BEYOND FORECASTING: MEMORY-AUGMENTED TIME SERIES ANOMALY PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series anomaly prediction (TSAP) aims to anticipate future anomalous states from historical observations for early warning and proactive decision-making. Existing point-level methods typically infer anomaly states from forecasted futures, yet forecasting favors the most likely, normal-dominated evolution, while anomaly prediction depends on preserving rare but discriminative abnormal dynamics. This mismatch often leads to over-smoothed futures and weakened point-level anomaly signals. To bridge this gap, we propose **LAMP**, a lightweight memory-augmented framework that complements parametric prediction with non-parametric evidence from previously observed future evolutions. Given a historical window, LAMP retrieves futures associated with similar historical patterns from a multi-scale memory and adaptively integrates them with parametric predictions, allowing TSAP to exploit not only what is likely to happen, but also what actually happened after similar histories. Experiments on five multivariate time-series benchmarks against 16 baselines show that LAMP consistently improves anomaly prediction, achieving average gains of 6.71% in V-PR and 9.37% in V-ROC, while training up to 8.2× faster with 14.0× lower GPU memory usage than forecasting-based alternatives. Code is available at `https://anonymous.4open.science/r/lamp/`.

1 INTRODUCTION

Time series anomaly prediction (TSAP) is a critical task that aims to infer future abnormal states from observed temporal signals, enabling early warning and proactive decision-making before anomalies occur (Jhin et al., 2023). Unlike time series anomaly detection, which identifies anomalies after they have manifested, TSAP shifts the learning objective toward anticipating impending anomalies from historical observations. This capability is particularly important for safety-sensitive and cost-critical systems, such as water system monitoring (Jhin et al., 2023; Zhao et al., 2025), and server operations (Ang et al., 2023; 2024), where timely prediction helps maintain service reliability, mitigate operational risks, and support effective resource allocation.

Existing TSAP methods can be categorized by prediction granularity into window-level and point-level anomaly prediction. Window-level methods predict whether anomalies will occur within a near-future window (Jhin et al., 2023; Zhao et al., 2024; 2025; Hu et al., 2024), providing an early warning without identifying when the anomaly is likely to occur. In contrast, point-level methods estimate the anomaly state at each future time step, enabling more fine-grained and timely responses. We therefore focus on point-level TSAP. Existing point-level methods typically infer future anomaly states from time-series forecasts, e.g., by comparing forecasts with reference sequences (Chen et al., 2026), exploiting disagreement among forecasting models (Kang et al., 2026), or learning forecasting models with synthetic anomalies (Park et al., 2025).

However, forecasting and point-level anomaly prediction have inherently different objectives. Forecasting tends to capture dominant normal patterns and produce smooth predictions, while point-level anomaly prediction requires preserving rare and abrupt changes that distinguish abnormal steps from normal ones. As illustrated in Figure 1, this mismatch can lead to temporal over-smoothing: abnormal changes in the true future are weakened or blurred in the forecasted sequence. Consequently, anomaly scores derived from these forecasts may respond late and weakly to anomaly onsets and

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.

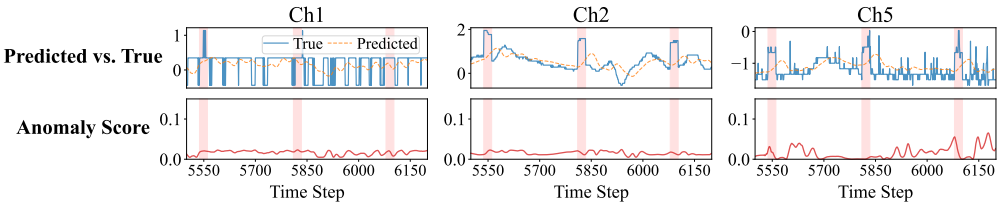

Figure 1: Over-smoothed anomaly prediction on GECCO. Red bands are ground truth anomaly.

fail to clearly distinguish normal and abnormal future steps. This limitation is particularly critical for point-level TSAP, which requires accurate anomaly estimates at individual future time steps.

This observation motivates a different perspective: rather than relying solely on what a parametric forecasting model considers most likely to happen, anomaly prediction can also benefit from what actually happened after similar histories in the past. Specifically, historical patterns similar to the current observation and their corresponding futures can provide complementary information about abnormal dynamics that may be weakened by parametric forecasting. This motivates us to explicitly retrieve and exploit such historical-future pairs for point-level anomaly prediction.

Based on this insight, we propose LAMP, a lightweight memory-augmented framework for point-level time series anomaly prediction. Given the current observation, LAMP retrieves similar historical representations from a multi-scale memory bank and uses their paired future representations to enhance the parametric prediction. Since similar historical patterns can lead to different future outcomes, LAMP adaptively integrates the retrieved future representations with the parametric prediction. In this way, LAMP leverages previously observed abnormal patterns to complement forecast-based prediction, helping preserve informative anomaly signals.

Our contributions can be summarized as follows:

- We identify temporal over-smoothing as a key limitation in forecast-based point-level TSAP: forecasting models trained on normal-dominated data tend to produce smooth future predictions, which may weaken or blur rare and abrupt abnormal changes, leading to delayed and weakened anomaly signals at individual future time steps.
- We propose LAMP, a lightweight memory-augmented framework for point-level TSAP. LAMP retrieves historical representations similar to the current observation from a memory bank, and enhances the parametric prediction with their paired future representations, alleviating the over-smoothing of forecast-based prediction.
- We conduct extensive experiments on five real-world multivariate datasets to evaluate the effectiveness and efficiency of LAMP. Compared with the strongest baseline at each forecast horizon, LAMP improves the average score by 6.71% V-PR and 9.37% V-ROC across six horizons. Meanwhile, LAMP achieves an average training speedup of $8.2\times$ and an average GPU-memory reduction of $14.0\times$ compared with existing TSAP methods.

2 Related Work

2.1 Time Series Anomaly Prediction

TSAP aims to forecast anomalies before they occur, in contrast to anomaly detection which identifies anomalies after the event. Existing studies can be organized according to prediction granularity.

Window-level anomaly prediction estimates whether a future window will contain anomalies. PAD (Jhin et al., 2023) first formalizes precursor-of-anomaly detection for time series, which uses neural controlled differential equations and knowledge distillation. FCM (Zhao et al., 2024) combines future forecasting with reconstruction to model future context, Multirc (Hu et al., 2024) uses multi-scale reconstructive contrastive learning for joint anomaly prediction and detection, and IGCL (Zhao et al., 2025) generates precursor patterns with a diffusion model and stores them in an importance-based memory bank. However, a window-level label reveals only whether an anomaly will occur within the horizon, leaving its timing and the affected steps unspecified.

Point-level anomaly prediction estimates future anomaly states or risks at each time step. A2P (Park et al., 2025) injects learned anomaly-related patterns into forecasting through a prompt pool, thereby steering future prediction toward anomaly-aware states. RED-F (Chen et al., 2026) measures future abnormality by comparing raw forecasts with reconstruction-eliminated forecasts. FATE (Kang et al., 2026) uses disagreement among multiple forecasting models as an uncertainty-based signal. However, their signals are shaped by forecasting models trained on normal-dominated data, so over-smoothing attenuates infrequent anomaly patterns in the resulting point-wise scores.

To address the above limitations, LAMP retrieves historical representations similar to the current observation from a memory bank that stores each representation together with the future that followed it, and fuses the retrieved future with the parametric prediction at the point level, restoring infrequent anomaly patterns that a forecaster alone would smooth away.

2.2 Time Series Anomaly Detection

Existing studies on time series anomaly detection either train specialized models or adapt pre-trained time series foundation models.

Specialized detectors are trained on the target series alone and model normality through density estimation (Breunig et al., 2000) or reconstruction (Su et al., 2019; Xu et al., 2022; Hu et al., 2026). Later architectures strengthen the modeling of temporal dependencies, as in the association discrepancy of Anomaly Transformer (Xu et al., 2022) and the patch-wise contrastive formulation of DCdetector (Yang et al., 2023). ModernTCN (Luo & Wang, 2024) modernizes temporal convolution with large kernels for long-range dependency modeling, and KAN-AD (Zhou et al., 2025) reformulates temporal modeling via Kolmogorov-Arnold networks. However, all of them identify anomalies only after the anomalies have occurred, which leaves no lead time for response.

Time series foundation models are pre-trained on large-scale time series data and support diverse tasks within a single model, including forecasting and anomaly detection (Goswami et al., 2024; Gao et al., 2024; Chen et al., 2025). Since one model already covers both tasks, composing its forecasting and detection outputs appears to suffice for anomaly prediction. However, the huge parameter count makes fine-tuning and inference costly, which limits deployment in scenarios that require timely and frequent predictions.

To address the above limitations, LAMP forecasts point-level anomaly states. Its lightweight architecture reduces training and inference overhead, ensuring efficiency for real-time applications.

3 Method

3.1 Problem Formalization

Consider a multivariate series $\mathcal{X} = \{x_t\}_{t=1}^{T}$ of length T, where each observation $x_t \in \mathbb{R}^{1\times N}$ contains N channels at step t. At time t, only the history window $\mathbf{X}_t^{(h)} = [x_{t-b+1}; \ldots; x_t] \in \mathbb{R}^{b\times N}$ of length b is observed. The objective of TSAP is to learn a mapping F that forecasts the binary label sequence of the future f steps:

$$\hat{Y}_t = \{\hat{y}_{t+1}, \ldots, \hat{y}_{t+f}\} = F\big(\mathbf{X}_t^{(h)}\big) \in \{0,1\}^f, \tag{1}$$

where $\hat{y}_i = 1$ indicates an anomaly and 0 otherwise. The future window $\mathbf{X}_t^{(f)} = [x_{t+1}; \ldots; x_{t+f}] \in \mathbb{R}^{f\times N}$ is available only during training, where it serves as memory values and prediction targets. Table 5 summarizes the notation used throughout.

