ANOMAMIND: AGENTIC TIME SERIES ANOMALY DETECTION WITH TOOL-AUGMENTED REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series anomaly detection is critical in many real-world applications, where effective solutions must localize anomalous regions and support reliable decision-making under complex settings. However, many existing methods use predefined modeling and scoring pipelines rather than an evidence-driven diagnostic process. This can limit their ability to adapt evidence acquisition to context-dependent anomalies and heterogeneous patterns. To address these challenges, we propose AnomaMind, a supervised agentic time series anomaly detection framework that organizes interval-level detection as a sequential decision-making process. AnomaMind operates through a coarse-to-fine workflow that first localizes suspicious intervals, then constructs diagnostic evidence through tool interaction, and finally refines anomaly decisions through self-reflection. The workflow is supported by reusable tool engines for context-aware diagnostic analysis. A key design is a hybrid inference mechanism: frozen general-purpose models handle tool interaction and self-reflective refinement, while a detection-specific policy is trained through supervised fine-tuning and reinforcement learning with verifiable output-level rewards. This combines task-specific optimization with flexible tool orchestration. Experiments under in-domain and multi-domain joint-training settings show strong detection performance on four benchmarks. The code is available at https://anonymous.4open.science/r/AnomaMind.

1 INTRODUCTION

Time series anomaly detection (TSAD) plays a critical role in many real-world systems, where detected anomalies often trigger downstream actions such as alarms, diagnosis, or intervention (Liu & Paparrizos, 2024; Wang et al., 2026). In such settings, effective anomaly detection requires more than assigning anomaly scores to individual observations. Practical solutions need to localize anomalous intervals, interpret diagnostic evidence, and support decision-making under evolving contextual conditions with limited or uncertain information. As time series data become increasingly diverse and context-dependent, these requirements call for evidence-driven and decision-oriented detection paradigms beyond pure predictive accuracy (Qiu et al., 2025).

Over the years, TSAD has witnessed substantial progress across diverse methodological paradigms, including statistical, density-based, reconstruction-based, and forecasting-based approaches (Gupta et al., 2013). Statistical methods characterize normal temporal behaviors through distributional or structural assumptions and identify significant deviations as anomalies (Thill et al., 2017), while density-based methods detect observations lying in sparse or low-density regions (Breunig et al., 2000; Schölkopf et al., 2001). Reconstruction-based approaches learn representations of normal patterns and detect anomalies through reconstruction discrepancies (Audibert et al., 2020; Xu et al., 2018), whereas forecasting-based methods model temporal dependencies and use prediction residuals to identify abnormal observations (Munir et al., 2018; Malhotra et al., 2015; Deng & Hooi, 2021). Despite these advances, most conventional methods still follow model-centric detection or scoring paradigms, relying on predefined modeling objectives and anomaly scoring mechanisms rather than an evidence-driven diagnostic process (Schmidl et al., 2022). Consequently, their inference procedures remain relatively fixed and can be less adaptive when anomaly patterns evolve or the underlying data distribution shifts (Kim et al., 2024; Zhang et al., 2022).

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.

We argue that these limitations stem from a fundamental mismatch between how anomaly detection is commonly formulated and how it is carried out in practice (Cheng et al., 2026). In real-world scenarios, TSAD is neither a single scoring step nor a fixed one-shot classification problem. Instead, it typically unfolds as a sequential decision-making process, where analysts progressively narrow down suspicious intervals, examine informative evidence, and update judgments based on intermediate findings (Gu et al., 2024). Decisions made at earlier stages influence what evidence is gathered next and how subsequent analyses are conducted across temporal scales and contextual assumptions (Jiang et al., 2024). However, conventional model-centric formulations are not designed to represent such diagnostic procedures (Yang et al., 2025a). These observations motivate moving beyond isolated predictive models toward agentic time series systems that organize anomaly detection as a system-level reasoning and decision process, integrating evidence acquisition, analysis, and refinement within an adaptive workflow under evolving real-world operating conditions.

While this perspective motivates a shift toward agentic time series systems, realizing it in practice poses substantial challenges in complex real-world anomaly scenarios. Effective anomaly reasoning requires diverse, reusable analytical capabilities, calling for tool engines that support feature preparation and diagnostic analysis without hard-coding task-specific pipelines (Wu et al., 2026). The detection workflow must also support progressive decision-making, integrating coarse localization with fine-grained analysis rather than treating them as isolated stages (Xu et al., 2022). Such a coarse-to-fine workflow enables progressively refined and interpretable anomaly analysis. Beyond workflow design, agentic inference introduces challenges in stability and optimization. While general-purpose models support open-ended reasoning and tool orchestration, their general-purpose objectives may not be sufficiently aligned with interval-level anomaly detection (Alnegheimish et al., 2024; Gu et al., 2025). This motivates a hybrid inference paradigm that separates flexible reasoning from task-specific decision learning, combining general-purpose models with explicit and verifiable learning objectives to enable anomaly reasoning.

