables in each sequence. Peak memory denotes the maximum allocated GPU memory during inference.

Configuration	Stored params (M)	Training (s/sequence)	Inference (s/sequence)	End-to-end (s/sequence)	Inference peak (MB)	Throughput (points/s)
PaAno	0.397	3.50	4.59	8.36	192	23,476
CovDiff-PaAno	0.397	3.54	15.94	20.08	1,919	6,759
CovDiff-PaAno / PaAno	1.00×	1.01×	3.47×	2.40×	10.01×	0.29×

With batches of q_b query embeddings, the dominant temporary tensor requires

$$\mathcal{O}\left(\max_b q_b K_b d\right)$$

memory. In comparison, the single-view cosine scorer used by PaAno requires $\mathcal{O}(QKd)$ pairwise operations. Encoder sharing avoids additional trainable models, but it does not remove the computational overhead of multi-view Mahalanobis scoring.

Table 8 shows that CovDiff-PaAno retains the same mean parameter count as PaAno, approximately 0.397 million. Mean training time increases only from 3.50 to 3.54 seconds per sequence. The shared encoder therefore preserves the parameter and training efficiency of the original backbone.

The principal overhead occurs during inference. Mean inference time increases from 4.59 to 15.94 seconds per sequence, corresponding to a 3.47× increase. Peak inference memory increases from 192 to 1,919 MB, while throughput decreases from 23,476 to 6,759 points per second. This cost arises primarily from processing three views and constructing dense query–memory difference tensors for full-covariance scoring. Although peak allocation remains below 2 GB in the evaluated environment, the increase may constrain deployment on longer sequences or limited-memory devices.

CovDiff-PaAno should therefore be characterized as parameter- and training-efficient, but not inference-cost-free. Its improved anomaly ranking is obtained at a measurable inference-time and memory cost. Chunked distance computation, memory-bank reduction, and structured covariance approximations are promising directions for improving this trade-off.

5 Discussion

Taken together, the experiments support a bounded but important conclusion: patch-based anomaly detection depends on more than encoder capacity. CovDiff-PaAno improved the reproduced PaAno pipeline without changing its backbone or training objective. The ablations instead attributed the gains to covariance-aware comparison and additional temporal evidence. Representation, temporal view, and scoring geometry should therefore be designed as coupled components rather than independent choices. This shifts part of the performance bottleneck from representation learning to the compatibility between embeddings and inference-time decision rules.

The benefit of Mahalanobis scoring is consistent with anisotropic structure in normal embeddings. Unlike cosine distance, it discounts variation along high-variance directions and emphasizes deviations along low-variance directions (Mahalanobis 1936). Related anomaly detectors have applied the same principle to spatial embedding distributions (Defard et al. 2021). CovDiff-PaAno extends it to patch-memory comparison without redesigning the representation backbone. Differential views address a separate limitation by making local rates of change and curvature more explicit before scoring. The ablation patterns support complementary temporal evidence, but they do not establish an exclusive correspondence between differential orders and anomaly types. They also do not prove that full covariance is uniquely optimal. Comparisons with Euclidean, diagonal-covariance, and shrinkage estimators are needed to isolate the contribution of cross-dimensional covariance.

The encoder and calibration analyses clarify the practical role of the default configuration. A shared raw-trained encoder retained most of the performance achieved by view-dependent training, while avoiding additional models. Conventional label-free calibration produced no consistent gain, and performance remained stable within the tested weight neighborhood. These results support a simple default, but do not establish encoder equivalence or globally optimal fusion weights. The efficiency results further narrow the lightweight characterization of CovDiff-PaAno. Encoder sharing preserves parameter and training efficiency, whereas multi-view Mahalanobis scoring increases inference time and temporary memory. Deployment should therefore consider this accuracy–efficiency trade-off explicitly. Chunked scoring, structured covariance, and label-free adaptive fusion may reduce the cost without expanding the backbone.

Several limitations delimit the present evidence. First, the primary controlled comparison concerns PaAno; published results for other methods provide context rather than locally reproduced head-to-head evidence. Second, component and weight ablations use a single seed, so small differences among configurations remain diagnostic rather than statistically conclusive. Third, the boundary-alignment rule has not been compared with padding, forward alignment, or boundary exclusion. Fourth, full-covariance estimation may become unstable when the memory bank is small relative to the embedding dimension. Future studies should test alternative covariance structures, repeat component analyses across seeds, and evaluate transfer to other patch encoders and streaming settings. These extensions will determine how broadly inference-stage statistical design generalizes beyond the evaluated PaAno and TSB-AD setting.