3.2 Overall Framework

As shown in Fig. 2, LAMP mitigates the over-smoothing of parametric prediction by retrieving previously observed future evolutions associated with historical patterns similar to the current input, and adaptively integrating them with the parametric prediction. Specifically, LAMP consists of three components: (1) a *residual representation module* that constructs paired historical and future representations for memory storage; (2) a *multi-scale contrastive memory module* that retrieves future representations associated with similar historical patterns across temporal scales; and (3) an

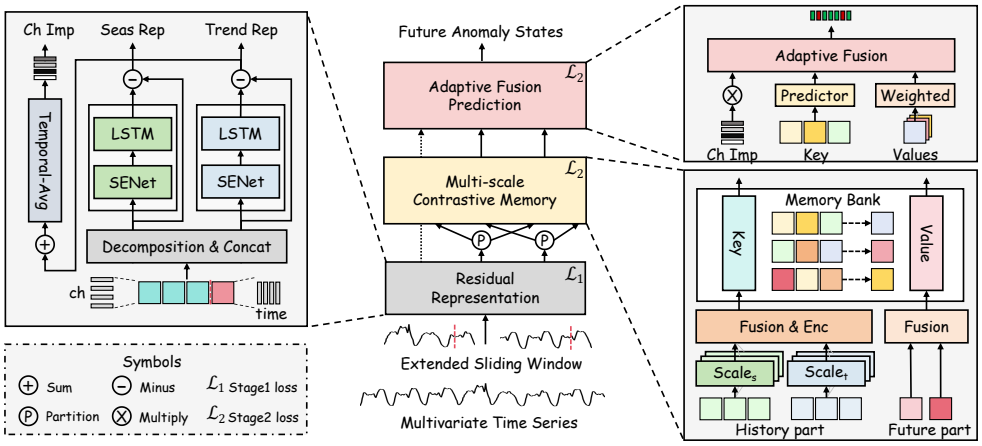

Figure 2: The architecture of LAMP. The residual representation module decomposes the extended window into seasonal and trend components and reconstructs them into paired historical and future residuals. The multi-scale contrastive memory stores the paired residuals as keys and values. The adaptive fusion prediction module fuses the retrieved futures with the parametric prediction through a gate and calibrates the scores with channel-importance weights.

adaptive fusion prediction module that combines retrieved and parametric evidence for point-level anomaly prediction.

3.3 Residual Representation

Temporal decomposition. Following moving-average decomposition (Wu et al., 2021; Wang et al., 2023), we extract the trend component of the history window via $\mathrm{AvgPool}$, and take the remainder as the seasonal component, $\mathbf{X}_t^{(h,\mathrm{trend})} = \mathrm{AvgPool}(\mathbf{X}_t^{(h)}), \mathbf{X}_t^{(h,\mathrm{seas})} = \mathbf{X}_t^{(h)} - \mathbf{X}_t^{(h,\mathrm{trend})}$. The trend component captures gradual changes and the seasonal component short-term deviations.

Dual-branch reconstruction. During training, we concatenate the historical component with the raw future window within each branch $c \in \{\mathrm{seas}, \mathrm{trend}\}$, $\mathbf{Z}_t^{(c)} = [\mathbf{X}_t^{(h,c)}; \mathbf{X}_t^{(f)}] \in \mathbb{R}^{(b+f)\times N}$. Each branch uses an LSTM-based reconstruction network $f_{\mathrm{rec}}^{(c)}$ with unidirectional temporal processing to reconstruct the entire sequence, $\hat{\mathbf{Z}}_t^{(c)} = f_{\mathrm{rec}}^{(c)}(\mathbf{Z}_t^{(c)})$. We split the reconstruction as $\hat{\mathbf{Z}}_t^{(c)} = [\hat{\mathbf{X}}_t^{(h,c)}; \hat{\mathbf{X}}_t^{(f,c)}]$ and compute the two residuals $\mathbf{R}_t^{(h,c)} = \left|\mathbf{X}_t^{(h,c)} - \hat{\mathbf{X}}_t^{(h,c)}\right|$ and $\mathbf{R}_t^{(f,c)} = \left|\mathbf{X}_t^{(f)} - \hat{\mathbf{X}}_t^{(f,c)}\right|$. For each component c, $\mathbf{R}_t^{(h,c)}$ and $\mathbf{R}_t^{(f,c)}$ are the historical and future representations, and their pairing connects the observation window with the future that follows it. Because the reconstruction branch processes time steps causally, the historical prefix of the training reconstruction is unchanged when the future part is omitted, $\hat{\mathbf{X}}_t^{(h,c)} = f_{\mathrm{rec}}^{(c)}(\mathbf{X}_t^{(h,c)})$. At inference, the branch therefore reconstructs solely from the historical component.

3.4 Multi-scale Contrastive Memory

Multi-scale construction and fusion. Let $\mathcal{S} \subseteq \{1, \ldots, b\}$ be the set of temporal scales. At scale $s \in \mathcal{S}$, Seg_s preserves the first s steps of $\mathbf{X}_t^{(h,c)}$ and zero-masks the remaining $b - s$ steps (Keogh et al., 2004; Lee et al., 2026). Reconstructing each view yields

$$\mathbf{R}_t^{(s,c)} = \left|\mathrm{Seg}_s(\mathbf{X}_t^{(h,c)}) - f_{\mathrm{rec}}^{(c)}\big(\mathrm{Seg}_s(\mathbf{X}_t^{(h,c)})\big)\right|. \tag{2}$$

We then aggregate the branch representations by element-wise summation across scales and between seasonal and trend branches: $\mathbf{R}_{\mathrm{ms},t}^{(h,c)} = \sum_{s\in\mathcal{S}} \mathbf{R}_t^{(s,c)}, \mathbf{R}_{\mathrm{ms},t}^{(h)} = \sum_c \mathbf{R}_{\mathrm{ms},t}^{(h,c)}, \mathbf{R}_t^{(f)} = \sum_c \mathbf{R}_t^{(f,c)}$.

The resulting multi-scale historical representation $\mathbf{R}_{\mathrm{ms},t}^{(h)} \in \mathbb{R}^{b\times N}$ is paired with the future representation $\mathbf{R}_t^{(f)} \in \mathbb{R}^{f\times N}$ constructed during training.

Memory bank and importance-based update. Let $\mathcal{M}$ be a bank of capacity M, with $j \in \{1, \ldots, M\}$ indexing retained training windows:

$$\mathcal{M} = \{(\mathbf{r}_j, \mathbf{v}_j)\}_{j=1}^{M}, \qquad \mathbf{r}_j = \mathbf{R}_{\mathrm{ms},j}^{(h)}, \quad \mathbf{v}_j = \mathbf{R}_j^{(f)}. \tag{3}$$

With limited capacity, storing every training pair can fill the bank with nearly flat future values, leaving fewer examples of pronounced deviations. We therefore rank entries by the mean absolute magnitude of their future values, using it as an importance score ρ_j. Here $\mathbf{v}_{j,\ell,n}$ denotes the value at step ℓ and channel n:

$$\rho_j = \frac{1}{fN} \sum_{\ell=1}^{f} \sum_{n=1}^{N} |\mathbf{v}_{j,\ell,n}|. \tag{4}$$

When capacity is exceeded, we retain the top-M entries by ρ_j. This keeps stronger future continuations available when similar patterns recur. Magnitude is only a proxy and may reflect noise.

Contrastive organization and retrieval. Let f_{enc} denote an MLP-based encoder. At each retrieval, it computes $\mathbf{q}_t = f_{\mathrm{enc}}(\mathbf{R}_{\mathrm{ms},t}^{(h)})$ and all memory key embeddings $\mathbf{k}_j = f_{\mathrm{enc}}(\mathbf{r}_j)$. Let sim denote cosine similarity and K_{neg} the number of negatives. For each query i, select the nearest key as $\mathbf{k}_i^+$ and the K_{neg} least similar keys as $\{\mathbf{k}_{i,k}^-\}_{k=1}^{K_{\mathrm{neg}}}$. The contrastive loss with temperature $\tau > 0$ is

$$\mathcal{L}_{\mathrm{cl}} = -\log \frac{\exp(\mathrm{sim}(\mathbf{q}_i, \mathbf{k}_i^+)/\tau)}{\exp(\mathrm{sim}(\mathbf{q}_i, \mathbf{k}_i^+)/\tau) + \sum_{k=1}^{K_{\mathrm{neg}}} \exp(\mathrm{sim}(\mathbf{q}_i, \mathbf{k}_{i,k}^-)/\tau)}, \tag{5}$$

This objective separates selected similar and dissimilar keys.

3.5 Adaptive Fusion Prediction

Gated Fusion. An LSTM-based predictor f_{pred} learns a mapping from historical to future representations. Given $\mathbf{R}_{\mathrm{ms},t}^{(h)}$, it produces $\hat{\mathbf{R}}_t^{(\mathrm{base})} = f_{\mathrm{pred}}(\mathbf{R}_{\mathrm{ms},t}^{(h)}) \in \mathbb{R}^{f \times N}$. For memory retrieval, the encoder computes $\mathbf{q}_t = f_{\mathrm{enc}}(\mathbf{R}_{\mathrm{ms},t}^{(h)})$ and $\mathbf{k}_j = f_{\mathrm{enc}}(\mathbf{r}_j)$ for each stored entry. Let K be the retrieval count and $\mathcal{N}_K(t)$ the indices of the K keys with highest cosine similarity to $\mathbf{q}_t$. Applying softmax over only these K scores gives weights $\omega_{t,j}$. Their paired future values form

$$\hat{\mathbf{R}}_t^{(\mathrm{mem})} = \sum_{j \in \mathcal{N}_K(t)} \omega_{t,j} \mathbf{v}_j \in \mathbb{R}^{f \times N}. \tag{6}$$

This retrieved representation provides instance-specific information for fusion. The gate input $\mathbf{g}_t$ is the concatenation of the flattened base and memory predictions. An MLP with sigmoid output maps $\mathbf{g}_t$ to element-wise gate weights $\boldsymbol{\alpha}_t \in (0,1)^{f \times N}$. Let $\odot$ denote element-wise multiplication and $\mathbf{1} \in \mathbb{R}^{f \times N}$ the all-ones matrix. The fused prediction is $\hat{\mathbf{R}}_t^{(\mathrm{final})} = \boldsymbol{\alpha}_t \odot \hat{\mathbf{R}}_t^{(\mathrm{mem})} + (\mathbf{1} - \boldsymbol{\alpha}_t) \odot \hat{\mathbf{R}}_t^{(\mathrm{base})}$.