To address these challenges, we propose AnomaMind, an agentic TSAD framework that formulates anomaly detection as a sequential decision-making process. AnomaMind operates through a workflow that incrementally constructs diagnostic evidence and refines anomaly decisions over multiple steps. Rather than relying on fixed feature inputs, AnomaMind progressively localizes anomalous intervals in a coarse-to-fine manner and augments detection through a toolkit box that integrates knowledge memory and numerical diagnostics. The knowledge memory includes visual anomaly patterns mined from training data, domain knowledge, and tool descriptions, while the numerical evidence toolkit provides statistical, value-based, change-based, and region-level operators for measurable verification. Crucially, AnomaMind adopts a hybrid inference mechanism: general-purpose models handle tool invocation and self-reflective refinement, while core anomaly detection decisions are optimized through reinforcement learning under verifiable output-level feedback. This design enables AnomaMind to combine flexible reasoning with task-specific optimization, while maintaining stability in complex detection scenarios.

Our main contributions are summarized as follows:

- We organize interval-level TSAD as a coarse-to-fine diagnostic process that couples adaptive evidence construction with a trainable detection policy.
- We propose **AnomaMind**, an agentic framework built around a coarse-to-fine workflow and a toolkit box that combines knowledge memory and numerical diagnostics.
- We design a hybrid inference mechanism in which frozen general-purpose models perform flexible reasoning and tool orchestration, while a detection-specific policy is optimized with verifiable output-level rewards.
- We conduct comprehensive experiments across widely used time series anomaly detection benchmarks, showing strong detection performance and supporting the effectiveness of the proposed agentic, tool-augmented framework.

2 Related Work

We review conventional and LLM-based approaches to TSAD, followed by recent advances in agentic and tool-augmented reasoning.

2.1 Conventional Anomaly Detection

TSAD spans statistical, density-based, reconstruction-based, and forecasting-based paradigms (Chandola et al., 2009). USAD and xLSTMAD use reconstruction or prediction discrepancies (Audibert et al., 2020; Faber et al., 2025), while TimesNet and MOMENT support general time-series analysis (Wu et al., 2023; Goswami et al., 2024). Multi-scale forecasting approaches include multi-resolution diffusion and recursive modeling of irregular series (Shen et al., 2024; Li et al., 2026). Complementary work selects TSAD models without anomaly labels through surrogate-metric rank aggregation (Goswami et al., 2023). These approaches focus on modeling, scoring, or model selection rather than adaptive evidence acquisition.

2.2 LLM-based Anomaly Detection

Recent studies have increasingly explored large language models (LLMs) for TSAD. Existing LLM-based TSAD approaches mainly rely on prompting or task-specific training. Prompting methods judge textualized series (Liu et al., 2025b; Alnegheimish et al., 2024), while visual approaches analyze time-series plots (Bechar et al., 2025). For zero-shot image anomaly detection, AnomalyCLIP learns object-agnostic prompts for a vision-language model (Zhou et al., 2024). AnomalyLLM instead distills anomaly-aware representations into a lightweight time-series detector (Liu et al., 2024). These approaches generally lack an explicit multi-step tool-use workflow for iterative evidence acquisition, diagnostic verification, and progressive decision refinement.

2.3 Agentic and Tool-Augmented Reasoning

Agentic approaches interleave reasoning and tool use. ARGOS synthesizes and validates detection rules (Gu et al., 2025); EvoTest evolves prompts, memory, and tool-use routines through test-time feedback (He et al., 2026). TimeOmni-1 uses supervised fine-tuning and reinforcement learning for time-series reasoning (Guan et al., 2026), while TimeSeriesExamAgent constructs reasoning benchmarks using LLM agents (Gwiazda et al., 2026). TimeART learns analytical tool use for time-series question answering (Wu et al., 2026), and Cast-R1 learns tool-augmented forecasting policies (Tao et al., 2026b). AnomaMind instead targets interval-level TSAD through coarse-to-fine evidence construction, detector-specific learning under verifiable rewards, and frozen general-purpose models for tool orchestration.

3 Preliminaries

We first formulate time series anomaly detection from a sequential decision-making perspective and then summarize the reasoning process that motivates agentic detection.