6 Conclusion

This work shows that a key bottleneck in patch-based time-series anomaly detection lies not only in representation learning, but also in inference-stage scoring. CovDiff-PaAno demonstrates that modelling anisotropic normal variation and temporal change can improve detection without redesigning the learned backbone. The controlled experiments consistently support the value of covariance-aware comparison and complementary differential evidence. These findings shift the focus from treating an embedding as the final solution toward viewing representation, temporal view, and scoring geometry as coupled components of an anomaly detector.

More broadly, inference-stage statistical mechanisms provide a modular route for strengthening existing patch-based systems when training a larger backbone is impractical. This principle may also extend to pretrained time-series encoders because it operates on their embedding spaces rather than their internal architectures. The present evidence remains limited to PaAno and TSB-AD, while full-covariance scoring introduces nontrivial inference costs. Future work should therefore investigate structured covariance estimation, memory-efficient scoring, label-free adaptive fusion, and transfer across backbones and streaming environments. Overall, this study establishes inference-stage statistical design as a distinct research direction that complements the prevailing emphasis on increasingly complex representation models.

References

Audibert J, Michiardi P, Guyard F, et al (2020) USAD: Unsupervised anomaly detection on multivariate time series. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Association for Computing Machinery, pp 3395–3404, https://doi.org/10.1145/3394486.3403392

Bäßler D, Kortus T, Gühring G (2022) Unsupervised anomaly detection in multivariate time series with online evolving spiking neural networks. Machine Learning 111:1377–1408. https://doi.org/10.1007/s10994-022-06129-4

Beggel L, Kausler BX, Schiegg M, et al (2019) Time series anomaly detection based on shapelet learning. Computational Statistics 34:945–976. https://doi.org/10.1007/s00180-018-0824-9

Blázquez-García A, Conde A, Mori U, et al (2021) A review on outlier/anomaly detection in time series data. ACM Computing Surveys 54(3):56:1–56:33. https://doi.org/10.1145/3444690

Boniol P, Palpanas T (2020) Series2graph: Graph-based subsequence anomaly detection for time series. Proceedings of the VLDB Endowment 13(11):1821–1834. https://doi.org/10.14778/3407790.3407792

Boniol P, Linardi M, Roncallo F, et al (2021) Unsupervised and scalable subsequence anomaly detection in large data series. The VLDB Journal 30(6):909–931. https://doi.org/10.1007/s00778-021-00655-8

Breunig MM, Kriegel HP, Ng RT, et al (2000) LOF: Identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. Association for Computing Machinery, pp 93–104, https://doi.org/10.1145/342009.335388

Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: A survey. ACM Computing Surveys 41(3):15:1–15:58. https://doi.org/10.1145/1541880.1541882

Choi K, Yi J, Park C, et al (2021) Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines. IEEE Access 9:120043–120065. https://doi.org/10.1109/ACCESS.2021.3107975

Das A, Kong W, Sen R, et al (2024) A decoder-only foundation model for time-series forecasting. In: Proceedings of the 41st International Conference on Machine Learning, Proceedings of Machine Learning Research, vol 235. PMLR, Vienna, Austria, pp 10148–10167, URL https://proceedings.mlr.press/v235/das24c.html

Defard T, Setkov A, Loesch A, et al (2021) PaDiM: A patch distribution modeling framework for anomaly detection and localization. In: Pattern Recognition. ICPR International Workshops and Challenges, Lecture Notes in Computer Science, vol 12664. Springer, Cham, pp 475–489, https://doi.org/10.1007/978-3-030-68799-1_35

Deng A, Hooi B (2021) Graph neural network-based anomaly detection in multivariate time series. Proceedings of the AAAI Conference on Artificial Intelligence 35(5):4027–4035. https://doi.org/10.1609/aaai.v35i5.16523

Farag A, Abdelkader H, Salem R (2022) Parallel graph-based anomaly detection technique for sequential data. Journal of King Saud University - Computer and Information Sciences 34(1):1446–1454. https://doi.org/10.1016/j.jksuci.2019.09.009

Fawzy AH, Wassif K, Moussa H (2023) Framework for automatic detection of anomalies in DevOps. Journal of King Saud University - Computer and Information Sciences 35(3):8–19. https://doi.org/10.1016/j.jksuci.2023.02.010