Channel-aware score calibration. We approximate channel importance from reconstruction residuals, borrowing from reconstruction-based detection methods (Xu et al., 2022; Shentu et al., 2025; Hu et al., 2026) the idea that reconstruction errors characterize anomaly strength. For each channel, we sum seasonal and trend historical residuals and average over training windows and time steps to obtain $\mathbf{w}_{\mathrm{ch}} \in \mathbb{R}^N$. We present the pseudo-code in Algorithm 1. The weights are broadcast across future steps and applied after fusion, $\tilde{\mathbf{R}}_t^{(\mathrm{final})} = \hat{\mathbf{R}}_t^{(\mathrm{final})} \odot \mathbf{w}_{\mathrm{ch}}$. Finally, the resulting anomaly scores $\tilde{\mathbf{R}}_t^{(\mathrm{final})}$ are thresholded to produce $\hat{Y}_t$.

3.6 Model Training

The residual representations serve as memory keys, memory values, and prediction targets, so LAMP adopts two-stage training, first training the reconstruction branches to keep them stable and then optimizing the remaining modules. In stage one, we train each reconstruction branch on the concatenated components $\mathbf{Z}_t^{(c)}$:

$$\mathcal{L}_1 = \mathcal{L}_{\mathrm{rec}} = \frac{1}{2(b+f)N} \sum_{c \in \{\mathrm{seas}, \mathrm{trend}\}} \left\| \mathbf{Z}_t^{(c)} - \hat{\mathbf{Z}}_t^{(c)} \right\|_2^2. \tag{7}$$

We then freeze both branches to construct memory entries and prediction targets. In stage two, we optimize the encoder, base predictor, and gate against the future representation:

$$\mathcal{L}_{\text{pred}} = \frac{1}{fN}\left\|\hat{\mathbf{R}}_t^{(\text{final})} - \mathbf{R}_t^{(f)}\right\|_2^2, \qquad \mathcal{L}_2 = \mathcal{L}_{\text{pred}} + \mathcal{L}_{\text{cl}}. \tag{8}$$

4 Experiments

4.1 Experimental Setup

Datasets We evaluate our method on five real-world multivariate time series datasets: GECCO (Lai et al., 2021), SMAP (Hundman et al., 2018), PSM (Abdulaal et al., 2021), NYC (Cui et al., 2016), and SMD (Su et al., 2019). These datasets cover diverse domains including water treatment systems, space sensor data, transportation data, and server machine metrics. More details are in Table 1.

Table 1: Details of datasets. "#Ch" denotes the number of variables/channels. "Train", "Valid" and "Test" denote the number of time points in the training, validation and testing sets, respectively. "AR" denotes the anomaly rate.

Dataset	#Ch	Train	Valid	Test	AR (%)
GECCO	9	55,408	13,852	69,261	1.25
SMAP	25	108,146	27,037	427,617	9.72
PSM	25	105,984	26,497	87,841	11.07
NYC	3	10,483	2,621	4,416	0.57
SMD	38	566,724	141,681	708,420	2.08

Baselines We compare our method against two categories of baselines. The first category contains point-level methods. As only a limited number of existing methods directly address point-level TSAP, we adopt A2P (Park et al., 2025) as a baseline. To obtain a broader and more comparable evaluation, we therefore follow A2P (Park et al., 2025) to construct Forecast-then-Detect baselines, which first predict future time series values using a forecasting model and then identify anomalies from the predicted results using an anomaly detection model. We use Autoformer (Auto) (Wu et al., 2021), PatchTST (Patch) (Nie et al., 2023), DLinear (DLin) (Zeng et al., 2023), and Duet (Qiu et al., 2025) as forecasters, and combine them with DCdetector (DC) (Yang et al., 2023), ModernTCN (Mod) (Luo & Wang, 2024), and KAN-AD (KAN) (Zhou et al., 2025) as detectors. We further include MOMENT (MOM) (Goswami et al., 2024), a time series foundation model that supports both forecasting and anomaly detection, as a Forecast-then-Detect baseline.

The second category contains window-level methods, including FCM (Zhao et al., 2024) and IGCL (Zhao et al., 2025). Although these methods ultimately output a window-level anomaly score, their implementations also produce anomaly scores for individual future time points before the final window-level aggregation. Following their implementations, we evaluate these time point scores for point-wise evaluation. Further details concerning baselines are available in Appendix B.1.

Metrics Recent studies on time series anomaly detection evaluation have shown that protocols relying on point adjustment or threshold tuning can misleadingly inflate performance scores (Paparrizos et al., 2022; Liu & Paparrizos, 2024; Sarfraz et al., 2024). These evaluation issues also affect anomaly prediction. Following existing works (Paparrizos et al., 2022; Liu & Paparrizos, 2024; Boniol et al., 2025), we adopt the Volume Under the Precision-Recall Surface (V-PR) and Volume Under the Receiver Operating Characteristic Surface (V-ROC) as our primary metrics. Further discussion and evaluation results under other metrics are provided in Appendix C.1.

Implementation Details LAMP is implemented in PyTorch and trained on an RTX 4090 GPU (48 GB). Adam is used for optimization, with an initial learning rate of 0.01, and the batch size is 256. The look-back window size b is fixed at 192 for all models. The moving-average decomposition uses a kernel size of 25. Each reconstruction network $f_{\text{rec}}^{(c)}$ consists of a SENet module followed by a three-layer LSTM with a hidden dimension of 128. The predictor f_{pred} is also a three-layer LSTM with a hidden dimension of 128. The output embedding dimension of f_{enc} is set to 32. More implementation details are provided in Appendix B.2.

Table 2: Short-term anomaly prediction results on five datasets. V-PR (%) and V-ROC (%) are reported under forecast horizons of 32, 64, and 96. The best and second-best results are highlighted in red bold and blue underlined text, respectively.

Method		GECCO						SMAP						PSM					
		32		64		96		32		64		96		32		64		96	
		V-PR	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*
Auto	DC	1.63	58.54	1.55	58.13	1.59	58.73	14.91	52.47	14.72	52.98	14.72	53.29	27.35	46.62	27.96	47.98	27.59	48.33
	Mod	7.16	77.62	6.14	75.78	4.89	71.67	13.44	50.54	15.01	51.66	15.55	51.66	28.84	48.31	28.45	47.47	32.25	50.47
	KAN	2.97	45.70	1.96	23.39	1.20	22.31	13.92	46.89	13.46	45.62	12.79	47.03	34.67	45.05	36.29	46.94	32.64	43.02
Patch	DC	1.54	52.51	1.49	55.68	1.66	57.27	14.09	52.82	14.08	53.36	14.20	53.28	27.66	48.12	27.53	45.63	27.77	47.52
	Mod	20.69	91.07	11.50	89.26	10.50	87.91	17.10	45.92	17.35	45.83	17.64	46.24	35.71	54.03	34.66	53.66	34.13	52.67
	KAN	2.77	39.05	3.03	20.10	2.79	16.64	12.83	42.30	14.12	47.96	11.33	38.67	35.20	48.38	33.71	45.14	33.69	43.47
DLin	DC	1.48	52.53	1.56	57.43	1.56	56.09	13.88	52.44	13.92	52.95	13.89	52.87	33.55	48.13	35.32	47.75	31.53	48.37
	Mod	12.74	**91.24**	7.25	79.56	5.54	72.96	17.11	46.15	17.20	46.23	17.51	46.40	34.87	53.60	34.27	54.05	33.55	51.91
	KAN	1.30	21.37	1.35	15.06	2.08	18.19	13.15	40.12	13.66	45.02	11.99	39.25	33.55	47.28	35.32	48.92	31.53	39.45
Duet	DC	1.66	58.68	1.66	57.95	1.64	57.75	13.88	52.63	14.09	53.21	14.20	53.07	27.83	48.06	27.45	43.01	28.16	45.68
	Mod	20.82	90.12	10.36	83.29	8.17	78.79	15.71	45.89	15.94	45.69	16.21	45.82	35.21	54.22	33.90	51.74	32.81	50.49
	KAN	1.39	18.87	1.40	18.35	1.93	21.93	15.23	49.23	18.17	55.56	13.29	44.10	35.54	50.97	35.94	50.36	35.57	47.05
MOM	MOM	38.46	83.66	33.03	88.06	34.69	**90.16**	13.14	43.30	14.06	43.82	15.20	44.76	47.04	64.59	49.73	**67.99**	50.96	66.96
	A2P	2.27	42.59	1.47	41.53	1.67	52.90	13.19	45.52	13.31	46.64	12.88	43.03	32.85	46.69	33.35	44.94	33.14	45.52
	FCM	1.69	49.72	2.64	50.07	2.40	49.93	14.28	50.23	13.78	50.15	14.46	50.18	29.26	50.02	28.67	50.02	29.50	50.05
	IGCL	1.48	24.64	1.39	20.71	1.39	21.44	12.87	43.36	13.41	43.22	12.08	35.12	35.36	44.62	35.39	44.23	36.46	46.42
LAMP (Ours)		**50.83**	89.48	**50.82**	**90.30**	**51.49**	89.59	**19.61**	**62.49**	**18.87**	**62.27**	**20.76**	**63.09**	**53.95**	**70.17**	**53.83**	67.97	**57.71**	**69.32**