3.1 Problem Formulation

We formalize TSAD as a *sequential decision-making process*. Given an input time series $\mathbf{x}_{1:T} = \{x_1, x_2, \ldots, x_T\}$ and optional contextual information $\mathbf{c}_{1:T}$, detection proceeds over steps $k = 1, \ldots, K$ rather than relying on a fixed one-shot feature input. At step k, the detection process maintains a state s_k that contains the original observations, contextual signals, intermediate analysis results, and evidence produced by external tools. Based on this evolving state, an action a_k is selected from a structured action space, such as refining candidate regions, invoking diagnostic analyses, or updating anomaly judgments. The action returns an observation or evidence item o_k, which is incorporated into the next state s_{k+1}. After a finite decision horizon or a stopping condition, the process outputs a set of detected anomalous intervals $\hat{\mathcal{S}}$. This formulation enables anomaly decisions to be progressively refined through multi-step evidence acquisition and revision.

3.2 Thinking Process for Anomaly Detection

Practical anomaly diagnosis can involve multiple stages. It often begins with coarse screening to identify potentially suspicious intervals while filtering evidently normal regions (Chandrayan et al., 2025). Targeted diagnostic analysis is then applied to these intervals by examining local statistical irregularities, temporal patterns, and contextual inconsistencies (Zhou & Yu, 2025; Tao et al., 2026a).

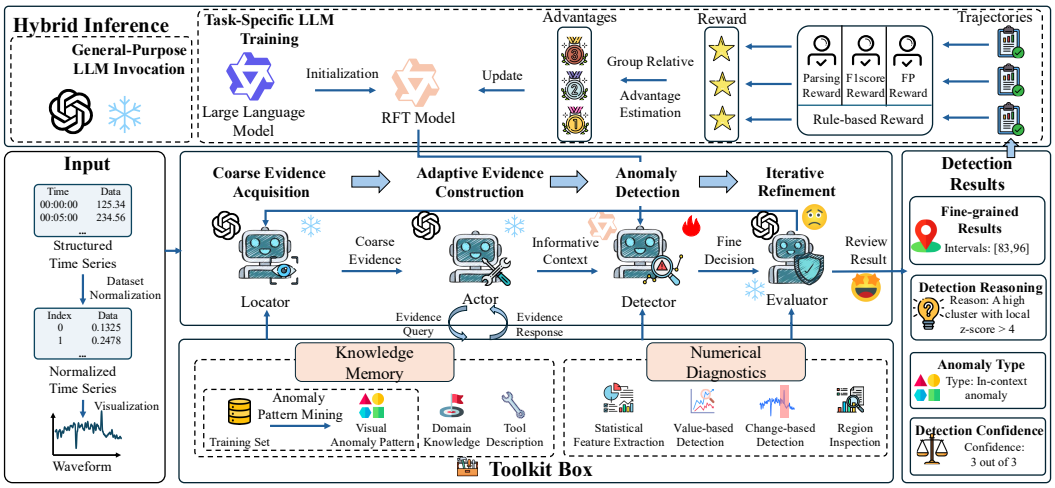


Figure 1: Overview of AnomaMind, an agentic time series anomaly detection framework that adopts a structured coarse-to-fine workflow built upon a toolkit box and supported by hybrid inference.

Since different intervals may require different evidence, early judgments are provisional and may be progressively revised when additional information becomes available, rather than relying on fixed thresholds or one-size-fits-all scoring rules across operational contexts (Gu et al., 2024; Jiang et al., 2026). This process suggests an agentic formulation in which anomaly detection is organized as candidate localization, evidence construction, and decision refinement.

4 The Proposed AnomaMind

AnomaMind combines a reusable evidence toolkit, a coarse-to-fine diagnostic workflow, and a hybrid decision mechanism for time series anomaly detection.

4.1 Framework Overview

Figure 1 presents the overall framework. A toolkit box supplies visual patterns, domain knowledge, tool descriptions, and numerical diagnostics instead of requiring the language model to infer numerical evidence internally. Four roles then coordinate sequentially: the locator proposes suspicious intervals, the actor gathers evidence, the detector makes interval-level decisions, and the evaluator either accepts the result or requests refinement. This organization narrows the search space before fine-grained reasoning and allows later decisions to reuse accumulated evidence. AnomaMind uses general-purpose LLMs for flexible perception and tool orchestration, while a detection-specific policy learns precise anomaly decisions from task feedback.