Goswami M, Szafer K, Choudhry A, et al (2024) MOMENT: A family of open time-series foundation models. In: Proceedings of the 41st International Conference on Machine Learning, Proceedings of Machine Learning Research, vol 235. PMLR, Vienna, Austria, pp 16115–16152, URL https://proceedings.mlr.press/v235/goswami24a.html

Hundman K, Constantinou V, Laporte C, et al (2018) Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Association for Computing Machinery, pp 387–395, https://doi.org/10.1145/3219819.3219845

Jastrzebska A (2022) Time series classification through visual pattern recognition. Journal of King Saud University - Computer and Information Sciences 34(2):134–142. https://doi.org/10.1016/j.jksuci.2019.12.012

Kriegel HP, Kröger P, Schubert E, et al (2011) Interpreting and unifying outlier scores. In: Proceedings of the 2011 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pp 13–24, https://doi.org/10.1137/1.9781611972818.2

Lee K, Lee K, Lee H, et al (2018) A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In: Advances in Neural Information Processing Systems

Liu FT, Ting KM, Zhou ZH (2008) Isolation forest. In: Proceedings of the 8th IEEE International Conference on Data Mining. IEEE, pp 413–422, https://doi.org/10.1109/ICDM.2008.17

Liu Q, Paparrizos J (2024) The elephant in the room: Towards a reliable time-series anomaly detection benchmark. In: Advances in Neural Information Processing Systems, pp 108231–108261, https://doi.org/10.52202/079017-3437

Liu Y, Hu T, Zhang H, et al (2024) iTransformer: Inverted transformers are effective for time series forecasting. In: International Conference on Learning Representations, URL https://openreview.net/forum?id=JePfAI8fah

Ma L, Zhang X, Wang K, et al (2025) Generative adversarial message passing-based anomaly detection. Journal of King Saud University Computer and Information Sciences 37:5. https://doi.org/10.1007/s44443-025-00021-6

Mahalanobis PC (1936) On the generalized distance in statistics. Proceedings of the National Institute of Sciences of India 2(1):49–55

Malhotra P, Vig L, Shroff G, et al (2015) Long short term memory networks for anomaly detection in time series. In: Proceedings of the 23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, pp 89–94

Nie Y, Nguyen NH, Sinthong P, et al (2023) A time series is worth 64 words: Long-term forecasting with transformers. In: International Conference on Learning Representations, URL https://openreview.net/forum?id=Jbdc0vTOcol

Pang G, Shen C, Cao L, et al (2021) Deep learning for anomaly detection: A review. ACM Computing Surveys 54(2):1–38. https://doi.org/10.1145/3439950

Paparrizos J, Boniol P, Palpanas T, et al (2022) Volume under the surface: A new accuracy evaluation measure for time-series anomaly detection. Proceedings of the VLDB Endowment 15(11):2774–2787. https://doi.org/10.14778/3551793.3551830

Park J, Kang S (2026) PaAno: Patch-based representation learning for time-series anomaly detection. In: International Conference on Learning Representations, URL https://openreview.net/forum?id=NXThkM7Iym

Rippel O, Mertens P, Merhof D (2021) Modeling the distribution of normal data in pre-trained deep features for anomaly detection. In: 2020 25th International Conference on Pattern Recognition. IEEE, pp 6726–6733, https://doi.org/10.1109/ICPR48806.2021.9412109

Roth K, Pemula L, Zepeda J, et al (2022) Towards total recall in industrial anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 14318–14328

Ruff L, Kauffmann JR, Vandermeulen RA, et al (2021) A unifying review of deep and shallow anomaly detection. Proceedings of the IEEE 109(5):756–795. https://doi.org/10.1109/JPROC.2021.3052449

Saheed YK, Abdulganiyu OH, Ait Tchakoucht T (2023) A novel hybrid ensemble learning for anomaly detection in industrial sensor networks and SCADA systems for smart city infrastructures. Journal of King Saud University - Computer and Information Sciences 35(5):101532. https://doi.org/10.1016/j.jksuci.2023.03.010

Sakurada M, Yairi T (2014) Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis. Association for Computing Machinery, New York, NY, USA, pp 4–11, https://doi.org/10.1145/2689746.2689747

Sarfraz MS, Chen MY, Layer L, et al (2024) Position: Quo vadis, unsupervised time series anomaly detection? In: Proceedings of the 41st International Conference on Machine Learning, Proceedings of Machine Learning Research, vol 235. PMLR, Vienna, Austria, pp 43461–43476