Method		NYC						SMD						AVERAGE					
		32		64		96		32		64		96		32		64		96	
		V-PR	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*
Auto	DC	4.07	54.02	3.76	53.82	4.00	52.99	4.51	52.08	4.48	52.20	4.49	50.28	10.49	52.75	10.49	53.02	10.48	52.72
	Mod	3.82	58.99	4.94	63.41	4.64	62.29	5.12	52.00	5.81	54.52	5.01	50.27	11.68	57.49	12.07	58.57	12.47	57.27
	KAN	4.49	62.40	5.60	67.80	6.44	67.32	5.73	48.20	5.35	45.76	4.59	41.54	12.36	49.65	12.53	45.90	11.53	44.24
Patch	DC	3.69	47.97	4.14	53.22	4.70	59.90	4.59	52.01	4.64	51.75	4.61	51.18	10.31	50.69	10.38	51.93	10.59	53.83
	Mod	6.60	67.92	5.22	66.78	10.51	72.69	9.05	65.78	7.61	59.74	7.13	58.48	17.83	64.94	15.27	63.05	15.98	63.60
	KAN	4.69	64.08	4.14	57.51	4.38	60.20	5.57	48.17	5.69	48.22	5.49	47.71	12.21	48.40	12.14	43.79	11.54	41.34
DLin	DC	4.27	53.72	4.61	55.39	5.01	57.54	4.64	51.81	4.79	52.50	4.88	52.71	11.56	51.73	12.04	53.20	11.37	53.52
	Mod	4.15	61.44	4.11	60.20	3.58	56.84	10.15	**70.04**	8.91	63.76	8.15	62.19	15.80	64.49	14.35	60.76	13.67	58.06
	KAN	4.44	61.55	4.26	57.69	4.98	62.67	7.08	54.39	6.75	53.19	6.28	50.23	11.90	44.94	12.27	43.98	11.37	41.96
Duet	DC	3.67	50.49	4.17	53.00	4.70	57.41	4.65	51.38	4.66	51.51	4.85	52.33	10.34	52.25	10.41	51.74	10.71	53.25
	Mod	10.81	71.76	12.97	70.77	11.97	70.83	8.14	64.95	7.33	62.01	6.77	60.49	18.14	65.39	16.10	62.70	15.19	61.28
	KAN	5.03	67.63	4.63	62.43	4.81	59.41	6.62	52.09	3.69	44.85	5.65	47.07	12.76	47.76	12.77	46.31	12.25	43.91
MOM	MOM	8.58	66.55	11.26	72.04	13.86	71.89	8.53	56.24	10.96	56.60	13.44	56.40	23.15	62.87	23.81	65.70	25.63	66.03
	A2P	8.36	65.17	9.27	64.01	8.86	65.38	6.97	51.80	6.42	50.74	6.07	48.71	12.73	50.35	12.76	49.57	12.52	51.11
	FCM	5.51	51.07	4.58	50.10	6.75	50.90	5.06	49.59	5.10	49.63	5.44	49.66	11.16	50.13	10.95	49.99	11.71	50.14
	IGCL	8.76	69.04	12.20	75.67	14.29	79.67	6.52	45.61	6.61	46.44	6.53	45.47	13.00	45.45	13.80	46.05	14.15	45.62
LAMP (Ours)		**14.13**	**77.88**	**13.29**	**80.43**	**16.17**	**83.58**	**12.27**	67.64	**12.90**	**65.46**	**13.97**	**66.55**	**30.16**	**73.53**	**29.94**	**73.29**	**32.02**	**74.43**

* LAMP ranks first in 15/15 V-PR and 11/15 V-ROC.

4.2 Results and Discussion

Main Results Tables 2 and 3 summarize performance across short-term horizons (32, 64, and 96) and long-term horizons (192, 384, and 576). LAMP achieves the best average V-PR and V-ROC at all six horizons. For short-term prediction, it improves over the strongest baseline by 7.01%, 6.13%, and 6.39% in V-PR and by 8.14%, 7.59%, and 8.40% in V-ROC, respectively, ranking first in all 15 V-PR settings and 11 of 15 V-ROC settings. For long-term prediction, the corresponding gains are 2.71%, 4.88%, and 13.16% in V-PR and 8.65%, 6.92%, and 16.54% in V-ROC. The increasing advantage at the longest horizon indicates that LAMP remains robust as prediction uncertainty accumulates. Across both settings, baseline performance varies considerably across datasets and horizons, revealing the difficulty of TSAP. Forecast-then-Detect pipelines vary substantially across forecasting and detection model combinations. One possible reason is that prediction errors from the forecasting stage interact differently with the scoring rules of different detectors. The point-level baseline A2P struggles to maintain competitive performance under long horizons, indicating that fine-grained temporal localization of future anomalies remains challenging, especially as prediction uncertainty accumulates over longer forecast windows. We can also observe that MOMENT obtains the strongest overall baseline performance among the two-stage pipelines, showing the benefit of pre-trained time series foundation models.

Ablation Study We conduct ablations on all datasets and report the average V-ROC across short- and long-term horizons in Table 4. Removing the memory bank causes the largest mean degradation of 12.03%, with drops ranging from 8.01 to 14.57% across horizons, confirming the central role of memory retrieval. Removing decomposition and the residual representation decreases V-ROC by 8.07% and 7.83% on average, respectively, showing that both temporal decomposition and residual-

Table 3: Long-term anomaly prediction results on five datasets. V-PR (%) and V-ROC (%) are reported under forecast horizons of 192, 384, and 576. The best and second-best results are highlighted in red bold and blue underlined text, respectively.

Method		GECCO						SMAP						PSM					
		192		384		576		192		384		576		192		384		576	
		V-PR	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*
Auto	DC	1.48	57.62	1.66	55.95	2.02	63.00	14.79	53.24	14.35	53.06	14.59	53.31	27.54	47.47	27.53	47.95	27.73	48.37
	Mod	2.30	61.63	2.70	63.48	2.37	64.07	15.88	53.87	15.44	52.04	16.25	51.62	32.68	51.21	34.40	51.80	26.62	42.49
	KAN	1.20	24.42	1.18	22.30	1.29	28.76	14.33	50.37	13.23	45.22	13.61	44.89	29.72	37.93	33.35	43.43	34.33	50.07
Patch	DC	1.76	59.99	1.33	49.97	1.44	53.36	14.24	52.81	14.11	52.42	14.06	51.95	27.84	47.94	27.85	46.82	27.81	48.78
	Mod	11.89	**91.46**	4.85	78.19	3.75	71.55	17.82	45.24	18.13	45.01	17.83	43.95	33.43	52.56	30.38	45.96	28.76	47.05
	KAN	1.57	30.14	1.61	40.31	1.19	20.52	14.68	47.25	14.28	48.70	11.03	36.34	35.98	48.05	32.87	41.66	29.69	36.62
DLin	DC	1.62	58.27	1.56	57.93	1.41	54.39	14.02	52.69	14.12	52.69	14.01	52.81	27.66	44.91	27.95	45.61	28.21	43.99
	Mod	6.35	82.63	3.16	68.91	2.90	67.41	17.68	45.28	17.70	44.61	17.72	44.04	34.57	53.87	30.80	46.98	28.06	42.42
	KAN	1.47	18.01	1.22	19.49	1.08	13.14	14.56	48.06	17.16	51.58	**21.81**	62.43	33.76	43.68	32.83	41.35	34.10	49.48
Duet	DC	1.51	56.05	1.83	59.81	1.48	52.18	14.75	52.87	14.02	52.59	14.11	52.17	27.64	46.84	28.06	42.96	27.92	46.41
	Mod	6.13	72.92	5.04	64.34	4.89	67.16	16.73	44.86	17.04	43.72	17.29	43.36	32.43	50.78	32.60	49.51	30.67	48.08
	KAN	1.16	18.23	1.07	13.10	1.13	16.49	14.67	47.44	14.30	46.33	15.93	54.32	31.08	38.58	28.94	34.35	32.97	42.07
MOM	MOM	39.67	78.87	15.39	82.70	-*	-*	18.59	47.30	19.78	50.16	-*	-*	45.37	64.20	46.53	65.25	-*	-*
A2P		1.90	56.33	1.96	49.53	1.91	52.56	12.84	42.54	13.50	41.98	13.23	45.45	33.19	46.15	32.91	45.71	32.50	44.24
FCM		3.06	50.00	2.86	49.91	3.20	50.03	15.31	50.23	16.20	50.51	16.11	50.52	31.70	50.07	30.67	50.15	30.94	50.18
IGCL		1.41	21.02	1.45	21.71	1.50	22.88	12.99	36.54	13.13	37.58	13.15	36.08	36.11	36.01	36.78	45.80	39.78	49.88
LAMP (Ours)		**40.47**	89.45	**41.84**	**89.61**	**38.96**	**87.33**	**22.37**	**60.21**	**20.03**	**62.04**	20.59	**64.67**	**56.21**	**67.43**	**53.88**	**65.79**	**54.81**	**71.48**