4.2 Toolkit Box Construction

The toolkit box is $\mathcal{B} = \{\mathcal{K}, \mathcal{N}\}$, where $\mathcal{K}$ denotes reusable knowledge and $\mathcal{N}$ denotes numerical diagnostics. At step k, the workflow maintains candidate intervals $\mathcal{S}^{(k)}$, accumulated diagnostics $\mathcal{Z}_{1:k}$, and retrieved context $\mathcal{C}_{1:k}$. The role-specific use of these resources is summarized in Table 1. Toolkit actions belong to $\mathbb{A}_{tool} = \mathbb{A}_{retrieve} \cup \mathbb{A}_{numeric}$: retrieval actions select contextual resources, while numerical actions select an operator and its arguments. Writing F_a for the interface associated with action a, a toolkit call produces:

$$o_k = F_{a_k}(\mathbf{x}_{1:T}, \mathcal{S}^{(k)}, \mathcal{C}_{1:k}), \tag{1}$$

where o_k is appended to the diagnostic or contextual pool according to the selected component.

Knowledge Memory. The knowledge memory contains:

$$\mathcal{K} = \{\mathcal{K}_{vap}, \mathcal{K}_{dom}, \mathcal{K}_{tool}\}, \tag{2}$$

Table 1: Role-specific use of knowledge memory and numerical diagnostics in AnomaMind.

Role	Workflow Stage	Knowledge Memory	Numerical Diagnostics	Output
Locator	Coarse Evidence Acquisition	Visual Anomaly Pattern; Domain Knowledge	Not directly used	Coarse Evidence
Actor	Adaptive Evidence Construction	Domain Knowledge; Tool Description	Invoke and construct	Informative Context
Detector	Anomaly Detection	Domain Knowledge	Use accumulated diagnostics	Fine Decision
Evaluator	Iterative Refinement	Domain Knowledge	Check diagnostic consistency	Review Feedback

where $\mathcal{K}_{vap}$ stores visual anomaly prototypes, $\mathcal{K}_{dom}$ stores anomaly semantics and domain constraints, and $\mathcal{K}_{tool}$ describes the interfaces and intended uses of available tools.

To construct $\mathcal{K}_{vap}$, we extract labeled anomalous segments from the training data, divide them into short, medium, and long groups, and map each segment to an 18-dimensional descriptor. The descriptor covers segment length and anomaly ratio, anomaly statistics, contextual deviation, boundary contrast, and temporal variation. Length-stratified KMeans then selects the segment nearest each cluster centroid as a visual prototype. Algorithm 1 summarizes the procedure, and Figure 4 shows representative anomaly types used to organize $\mathcal{K}_{dom}$. Retrieved knowledge is added to $\mathcal{C}_{1:k}$ to guide localization, tool selection, and detection.

Numerical Diagnostics. The numerical toolkit contains:

$$\mathcal{N} = \{\mathcal{N}_{stat}, \mathcal{N}_{val}, \mathcal{N}_{chg}, \mathcal{N}_{reg}\}. \tag{3}$$

$\mathcal{N}_{stat}$ reports local and global statistics such as the mean, variance, percentiles, extrema, and median. $\mathcal{N}_{val}$ detects magnitude outliers relative to global value statistics, while $\mathcal{N}_{chg}$ detects abrupt transitions from normalized adjacent-point differences. $\mathcal{N}_{reg}$ returns the values in a requested interval for direct inspection. For an interval $I_j = [l_j, r_j]$, a numerical action returns $z_j^{(k)} = F_{a_k}(\mathbf{x}_{1:T}, I_j)$, retaining the full input as context when global statistics are required. Its output is added to $\mathcal{Z}_{1:k}$ as measurable evidence for distinguishing anomalies from visually salient but normal variation.

4.3 Coarse-to-Fine Detection Workflow

At each step, the four roles update the diagnostic state using the selected action, toolkit observation, and current interval decisions:

$$s_{k+1} = \mathcal{U}\big(s_k, a_k, o_k, \hat{\mathcal{Y}}^{(k)}\big), \tag{4}$$

where $\mathcal{U}$ is the state-update operator, $\hat{\mathcal{Y}}^{(k)} = \{(\hat{I}_j, \hat{y}_j)\}$ contains interval decisions, and the anomaly set is $\hat{\mathcal{S}} = \{\hat{I}_j \mid \hat{y}_j = 1\}$. Here s_k denotes a workflow state, whereas I_j denotes a temporal interval.

Coarse Evidence Acquisition. The locator renders the input series and uses visual patterns and domain knowledge to propose candidate regions, aiming to retain suspicious intervals, $\mathcal{S}^{(0)} = \mathcal{L}(\mathbf{x}_{1:T}, \mathcal{K}_{vap}, \mathcal{K}_{dom})$. It searches for shape irregularities, abrupt changes, and distributional shifts while retaining uncertain boundaries. This step focuses subsequent diagnostic reasoning on the proposed intervals; it does not by itself establish a reduction in inference cost.