Schmidl S, Wenig P, Papenbrock T (2022) Anomaly detection in time series: A comprehensive evaluation. Proceedings of the VLDB Endowment 15(9):1779–1797. https://doi.org/10.14778/3538598.3538602

Shentu Q, Li B, Zhao K, et al (2025) Towards a general time series anomaly detector with adaptive bottlenecks and dual adversarial decoders. In: International Conference on Learning Representations, URL https://openreview.net/forum?id=aKcd7ImG5e

Su Y, Zhao Y, Niu C, et al (2019) 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. Association for Computing Machinery, New York, NY, USA, pp 2828–2837, https://doi.org/10.1145/3292500.3330672

Tatbul N, Lee TJ, Zdonik S, et al (2018) Precision and recall for time series. In: Advances in Neural Information Processing Systems

Tax DMJ, Duin RPW (2004) Support vector data description. Machine Learning 54(1):45–66. https://doi.org/10.1023/B:MACH.0000008084.60811.49

Tuli S, Casale G, Jennings NR (2022) TranAD: Deep transformer networks for anomaly detection in multivariate time series data. Proceedings of the VLDB Endowment 15(6):1201–1214. https://doi.org/10.14778/3514061.3514067

Wang W, Arora R, Livescu K, et al (2015) On deep multi-view representation learning. In: Bach F, Blei D (eds) Proceedings of the 32nd International Conference on Machine Learning, Proceedings of Machine Learning Research, vol 37. PMLR, Lille, France, pp 1083–1092, URL https://proceedings.mlr.press/v37/wangb15.html

Wang X, Lin J, Patel NS, et al (2018) Exact variable-length anomaly detection algorithm for univariate and multivariate time series. Data Mining and Knowledge Discovery 32(6):1806–1844. https://doi.org/10.1007/s10618-018-0569-7

Wu R, Keogh E (2023) Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. IEEE Transactions on Knowledge and Data Engineering 35(3):2421–2429. https://doi.org/10.1109/TKDE.2021.3112126

Wu X, Qiu X, Li Z, et al (2025) CATCH: Channel-aware multivariate time series anomaly detection via frequency patching. In: International Conference on Learning Representations

Xu J, Wu H, Wang J, et al (2022) Anomaly transformer: Time series anomaly detection with association discrepancy. In: International Conference on Learning Representations, URL https://openreview.net/forum?id=LzQQ89U1qm_

Yang Y, Zhang C, Zhou T, et al (2023) DCdetector: Dual attention contrastive representation learning for time series anomaly detection. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. Association for Computing Machinery, New York, NY, USA, pp 3033–3045, https://doi.org/10.1145/3580305.3599295

Yeh CCM, Zhu Y, Ulanova L, et al (2016) Matrix profile i: All pairs similarity joins for time series: A unifying view that includes motifs, discords and shapelets. In: 2016 IEEE 16th International Conference on Data Mining. IEEE, pp 1317–1322, https://doi.org/10.1109/ICDM.2016.0179

Zamanzadeh Darban Z, Webb GI, Pan S, et al (2024) Deep learning for time series anomaly detection: A survey. ACM Computing Surveys 57(1):15:1–15:42. https://doi.org/10.1145/3691338

Zeng A, Chen M, Zhang L, et al (2023) Are transformers effective for time series forecasting? Proceedings of the AAAI Conference on Artificial Intelligence 37(9):11121–11128. https://doi.org/10.1609/aaai.v37i9.26317

Zhang C, Song D, Chen Y, et al (2019) A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. Proceedings of the AAAI Conference on Artificial Intelligence 33(1):1409–1416. https://doi.org/10.1609/aaai.v33i01.33011409

Zhao H, Wang Y, Duan J, et al (2020) Multivariate time-series anomaly detection via graph attention network. In: 2020 IEEE International Conference on Data Mining.

IEEE, pp 841–850, https://doi.org/10.1109/ICDM50108.2020.00093

Zhou Q, Pei C, Sun F, et al (2025) KAN-AD: Time series anomaly detection with kolmogorov–arnold networks. In: Proceedings of the 42nd International Conference on Machine Learning, Proceedings of Machine Learning Research, vol 267. PMLR, Vancouver, Canada, pp 79136–79149, URL https://proceedings.mlr.press/v267/zhou25u.html

Zong B, Song Q, Min MR, et al (2018) Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: International Conference on Learning Representations