Method		NYC						SMD						AVERAGE					
		192		384		576		192		384		576		192		384		576	
		V-PR	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*	*V-PR*	*V-ROC*
Auto	DC	3.91	53.70	3.92	51.84	4.29	57.10	4.42	50.70	4.24	51.00	4.42	51.80	10.43	52.55	10.34	51.96	10.61	54.72
	Mod	3.51	55.24	3.69	58.24	4.13	62.78	4.85	48.51	4.67	48.43	4.58	48.56	11.84	54.09	12.18	54.80	10.79	53.90
	KAN	5.15	68.38	4.69	64.50	4.17	58.57	4.14	38.94	4.49	39.39	4.36	38.07	10.91	44.01	11.39	42.97	11.55	44.07
Patch	DC	3.86	50.62	3.50	48.35	3.52	48.23	4.83	52.04	5.02	52.10	4.85	52.03	10.51	52.68	10.36	49.93	10.34	50.87
	Mod	2.60	37.24	4.58	67.31	4.72	68.04	6.59	57.00	5.89	53.49	5.54	51.99	14.47	56.70	12.77	57.99	12.12	56.52
	KAN	4.27	60.93	4.40	62.12	4.09	58.86	3.81	31.71	5.30	44.89	4.46	37.19	12.06	43.62	11.69	47.54	10.09	37.91
DLin	DC	4.92	57.97	4.53	54.65	4.04	52.81	5.21	53.08	5.06	53.00	5.17	52.87	10.69	53.38	10.64	52.78	10.57	51.37
	Mod	3.88	59.04	4.75	67.26	4.18	63.49	7.19	58.82	6.56	57.39	6.09	55.49	13.93	59.93	12.59	57.03	11.79	54.57
	KAN	4.79	59.48	4.99	61.08	4.50	61.46	5.92	48.97	5.22	45.22	5.32	44.76	12.10	43.64	12.28	43.74	13.36	46.25
Duet	DC	3.94	51.75	4.38	55.60	4.18	53.28	4.92	52.86	5.13	52.99	4.85	52.72	10.44	52.07	10.68	52.79	10.51	51.35
	Mod	6.13	65.90	6.68	66.42	5.19	63.98	6.49	56.43	5.80	53.84	5.52	50.33	13.58	58.18	13.43	55.57	12.71	54.58
	KAN	4.30	62.33	4.66	62.71	4.28	58.45	5.49	48.59	4.75	40.19	4.55	40.40	11.34	43.03	10.74	39.34	11.77	42.35
MOM	MOM	**17.79**	72.05	**24.85**	76.08	-*	-*	**10.67**	56.61	10.28	57.61	-*	-*	26.42	63.81	23.37	66.36	-*	-*
A2P		3.39	52.90	4.72	57.33	5.13	58.11	5.86	48.03	5.54	46.64	5.41	46.27	11.44	49.19	11.73	48.24	11.64	49.33
FCM		7.61	50.96	8.85	51.20	8.95	51.08	6.56	49.83	6.47	49.69	6.46	49.81	12.85	50.22	13.01	50.29	13.13	50.32
IGCL		17.15	**83.02**	12.28	81.69	**15.94**	**85.61**	6.59	45.54	6.05	44.28	6.12	45.27	14.85	44.43	13.94	46.21	15.30	47.94
LAMP (Ours)		16.03	80.23	13.93	**83.29**	15.83	79.39	10.57	**64.96**	**11.57**	**65.69**	**12.13**	**62.44**	**29.13**	**72.46**	**28.25**	**73.28**	**28.46**	**73.06**

* LAMP ranks first in 10/15 V-PR and 12/15 V-ROC.
* The maximum forecast horizon of the MOMENT (MOM) is 512 time steps.

based representations contribute substantially to prediction. Removing multi-scale modeling results in a 7.65% average drop and is particularly detrimental at horizons 192 and 384, indicating the need to capture precursor patterns over different temporal spans. The contrastive loss and channel importance weights yield average gains of 5.28% and 6.08%, with the contrastive loss becoming especially important at horizon 576. We validate the learned channel importance weights against label-based channel relevance in Appendix C.2.

Sensitivity Analysis We also demonstrate that LAMP is robust to its hyperparameters, as it maintains stable performance across different settings. Figure 3 shows the results obtained by varying the top-K, the memory bank capacity, and the scale factor. When the top-K, the memory bank capacity, or the scale factor changes over its full search range, V-ROC varies by less than 2%, which indicates that the robustness of LAMP.

Table 4: Ablation study of LAMP averaged over all five datasets. We report average V-ROC (%) under short-term and long-term forecast horizons.

Method	Forecast Horizons					
	32	64	96	192	384	576
LAMP (Full model)	**73.53**	**73.29**	**74.43**	**72.46**	**73.28**	**73.06**
w/o Residual Representation	67.76	65.92	63.94	68.17	64.48	62.81
w/o Decomposition	61.48	68.63	64.77	70.19	65.85	60.70
w/o Memory bank	59.26	61.31	60.92	64.45	63.42	58.49
w/o Contrastive loss	72.14	68.07	71.56	69.22	66.07	61.28
w/o Multi-scale	67.17	65.75	69.11	63.77	63.24	65.13
w/o Channel score calibration	69.00	68.36	66.70	64.25	65.62	69.67

Efficiency Analysis We evaluate the computational efficiency of LAMP from two perspectives: training overhead and inference cost. All methods are trained under unified hyperparameters whenever applicable. Experimental details are in Appendix B.4.

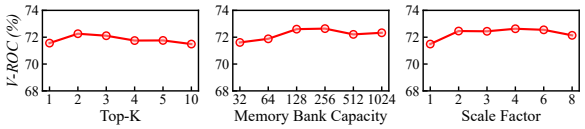

Figure 3: Sensitivity analysis on Top-K (left), memory bank capacity (middle), and scale factor (right). We report V-ROC(%) under forecast horizon of 32, averaged over five datasets.

Figure 4 reports the training overhead. LAMP completes each epoch in 4.8 seconds with 2.0 GB peak GPU memory, ranking among the top three of all compared methods on both metrics. By contrast,

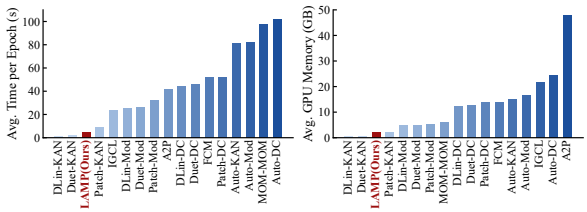

Figure 4: Efficiency analysis on GECCO. We report the training time per epoch and the GPU memory, each averaged over forecast horizons of 32, 96, 384, and 576.

Figure 5 further examines the inference cost by jointly considering performance, model size, and computational complexity. LAMP lies in the upper-left region of the performance–cost plane, achieving the best V-ROC with only 0.4M parameters and 53G FLOPs, while other methods generally require larger parameter budgets or heavier computation for comparable performance, especially MOMENT, whose 40M parameters and inference cost of 3,000T FLOPs are far beyond the plotted scale, and therefore we omit it from the figure. It highlights LAMP practical efficiency for real-time applications.

5 Conclusion

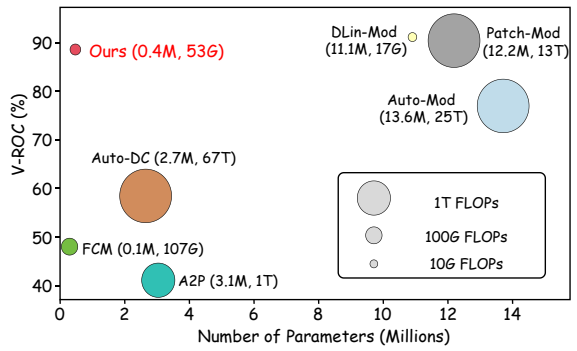

Figure 5: Perf-cost analysis on GECCO under a forecast horizon of 32. The plot shows model performance against the number of parameters and the computational cost.

In this paper, we studied point-level time series anomaly prediction, which estimates the anomaly state of each future time step to support timely and fine-grained responses. We identified temporal over-smoothing as a key obstacle for forecast-based methods, since models trained on normal-dominated data tend to forecast smooth futures, making anomaly scores respond late and weakly to anomaly onsets. To address this problem, we proposed LAMP, a lightweight memory-augmented framework that retrieves historical representations similar to the current observation from a memory bank and enhances the parametric prediction with their paired future representations. These pairs are constructed from reconstruction residuals and organized across temporal scales, and an adaptive gate fuses the retrieved futures with the parametric prediction. Extensive experiments on five multivariate datasets show that LAMP consistently leads under both short-term and long-term horizons while maintaining high efficiency.

Although LAMP performs well on standard benchmarks, its evaluation cannot cover all deployment conditions in real monitoring systems. Sensor faults, irregular sampling, and changing operating regimes may degrade the memory bank and the reliability of retrieved futures, and the unsupervised residual proxy may inflate noise or miss anomalous channels with small reconstruction errors. Future work will address these conditions with more robust memory designs.

AI USE STATEMENT

We used generative AI tools to draft and polish parts of the manuscript, including the method section and the description of experimental protocols and hyperparameter settings in the appendix, and to retrieve publicly available information such as the implementation details of baseline methods. We did not use generative AI tools for any other part of the work: no synthetic datasets were generated, the method and its formulation were developed by the authors, and all experimental results were produced by the authors' own implementation. We have reviewed all AI-assisted work and verified the drafted text against our experiments and the retrieved information against its original sources. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

All datasets are publicly available, and their statistics are summarized in Table 1. The problem formulation and notation are given in Section 3.1 and Table 5, respectively. The baseline and training protocols are described in Appendix B.1, hyperparameter settings for all methods in Appendix B.3, the channel-importance computation in Algorithm 1, and the measurement protocol of the efficiency analysis in Appendix B.4. Code is available at `https://anonymous.4open.science/r/lamp/`.

REFERENCES

Ahmed Abdulaal, Zhuanghua Liu, and Tomer Lancewicki. Practical approach to asynchronous multivariate time series anomaly detection and localization. In *KDD*, pp. 2485–2494, 2021.

Yihao Ang, Qiang Huang, Anthony K. H. Tung, and Zhiyong Huang. A stitch in time saves nine: Enabling early anomaly detection with correlation analysis. In *ICDE*, pp. 1832–1845, 2023.

Yihao Ang, Qiang Huang, Anthony K. H. Tung, and Zhiyong Huang. Eads: An early anomaly detection system for sensor-based multivariate time series. In *ICDE*, pp. 5433–5436, 2024.

Paul Boniol, Ashwin K. Krishna, Marine Bruel, Qinghua Liu, Mingyi Huang, Themis Palpanas, Ruey S. Tsay, Aaron Elmore, Michael J. Franklin, and John Paparrizos. Vus: effective and efficient accuracy measures for time-series anomaly detection: Vus: effective and efficient accuracy measures for time-series anomaly detection... *The VLDB Journal*, 34(3), 2025.

Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: identifying density-based local outliers. *SIGMOD Rec.*, 29(2):93–104, 2000.

PengYu Chen, Xiaohou Shi, Yuan Chang, Yan Sun, and Sajal K. Das. Red-f: Reconstruction-elimination based dual-stream contrastive forecasting for multivariate time series anomaly prediction. *arXiv preprint arXiv:2511.20044*, 2026.