Adaptive Evidence Construction. For the current state, the actor selects $a_k = \mathcal{A}(s_k, \mathcal{K}_{tool})$ and invokes the corresponding toolkit component. The choice can depend on the candidate shape, previous observations, and unresolved uncertainty, allowing different diagnostic paths rather than prescribing a fixed tool sequence. Numerical outputs extend $\mathcal{Z}_{1:k}$, while retrieved knowledge extends $\mathcal{C}_{1:k}$. This is an architectural capability, not a claim that the learned detector optimizes tool selection or interaction cost.

Anomaly Detection. The detection-specific policy integrates each candidate with the accumulated numerical and contextual evidence:

$$(\hat{I}_j, \hat{y}_j) = \mathcal{D}_\theta(I_j, \mathcal{Z}_{1:k}, \mathcal{C}_{1:k}), \qquad \hat{y}_j \in \{0, 1\}. \tag{5}$$

The detector interprets local measurements together with the broader diagnostic context to decide which intervals are anomalous and refine their boundaries. The serialized output lists positive intervals with explanations and ordinal confidence scores in $\{1, 2, 3\}$; an empty list denotes no detected anomaly. This separates coarse localization from boundary-sensitive anomaly judgment.

Iterative Refinement. The evaluator reviews decisions against the accumulated evidence for critical logical errors or unsupported reasoning. Its feedback $f_k = \mathcal{E}(s_k, \hat{\mathcal{Y}}^{(k)})$ either terminates the workflow when `needs_refinement=false` or is incorporated by $\mathcal{U}$ to request further evidence acquisition, boundary adjustment, or reconsideration. Tool use is capped at six invocations per input sample, after which no further tool calls are permitted; this is not a six-turn reasoning limit. The reported evaluator prompt uses a lenient critical-error check; acceptance is not a guarantee of correct anomaly labels or boundaries.

4.4 Hybrid Decision Mechanism

AnomaMind separates flexible workflow control from task-specific anomaly learning. General-purpose LLMs drive visual localization, tool invocation, and evaluation because these stages require instruction following, dynamic control flow, and integration of heterogeneous evidence. Keeping these components training-free also allows tools and domain knowledge to be updated without retraining the complete workflow. The detector $\mathcal{D}_\theta$ instead uses a specialized policy because precise anomaly boundaries require stable task-specific decision signals.

Detection-Specific Policy Learning. We first apply supervised fine-tuning to teach the detector to interpret diagnostic context, follow the output schema, and produce interval-level reasoning traces. Reinforcement learning then uses feedback from complete detector outputs. By default, AnomaMind optimizes only the detection-specific policy, with the locator, actor, and evaluator remaining frozen. Given diagnostic state s_k, the policy outputs $\hat{\mathcal{Y}}^{(k)}$ and receives:

$$r_k = \lambda_p r_k^{parse} + \lambda_f r_k^{F1} - \lambda_{fp} r_k^{FP}. \tag{6}$$

Here, r_k^{parse} rewards valid machine-readable outputs, r_k^{F1} measures agreement with training ground truth, and r_k^{FP} penalizes false positives. We use sample-dependent reward weighting for anomalous and anomaly-free training instances, with full details provided in Appendix C.1. Let $\mathcal{D}_{det}$ denote diagnostic contexts paired with ground-truth labels. The detector-level objective is:

$$\max_\theta \ \mathbb{E}_{(s,\mathcal{Y})\sim\mathcal{D}_{det}} \mathbb{E}_{\hat{\mathcal{Y}}\sim\pi_\theta(\cdot|s)} \left[r(\hat{\mathcal{Y}}, \mathcal{Y}) \right]. \tag{7}$$

The reward is assigned to a complete detector response, not to the full sequence of tool-selection actions. Supervised initialization supplies task alignment, while the reward terms target parsability, localization quality, and false-positive control.

5 Experiments

In this section, we introduce the experimental settings and implementation details, followed by the main results, exploratory analyses, and finally a case study.

5.1 Experimental Settings

5.1.1 Datasets

Table 2 summarizes the statistics and annotation granularity of the datasets used in our experiments. We evaluate our method on four widely used TSAD benchmarks (Liu & Paparrizos, 2024). Yahoo (Laptev et al., 2015) contains real and synthetic time series derived from production traffic, with both point- and segment-level anomalies. KPI and WSD (Zhang et al., 2022) consist of real-world key performance indicators collected from large-scale web services, while IOPS (Xu et al., 2018) contains system-level performance metrics reflecting service scale, quality, and machine health.

Table 2: Statistics of the evaluated time series anomaly detection datasets.