Shengchao Chen, Guodong Long, Jing Jiang, and Chengqi Zhang. Federated foundation models on heterogeneous time series. In *AAAI*, volume 39, pp. 15839–15847, 2025.

Yuwei Cui, Chetan Surpur, Subutai Ahmad, and Jeff Hawkins. A comparative study of htm and other neural network models for online sequence learning with streaming data. In *IJCNN*, pp. 1530–1538, 2016.

Shanghua Gao, Teddy Koker, Owen Queen, Thomas Hartvigsen, Theodoros Tsiligkaridis, and Marinka Zitnik. Units: A unified multi-task time series model. *NeurIPS*, 37:140589–140631, 2024.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. In *ICML*, 2024.

Shiyan Hu, Kai Zhao, Xiangfei Qiu, Yang Shu, Jilin Hu, Bin Yang, and Chenjuan Guo. Multirc: Joint learning for time series anomaly prediction and detection with multi-scale reconstructive contrast. *arXiv preprint arXiv:2410.15997*, 2024.

Shiyan Hu, Jianxin Jin, Yang Shu, Peng Chen, Bin Yang, and Chenjuan Guo. Towards multimodal time series anomaly detection with semantic alignment and condensed interaction. In *ICLR*, 2026.

Alexis Huet, Jose Manuel Navarro, and Dario Rossi. Local evaluation of time series anomaly detection algorithms. In *KDD*, pp. 635–645, 2022.

Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *KDD*, pp. 387–395, 2018.

Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *TOIS*, 20(4):422–446, 2002.

Sheo Yon Jhin, Jaehoon Lee, and Noseong Park. Precursor-of-anomaly detection for irregular time series. In *KDD*, pp. 917–929, 2023. doi: 10.1145/3580305.3599469.

Hyeongwon Kang, Jinwoo Park, Seunghun Han, and Pilsung Kang. Forecasting anomaly precursors via uncertainty-aware time-series ensembles. *arXiv preprint arXiv:2602.17028*, 2026.

Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. Segmenting time series: A survey and novel approach. *Data mining in time series databases*, 57(2004):1–22, 2004.

Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. Towards a rigorous evaluation of time-series anomaly detection. In *AAAI*, volume 36, pp. 7194–7201, 2022.

Kwei-Herng Lai, Daochen Zha, Junjie Xu, Yue Zhao, Guanchu Wang, and Xia Hu. Revisiting time series outlier detection: Definitions and benchmarks. In *NeurIPS Datasets and Benchmarks*, 2021.

Cheonwoo Lee, Dooho Lee, Doyun Choi, and Jaemin Yoo. Generalizing multi-scale time-series modeling with a single operator. *arXiv preprint arXiv:2605.31129*, 2026.

Qinghua Liu and John Paparrizos. The elephant in the room: Towards a reliable time-series anomaly detection benchmark. In *NeurIPS*, volume 37, pp. 108231–108261, 2024.

Donghao Luo and Xue Wang. Moderntcn: A modern pure convolution structure for general time series analysis. In *ICLR*, 2024.

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

John Paparrizos, Paul Boniol, Themis Palpanas, Ruey S. Tsay, Aaron Elmore, and Michael J. Franklin. Volume under the surface: a new accuracy evaluation measure for time-series anomaly detection. *Proc. VLDB Endow.*, 15(11):2774–2787, 2022.

Min-Yeong Park, Won-Jeong Lee, Seong Tae Kim, and Gyeong-Moon Park. When will it fail?: Anomaly to prompt for forecasting future anomalies in time series. In *ICML*, pp. 48086–48103, 2025.

Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. Duet: Dual clustering enhanced multivariate time series forecasting. In *KDD*, pp. 1185–1196, 2025.

M Saquib Sarfraz, Mei-Yen Chen, Lukas Layer, Kunyu Peng, and Marios Koulakis. Position: Quo vadis, unsupervised time series anomaly detection? In *ICML*, pp. 43461–43476, 2024.

Qichao Shentu, Beibu Li, Kai Zhao, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Chenjuan Guo. Towards a general time series anomaly detector with adaptive bottlenecks and dual adversarial decoders. In *ICLR*, 2025.

Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *KDD*, pp. 2828–2837, 2019.

Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. MICN: Multi-scale local and global context modeling for long-term series forecasting. In *ICLR*, 2023.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *NeurIPS*, 34:22419–22430, 2021.

Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly transformer: Time series anomaly detection with association discrepancy. In *ICLR*, 2022.

Yiyuan Yang, Chaoli Zhang, Tian Zhou, Qingsong Wen, and Liang Sun. Dcdetector: Dual attention contrastive representation learning for time series anomaly detection. In *KDD*, pp. 3033–3045, 2023.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI*, volume 37, pp. 11121–11128, 2023.

Kai Zhao, Zhihao Zhuang, Chenjuan Guo, Hao Miao, Christian S. Jensen, Yunyao Cheng, and Bin Yang. Unsupervised time series anomaly prediction with importance-based generative contrastive learning. In *KDD*, pp. 3945–3956, 2025. doi: 10.1145/3711896.3737174.

Sinong Zhao, Wenrui Wang, Hongzuo Xu, Zhaoyang Yu, Qingsong Wen, Gang Wang, Guansong Pang, et al. Abnormality forecasting: Time series anomaly prediction via future context modeling. *arXiv preprint arXiv:2410.12206*, 2024.

Quan Zhou, Changhua Pei, Fei Sun, Zhengwei Gao, Haiming Zhang, Gaogang Xie, Dan Pei, et al. Kan-ad: Time series anomaly detection with kolmogorov–arnold networks. In *ICML*, 2025.

Appendix

A Additional Information

The symbols used in this paper are summarized in Table 5. Ane the computation of the channel importance weights $\mathbf{w}_{\mathrm{ch}}$ is summarized in Algorithm 1.

Algorithm 1 Channel Importance Computation

Input: Frozen reconstruction branches $f_{\mathrm{rec}}^{(\mathrm{seas})}$ and $f_{\mathrm{rec}}^{(\mathrm{trend})}$; training history windows $\mathcal{B}$
Output: Channel weights $\mathbf{w}_{\mathrm{ch}} \in \mathbb{R}^N$

Initialize $\mathbf{s} \leftarrow \mathbf{0} \in \mathbb{R}^N$ and $m \leftarrow 0$
for each history window $\mathbf{X}_t^{(h)}$ in $\mathcal{B}$ **do**
  $\mathbf{X}_t^{(h,\mathrm{trend})} \leftarrow \mathrm{AvgPool}(\mathbf{X}_t^{(h)})$; $\mathbf{X}_t^{(h,\mathrm{seas})} \leftarrow \mathbf{X}_t^{(h)} - \mathbf{X}_t^{(h,\mathrm{trend})}$
  for $c \in \{\mathrm{seas}, \mathrm{trend}\}$ **do**
    $\hat{\mathbf{X}}_t^{(h,c)} \leftarrow f_{\mathrm{rec}}^{(c)}(\mathbf{X}_t^{(h,c)})$
  end for
  $\mathbf{E}_t \leftarrow \sum_{c\in\{\mathrm{seas},\mathrm{trend}\}} \left|\mathbf{X}_t^{(h,c)} - \hat{\mathbf{X}}_t^{(h,c)}\right|$
  $\mathbf{s} \leftarrow \mathbf{s} + \sum_{i=1}^{b} \mathbf{E}_{t,i}$; $m \leftarrow m + b$
end for
$\mathbf{w}_{\mathrm{ch}} \leftarrow \mathbf{s}/\max(m, 1)$
return $\mathbf{w}_{\mathrm{ch}}$

B Experimental Details

B.1 Baselines

All baselines are implemented in PyTorch and trained on a single RTX 4090 GPU (48 GB), using a batch size of 256. Hyperparameters for all methods are reported in Appendix B.3.

These Forecast-then-Detect baselines use a two-stage protocol in which the forecast sequences produced by the forecasting stage are passed to the detection stage. The procedure consists of the following two steps:

Table 5: Notation used in this paper.

Symbol	Description
T, N	Series length and number of channels
$x_t \in \mathbb{R}^{1\times N}$	Observation at time t
$\mathcal{X} = \{x_t\}_{t=1}^{T}$	Input time series
b, f	Look-back length and forecast horizon
$\mathbf{X}_t^{(h)} \in \mathbb{R}^{b\times N}$	Historical window
$\mathbf{X}_t^{(f)} \in \mathbb{R}^{f\times N}$	Future window (training only)
$\hat{Y}_t \in \{0,1\}^f$	Predicted anomaly labels for the future window
$\mathrm{AvgPool}$	Length-preserving moving-average pooling
$c \in \{\mathrm{seas}, \mathrm{trend}\}$	Seasonal or trend branch
$\mathbf{Z}_t^{(c)} \in \mathbb{R}^{(b+f)\times N}$	Concatenation of the history component and the future window
$f_{\mathrm{rec}}^{(c)}$	Reconstruction network of branch c
$\mathbf{R}_t^{(u,c)}$, $u \in \{h, f\}$	Historical and future residual representations of branch c
$\mathcal{S}$, s	Set of temporal scales and a scale in it
Seg_s	Operator keeping the first s steps and zero-masking the rest
$\mathbf{R}_{\mathrm{ms},t}^{(h)}$, $\mathbf{R}_t^{(f)}$	Multi-scale historical and future representations
$\mathcal{M} = \{(\mathbf{r}_j, \mathbf{v}_j)\}_{j=1}^{M}$	Memory bank with capacity M
ρ_j	Importance score of memory entry j
f_{enc}, $\mathbf{q}_t$, $\mathbf{k}_j$	Query/key encoder, query embedding, and key embeddings
sim, τ, K_{neg}	Cosine similarity, temperature, and number of negatives
f_{pred}	Parametric predictor
$K, \mathcal{N}_K(t), \omega_{t,j}$	Retrieval count, retrieved index set, and retrieval weights
$\hat{\mathbf{R}}_t^{(\mathrm{base})}$, $\hat{\mathbf{R}}_t^{(\mathrm{mem})}$	Base and memory-retrieved predictions
$\boldsymbol{\alpha}_t \in (0,1)^{f\times N}$	Element-wise gate weights
$\hat{\mathbf{R}}_t^{(\mathrm{final})}$, $\tilde{\mathbf{R}}_t^{(\mathrm{final})}$	Fused prediction and calibrated anomaly scores
$\mathbf{w}_{\mathrm{ch}} \in \mathbb{R}^N$	Channel importance weights
$\mathcal{L}_{\mathrm{rec}}, \mathcal{L}_{\mathrm{cl}}, \mathcal{L}_{\mathrm{pred}}$	Reconstruction, contrastive, and prediction losses

1. **Forecasting.** We first train the forecasting model on the training split with its native objective and use the validation split for early stopping. The trained model then generates forecasts for the training, validation, and test sequences in a sliding-window manner with a look-back length of 192. Because adjacent forecast windows overlap, we average the predictions covering each time step to obtain one forecast sequence per split. The first 192 time steps are treated as a warm-up region.
2. **Detection.** We train the detection model on the forecast sequence of the training split using its native objective and apply the same early-stopping protocol to the validation forecasts. At inference time, the detector assigns an anomaly score to each sliding window of the test forecast sequence. We average the scores from overlapping windows at each time step to obtain the final point-level anomaly score.