Dataset	All Points	Train Points	Test Points	Anomaly Rate	Category
YAHOO	182,600	107,200	75,400	0.63%	Point & Segment
KPI	1,071,000	483,000	588,000	2.04%	Segment
IOPS	309,000	147,000	162,000	1.50%	Segment
WSD	203,000	89,000	114,000	0.60%	Segment

5.1.2 Baselines

We compare AnomaMind with a broad set of TSAD baselines, including DWT-MLEAD (Thill et al., 2017), FFT-AD[1], MatrixProfile (Yeh et al., 2016), SR (Ren et al., 2019), Sub-PCA (Shyu

[1] https://github.com/TimeEval/TimeEval-algorithms

Table 3: Comparison of anomaly detection performance in terms of F1 and Range-F1 on multiple datasets. **Bold** and underlined values indicate the best and second-best performance, respectively.

Model	YAHOO		KPI		WSD		IOPS	
	F1	Range-F1	F1	Range-F1	F1	Range-F1	F1	Range-F1
DWT_MLEAD	0.268	0.407	0.227	0.360	0.472	0.642	0.388	0.431
FFT	0.361	0.695	0.267	0.373	0.159	0.292	0.172	0.389
SR	0.573	0.759	0.293	0.573	0.296	0.625	0.127	0.542
MatrixProfile	0.459	0.658	0.396	0.529	0.324	0.460	0.369	0.486
Sub_PCA	0.249	0.385	0.219	0.557	0.381	0.589	0.453	0.538
CNN	0.408	0.661	0.214	0.512	0.308	0.651	0.175	0.537
LSTMAD	0.426	0.694	0.161	0.535	0.422	0.664	0.161	0.528
M2N2	0.379	0.578	0.184	0.498	0.216	0.597	0.178	0.493
OmniAnomaly	0.351	0.529	0.400	0.389	0.226	0.448	0.265	0.439
TranAD	0.253	0.445	0.359	0.372	0.199	0.431	0.238	0.420
Chronos	0.698	0.863	0.185	0.493	0.305	0.581	0.149	0.543
TimesFM	0.681	0.896	0.197	0.538	0.355	0.603	0.147	0.354
OFA	0.491	0.753	0.400	0.415	0.294	0.500	0.551	0.652
LLM-AD	0.352	0.362	0.381	0.538	0.293	0.372	0.277	0.533
LLM-TSAD	0.626	0.669	0.437	0.527	0.422	0.470	0.316	0.465
ARGOS	0.484	0.590	0.477	0.548	0.364	0.408	0.301	0.236
NCAD (sup.)	0.453	0.527	0.276	0.448	0.510	0.567	0.281	0.491
Qwen3-8B-SFT	0.526	0.542	0.367	0.453	0.392	0.442	0.283	0.472
Ours (In-domain)	**0.902**	**0.903**	**0.679**	**0.804**	0.697	0.721	0.801	0.840
Ours (Multi-domain)	0.891	0.892	0.606	0.800	**0.805**	**0.842**	**0.855**	**0.909**

Table 4: Ablation study of supervised fine-tuning (SFT) and reinforcement learning (RL).

Model	YAHOO		KPI		WSD		IOPS	
	F1	Range-F1	F1	Range-F1	F1	Range-F1	F1	Range-F1
w/o Both	0.220	0.528	0.065	0.424	0.404	0.558	0.400	0.673
w/o RL	0.859	0.868	0.596	0.796	0.791	0.821	0.801	0.820
w/o SFT	0.607	0.661	0.334	0.673	0.691	0.690	0.728	0.797
Ours	**0.891**	**0.892**	**0.606**	**0.800**	**0.805**	**0.842**	**0.855**	**0.909**

et al., 2003), CNN (Munir et al., 2018), M2N2 (Kim et al., 2024), LSTMAD (Malhotra et al., 2015), TranAD (Tuli et al., 2022), OmniAnomaly (Su et al., 2019), TimesFM (Das et al., 2024), Chronos (Ansari et al., 2024), LLM-TSAD (Park et al., 2025), LLM-AD (Liu et al., 2025a), OFA (Zhou et al., 2023), ARGOS (Gu et al., 2025), and supervised NCAD (Carmona et al., 2022). We further include Qwen3-8B-SFT, a non-agentic supervised baseline trained with the same SFT data as AnomaMind but without tool-augmented reasoning or reinforcement learning, providing a comparison with the full agentic framework. Baseline descriptions and evaluation procedures are provided in Appendices B and C.3, respectively.

5.1.3 Implementation Details

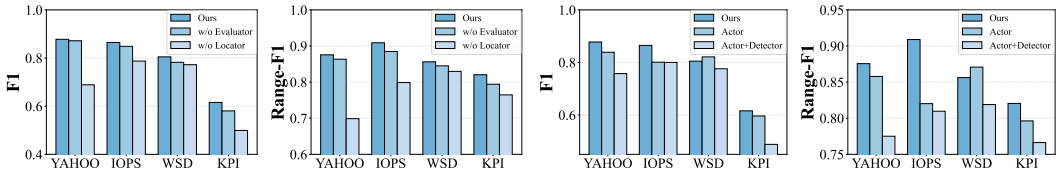
We use Qwen3-8B (Yang et al., 2025b) as the trainable detection policy and Gemini-3.1-flash for frozen general-purpose roles. AnomaMind uses training labels for prototype mining, SFT, and RL rewards; detailed training and inference settings are provided in Appendix C.1. We use official baseline implementations and evaluate all methods on the same test splits. Supervised baselines use the same labeled training splits as AnomaMind and serve as the primary supervision-matched comparisons, while unsupervised and zero-shot methods are included for broader context. The train and test splits are strictly separated throughout preprocessing, prototype construction, model training, and evaluation. We evaluate AnomaMind under in-domain and multi-domain joint-training settings. The former trains a separate model for each dataset, whereas the latter pools the training splits of all four datasets and evaluates the resulting model separately on each test split. Thus, every evaluated domain is represented during training, and this setting should not be interpreted as held-out-domain generalization. Each experiment is repeated three times, and we report mean performance; small differences are interpreted descriptively. We use F1 as the primary metric and Range-F1 for interval localization, with Best-F1 and AUC-PR results provided in Appendix E.1.

5.2 Main Results

As shown in Table 3, AnomaMind achieves competitive detection and interval-localization performance across all four benchmarks under both in-domain and multi-domain joint-training settings. In-domain training performs best on YAHOO and KPI, while multi-domain training achieves the strongest results on WSD and IOPS, indicating dataset-dependent benefits of the two training settings. Moreover, both variants substantially outperform Qwen3-8B-SFT across datasets and metrics, suggesting that the gains extend beyond supervised fine-tuning and benefit from AnomaMind's

Table 5: Ablation study of the different detection tool variants across diverse datasets.

Model	YAHOO		KPI		WSD		IOPS	
	F1	Range-F1	F1	Range-F1	F1	Range-F1	F1	Range-F1
w/o Feature Extraction	0.875	0.871	0.569	0.738	0.698	0.712	0.842	0.820
w/o Value-based Detection	0.863	0.863	0.528	0.697	0.647	0.693	0.796	0.786
w/o Change-based Detection	0.771	0.771	0.587	0.764	0.735	0.724	0.808	0.810
w/o Knowledge Memory	0.845	0.845	**0.610**	0.784	0.720	0.753	0.845	0.827
Ours	**0.891**	**0.892**	0.606	**0.800**	**0.805**	**0.842**	**0.855**	**0.909**



(a) Ablation of key workflow components.

(b) Comparison of trained agent components.

Figure 2: Analysis studies of AnomaMind across F1 and Range-F1 metrics.

evidence-driven, tool-augmented diagnostic process and reinforcement learning. These results support the effectiveness of the proposed hybrid agentic framework across heterogeneous anomaly patterns in diverse real-world detection settings.

5.3 Ablation Studies

5.3.1 Training Strategy Ablation

As shown in Table 4, the full AnomaMind achieves the best reported performance, supporting the combination of SFT and RL. Removing both stages leads to a clear performance drop relative to task-specific optimization. Removing RL also reduces detection performance, while removing SFT causes a larger degradation. These results suggest complementary roles: SFT provides task-aligned initialization, and RL further refines anomaly decisions. Overall, SFT and RL play complementary roles: SFT builds the basic anomaly reasoning capability, while RL further enhances decision quality through feedback-driven optimization.

5.3.2 Detection Tool Ablation

As shown in Table 5, removing individual tool components generally reduces performance, supporting their complementary roles in evidence construction. Change-based and value-based detection assess temporal transitions and magnitude deviations, respectively, while feature extraction supplies statistics and knowledge memory supplies contextual references. The full model has the highest reported Range-F1 on all four datasets, but not the highest F1 in every case: removing knowledge memory slightly improves KPI F1 from 0.606 to 0.610, while reducing its Range-F1 from 0.800 to 0.784. Thus, the benefit of each component depends on the dataset and metric.

5.3.3 Workflow Component Ablation

As shown in Figure 2a, removing either the evaluator or the locator reduces the reported performance. The evaluator ablation supports the usefulness of critical-error review and refinement. The larger degradation without the locator suggests that coarse localization helps identify informative candidate regions for subsequent evidence construction and range-level detection. Overall, these results confirm that the locator and evaluator provide complementary support: the locator improves evidence acquisition, while the evaluator enhances decision reliability.