For MOMENT, we use the 40M-parameter version and follow its recommended protocol: we fine-tune its forecasting head and use its pretrained reconstruction head for zero-shot anomaly detection.

B.2 Implementation Details

We set the scale factor to 3, the memory bank capacity to 256, and the Top-K in memory retrieval to 3. Since $b = 192$, scale factor 3 constructs the multi-scale set as $\mathcal{S} = \{b/3, 2b/3, b\} = \{64, 128, 192\}$. The temperature τ is set to 1 and the number of negatives K_{neg} to 3. Both training stages run for 10 epochs, with early stopping on the validation split. For each time step covered by multiple overlapping prediction windows, we average its anomaly scores to obtain the final score.

B.3 HYPERPARAMETER SETTINGS

The hyperparameters of the baseline methods are set as follows. All baselines are trained on z-normalized inputs with a standard scaler, except that MOMENT uses a robust scaler. The hyperparameters of LAMP follow the implementation details in the main paper.

- **Autoformer**, the window size of the series-decomposition moving average is set to 25. The encoder consists of 2 layers and the decoder of 1 layer, with 4 attention heads, a model dimension of 256, a feed-forward dimension of 512, and GELU activation. The dropout rate is 0.01.
- **PatchTST**, the patch length and stride are set to 16 and 8. The model stacks 3 Transformer encoder layers with 16 attention heads, a model dimension of 128, and a feed-forward dimension of 256.
- **DLinear**, a channel-shared linear layer is employed for prediction, and the window size of the series-decomposition moving average is set to 25.
- **Duet**, the encoder and decoder consist of 2 and 1 layers, respectively, with 8 attention heads, a model dimension of 256, a feed-forward dimension of 1024, and a hidden size of 128. The segment length, window size, patch length, stride, and period length are set to 6, 2, 16, 8, and 4, respectively, and the moving-average window is 25. The mixture-of-experts layer contains 4 experts with top-1 routing and noisy gating. The dropout rates are 0.05 for the model and 0.2 for the fc layer, and the Huber loss is adopted.
- **DCdetector**, the patch size is set to 55 and the top-k of the guided attention is 3. The model stacks 3 encoder layers with 4 attention heads and a model dimension of 256. The dropout rate is 0.01.
- **ModernTCN**, the patch size and stride are set to 16 and 8, respectively. The network comprises 4 stages, each containing a single block, with model and depthwise dimensions of 256 across stages; the large kernel sizes of the four stages are [31, 29, 27, 13] and the small kernel sizes are [5, 5, 5, 5]. The stem ratio, downsample ratio, and FFN ratio are 6, 2, and 2, respectively, and the multi-scale architecture is enabled.
- **KAN-AD**, the order of the Fourier basis expansion is set to 2, yielding $2\times2+1$ input channels. The backbone stacks four bias-free convolutional layers with batch normalization, GELU activation, and two residual connections.
- **MOMENT**, the input windows are padded to the model context length of 512. Following a linear-probing protocol, the encoder and the embedder are frozen, and only the forecasting head is fine-tuned, with a head dropout of 0.1 and the MSE loss.
- **A2P**, the forecasting and detection branches stack 3 encoder layers with 8 attention heads, a model dimension of 256, and a feed-forward dimension of 512. The patch length and stride are 10 and 8 with end padding, and the top-k of the prior association is 3. The anomaly prompt pool has a size of 10, from which the top-3 prompts are retrieved.
- **FCM**, each of the dual Transformer encoders stacks 2 encoder layers with an attention factor of 3, a dropout rate of 0.1, GELU activation, and time-feature embedding.
- **IGCL**, the network stacks 3 blocks of 3 dilated causal convolution layers with a kernel size of 5. The residual and dilation channel dimensions are 4, the skip and end channel dimensions are 32, and the dropout rate is 0.2. The look-ahead window for label construction is 8.

B.4 EFFICIENCY ANALYSIS DETAILS

We benchmark training efficiency on the GECCO dataset under a unified protocol, reporting for each method the wall-clock time per training epoch and the peak GPU memory. The per-epoch time is taken from the second training epoch to exclude one-time overheads such as CUDA initialization and kernel autotuning, and it includes the validation pass for methods that perform validation within each epoch. For the Forecast-then-Detect baselines, whose pipeline consists of forecaster training followed by detector training, we report the sum of the per-epoch times of the two stages.

The peak GPU memory is the per-process maximum reported by `nvidia-smi`, polled once per second during training. All experiments use a batch size of 256 and run exclusively on a single RTX 4090 GPU (48 GB), so that no measurement is affected by concurrent workloads. The forecast horizon is varied over $\{32, 96, 384, 576\}$ with a look-back length of 192, and the reported values are averaged over the four horizons.

C Full Results

C.1 Additional Evaluation Metrics

Notations For a dataset and forecast horizon, let $\mathbf{y} = (y_1, \ldots, y_T) \in \{0,1\}^T$ denote the ground-truth labels of the evaluated time points and let $\mathbf{s} = (s_1, \ldots, s_T) \in \mathbb{R}^T$ denote the corresponding anomaly scores, where a larger score indicates a higher likelihood of an anomaly. Given a threshold λ, the binary prediction is $\hat{y}_t(\lambda) = \mathbb{I}[s_t \geq \lambda]$. We use $\mathrm{TP}(\lambda)$, $\mathrm{FP}(\lambda)$, and $\mathrm{FN}(\lambda)$ to denote the resulting point-wise counts, and regard each maximal contiguous interval of positive labels as an anomaly event.

Point-F1 At threshold λ, point-wise precision and recall are $P(\lambda) = \mathrm{TP}(\lambda)/(\mathrm{TP}(\lambda) + \mathrm{FP}(\lambda))$ and $R(\lambda) = \mathrm{TP}(\lambda)/(\mathrm{TP}(\lambda) + \mathrm{FN}(\lambda))$, respectively. Their harmonic mean is $F_1(\lambda) = 2P(\lambda)R(\lambda)/(P(\lambda)+R(\lambda))$. Point-F1 is the maximum $F_1(\lambda)$ over all distinct thresholds induced by the anomaly scores. It therefore measures the best point-wise discrimination attainable by a method, but depends on post-hoc threshold selection.

Affiliation-F1 Affiliation-F1 evaluates predicted and ground-truth anomaly events using the local affiliation framework of Huet et al. (2022). The timeline is partitioned into affiliation zones associated with individual ground-truth events. Affiliation precision measures the temporal proximity of predicted anomalous points to their affiliated events, while affiliation recall measures how well each ground-truth event is covered; both quantities are normalized relative to random predictions within the corresponding zone. Affiliation-F1 is their harmonic mean. For continuous anomaly scores, predicted labels are obtained by thresholding the scores: a time step is predicted anomalous if and only if its score is no smaller than the threshold λ, i.e., $\hat{y}_t(\lambda) = \mathbb{I}[s_t \geq \lambda]$. Unlike Point-F1, which relies on post-hoc threshold selection, we fix λ to the $(100-\eta)$th percentile of the anomaly scores concatenated over the training and test sets, following Xu et al. (2022); Yang et al. (2023), where η denotes the anomaly rate of the corresponding dataset.

AUC-PR & AUC-ROC AUC-PR is the area under the precision–recall curve obtained by sweeping λ, whereas AUC-ROC is the area under the receiver operating characteristic curve, which plots the true-positive rate against the false-positive rate. Unlike Point-F1, both metrics summarize performance over all thresholds. AUC-PR is particularly informative for the highly imbalanced anomaly-prediction setting, while AUC-ROC measures the overall ranking quality of anomaly scores.

VUS-PR & VUS-ROC Volume Under the Surface (VUS) metrics extend AUC-PR and AUC-ROC by evaluating each threshold across a range of temporal tolerance-window sizes (Paparrizos et al., 2022; Boniol et al., 2025). For each window size, range-aware labels give partial credit to predictions near a ground-truth anomaly interval; the resulting PR or ROC surface is then integrated over both the decision threshold and the tolerance-window size. This reduces sensitivity to a single threshold or a manually selected tolerance window while accounting for slight temporal misalignment. We denote VUS-PR and VUS-ROC as V-PR and V-ROC, respectively, in the main paper.

Table 6 shows that LAMP achieves the best Point-F1 at five of the six horizons, the best AUC-PR at four horizons, and the best AUC-ROC at all horizons, confirming its robust point-level anomaly prediction ability.

Discussion Point adjustment (Kim et al., 2022) treats an entire anomalous segment as correctly detected once any point in the segment is identified as anomalous. Threshold tuning is typically performed as a post-hoc step, making the reported score sensitive to the chosen thresholding strategy (Paparrizos et al., 2022; Liu & Paparrizos, 2024). In addition, a universal threshold is difficult to define because anomaly rates vary substantially across datasets. Following these works (Paparrizos et al., 2022; Liu & Paparrizos, 2024; Boniol et al., 2025), we adopt the V-PR and V-ROC as our primary metrics. These metrics provide a more robust evaluation of anomaly prediction performance without relying on point-wise adjustments or threshold tuning.

Table 6: Anomaly prediction results under additional evaluation metrics, averaged over five datasets. Point-F1 (%), Affiliation-F1 (%), AUC-PR (%), and AUC-ROC (%) are reported for forecast horizons of 32, 64, 96, 192, 384, and 576. The best result at each horizon is highlighted in bold.

Method		Metric											
		Point-F1						Affiliation-F1					
		32	64	96	192	384	576	32	64	96	192	384	576
Auto	DC	16.58	16.47	16.31	16.25	16.77	16.42	67.56	67.76	67.62	**68.06**	64.83	**68.28**
	Mod	20.50	20.61	20.01	18.08	18.00	18.01	57.71	56.19	55.57	48.50	49.65	50.77
	KAN	23.63	22.02	21.22	20.04	20.74	20.58	40.78	38.63	37.01	33.40	32.60	25.97
Patch	DC	16.49	16.39	17.03	16.87	16.35	16.35	67.37	67.69	67.32	65.98	64.92	64.46
	Mod	25.23	24.09	24.57	20.94	19.39	19.15	66.21	65.42	63.68	59.66	52.60	52.27
	KAN	25.42	25.07	24.27	19.87	17.78	16.54	42.26	41.56	40.39	33.74	35.77	35.33
DLin	DC	16.76	17.07	17.61	17.00	17.06	16.96	**67.58**	**67.80**	**67.65**	67.10	67.65	67.36
	Mod	24.61	23.83	22.30	20.44	19.19	18.07	60.79	57.92	53.82	54.30	54.24	46.91
	KAN	20.73	21.76	24.59	22.60	19.93	19.26	39.41	39.77	43.07	43.49	42.11	34.75
Duet	DC	16.45	16.33	16.62	16.34	16.79	16.74	67.24	67.47	66.83	65.52	**67.68**	67.45
	Mod	26.25	27.03	25.69	24.71	24.20	23.45	67.39	63.85	61.70	54.12	53.99	47.01
	KAN	21.79	21.26	24.09	17.30	17.61	18.30	36.31	33.52	37.90	31.14	31.09	31.91
MOM	MOM	23.68	23.97	27.06	25.60	**42.10**	-*	55.02	53.05	52.72	57.84	59.39	-*
FCM		16.29	16.58	17.00	18.56	18.61	18.67	59.38	59.56	60.86	58.43	59.94	56.96
A2P		20.21	19.14	18.29	18.07	18.41	18.64	61.76	64.09	64.69	64.74	62.24	62.41
IGCL		21.12	20.99	22.05	24.15	22.81	23.34	67.56	67.51	67.51	67.47	67.28	67.22
LAMP(Ours)		**38.61**	**36.56**	**38.28**	**37.53**	36.93	**37.42**	62.68	67.30	65.95	67.51	66.28	66.39

Method		Metric											
		AUC-PR						AUC-ROC					
		32	64	96	192	384	576	32	64	96	192	384	576
Auto	DC	9.66	9.63	9.56	9.47	9.48	9.55	49.53	49.99	49.73	49.76	48.62	50.00
	Mod	11.26	11.56	12.00	11.35	11.90	10.14	55.09	56.65	56.23	53.84	53.76	53.28
	KAN	15.12	13.74	11.76	10.55	10.98	11.09	56.50	53.76	49.81	47.91	47.44	47.57
Patch	DC	9.48	9.41	9.68	9.66	9.60	9.49	47.34	48.24	51.18	50.01	47.17	47.49
	Mod	17.06	15.92	16.69	14.63	12.55	12.02	52.39	50.81	49.14	50.86	54.51	44.27
	KAN	16.97	17.21	16.13	13.64	11.99	9.97	63.91	62.54	65.09	56.95	61.69	60.14
DLin	DC	9.53	9.61	9.86	9.85	9.95	9.88	48.95	50.63	52.02	51.95	51.31	50.90
	Mod	18.83	16.28	14.68	13.80	12.49	11.42	62.36	60.32	58.19	59.20	57.82	55.79
	KAN	13.67	14.32	15.76	13.17	13.18	13.40	48.43	49.13	49.63	51.52	49.66	50.53
Duet	DC	9.45	9.45	9.79	9.67	9.77	9.67	47.82	50.02	51.06	49.70	51.44	49.77
	Mod	19.11	20.08	18.62	16.70	16.37	15.46	64.82	61.45	59.72	59.09	58.07	56.84
	KAN	14.61	14.43	14.62	11.58	11.17	11.44	53.96	49.29	52.58	48.15	45.66	49.57
MOM	MOM	15.12	16.38	17.97	**25.11**	**31.00**	-*	57.20	56.87	56.81	58.64	57.07	-*
FCM		9.75	9.92	10.15	11.15	11.15	11.18	49.98	50.04	50.09	50.05	50.06	50.02
A2P		14.40	12.96	12.26	11.83	11.94	11.86	53.26	52.81	52.41	50.39	49.55	50.75
IGCL		14.38	15.99	16.41	15.34	16.89	17.62	50.68	53.73	54.63	53.89	52.94	58.42
LAMP(Ours)		**20.65**	**22.79**	**21.99**	22.84	24.72	**30.97**	**74.84**	**76.91**	**76.00**	**77.51**	**75.42**	**76.91**

* The maximum forecast horizon of the MOMENT (MOM) is 512 time steps.

C.2 Channel Importance Validation

The ablation in Table 4 shows that removing channel score calibration decreases the average V-ROC by 6.08%. To understand where this gain comes from, we examine whether the label-free channel importance weights $\mathbf{w}_{\mathrm{ch}}$ are consistent with label-based channel relevance. For this analysis only, we use the test set to compute the Point-biserial correlation between each channel and the anomaly labels, and normalize the resulting relevance scores to sum to one; the labels are used solely for this diagnostic and are never involved in training, scoring, or thresholding. We compare these label-based relevance scores with $\mathbf{w}_{\mathrm{ch}}$, which is computed by Algorithm 1 without using anomaly labels under a forecast horizon of 32. To quantify the consistency, we report Pearson correlation r and NDCG@K. Pearson correlation measures the global linear agreement between the two normalized channel-importance distributions. NDCG@K evaluates whether the top-K channels ranked by $\mathbf{w}_{\mathrm{ch}}$ match the label-based relevance ranking (Järvelin & Kekäläinen, 2002). As shown in Figure 6, LAMP achieves strong consistency on both datasets. On PSM, it obtains $r = 0.902$, NDCG@10$= 0.993$, and NDCG@20$= 0.984$. On SMAP, it obtains $r = 0.852$,

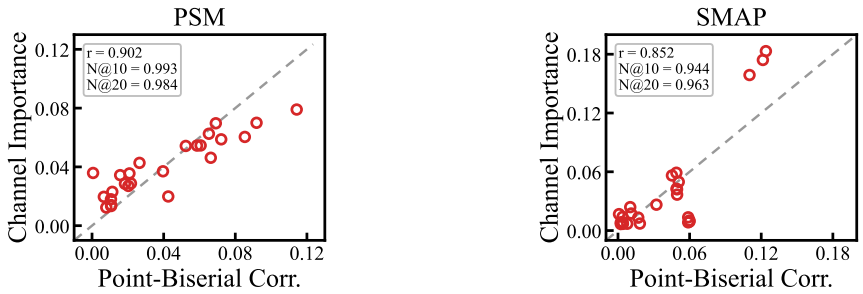

Figure 6: Consistency between label-based channel relevance and the learned channel importance weights on the PSM and SMAP datasets. The x-axis reports the normalized Point-biserial correlation between each channel and the anomaly labels on the test set. The y-axis reports the normalized channel importance weights $\mathbf{w}_{\text{ch}}$ computed by LAMP. Pearson correlation r measures global linear consistency, while NDCG@K measures whether the top-ranked channels under $\mathbf{w}_{\text{ch}}$ align with the label-based relevance ranking.

NDCG@10= 0.944, and NDCG@20= 0.963. These results suggest that the learned weights identify channels that are more related to anomalous states and reduce the dilution of anomaly-relevant channels during score aggregation, which explains the performance drop when channel score calibration is disabled in the ablation. However, in this unsupervised paradigm, noise may inflate the residuals of normal channels, while anomalous channels with small reconstruction errors are down-weighted and remain undetectable.

C.3 Scalability Analysis

We analyze the scalability of LAMP under different data scales. Specifically, we vary the training data scale from 10% to 100%. Figure 7 shows that LAMP benefits clearly from additional training data. Under the forecast horizon of 32, both V-PR and V-ROC improve as the data scale increases. Overall, these results suggest that the proposed residual representation and memory retrieval mechanism can effectively exploit larger training data.

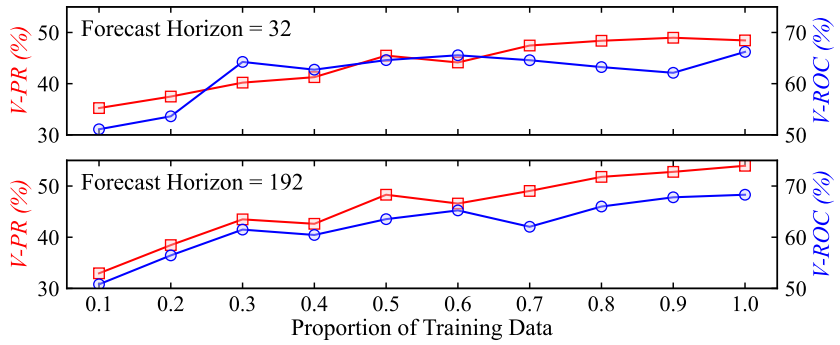

Figure 7: Data-scale analysis. We report V-PR (%) and V-ROC (%) by varying the proportion of training data from 10% to 100% under a forecast horizon of 32, averaged over five datasets.