5.4 Exploratory Analysis

5.4.1 Impact of Different Trained Agent Components

Figure 2b compares the default AnomaMind configuration, which optimizes only the detector, with exploratory variants that additionally train the Actor or both the Actor and Detector. These variants are introduced solely to study the effect of optimizing different agent components and are not part

Table 6: Impact of backbone model size on anomaly detection performance under specialized reinforcement learning.

Model	YAHOO		KPI		WSD		IOPS	
	F1	Range-F1	F1	Range-F1	F1	Range-F1	F1	Range-F1
Qwen3-0.6B	0.827	0.833	0.533	0.737	0.705	0.763	0.806	0.811
Qwen3-1.7B	0.852	0.854	0.523	0.751	0.699	0.758	0.803	0.807
Qwen3-4B	0.870	0.867	0.575	**0.802**	0.728	0.767	0.837	0.895
Qwen3-8B	**0.891**	**0.892**	**0.606**	0.800	**0.805**	**0.842**	**0.855**	**0.909**

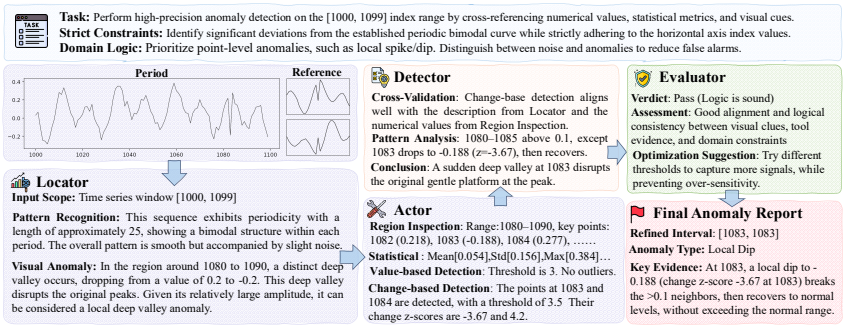


Figure 3: An illustrative example of a structured coarse-to-fine workflow for agentic time series anomaly detection.

of the default framework. The default detector-only configuration achieves the strongest reported overall performance, while jointly training Actor and Detector does not consistently outperform training the Actor alone. These results support the detector-specific optimization design.

5.4.2 Impact of Backbone Size

As shown in Table 6, Qwen3-8B achieves the strongest overall performance among the evaluated model sizes. This suggests that additional model capacity can benefit the learned detection policy under our current setting, although the results do not isolate whether the gains arise from reasoning ability, representation capacity, or optimization behavior. The results also show that performance does not increase monotonically from 0.6B to 1.7B, indicating that moderate scaling alone does not guarantee improvement. However, increasing the backbone size to 4B and 8B leads to more consistent gains, suggesting that sufficient model capacity is important for learning task-specific decision policies through reinforcement learning.

5.5 Case Study

Figure 3 illustrates a coarse-to-fine detection case. For a periodic bimodal time series, the locator identifies a suspicious local valley around indices 1080–1090. The actor then performs tool-assisted inspection: value-based detection finds no clear amplitude outliers, while change-based detection reveals sharp transitions around indices 1083 and 1084. The detector identifies an anomaly at index 1083, and the evaluator checks the consistency between the detected anomaly and the accumulated supporting evidence. Finally, the framework outputs the refined interval $[1083, 1083]$, illustrating precise localization through multi-stage reasoning and tool interaction while avoiding unnecessary alarms on nearby normal observations in this case.

6 Conclusion

In this work, we propose AnomaMind, an agentic TSAD framework that organizes interval-level anomaly detection as an evidence-driven sequential decision-making process. AnomaMind follows a coarse-to-fine workflow that localizes suspicious intervals, constructs diagnostic evidence through tool interaction, and refines decisions through self-reflection. AnomaMind is supported by a toolkit box that integrates knowledge memory and numerical diagnostics. It further adopts a hybrid inference mechanism, where general-purpose models handle flexible reasoning and tool invocation, while a detection-specific policy is optimized with rule-based rewards. Experiments under in-domain and multi-domain joint-training settings support the effectiveness of this design on four benchmarks. Held-out-domain generalization and inference efficiency remain directions for future work.

AI USE STATEMENT

Apart from the language models evaluated in this study, generative AI assistance was limited to manuscript wording, consistency checks between result descriptions and reported tables, and LaTeX formatting. It was not used for research design, experimental implementation, data generation or processing, additional result analysis, or translation. The authors take responsibility for the final content of this work.

REFERENCES

Sarah Alnegheimish, Linh Nguyen, Laure Berti-Equille, and Kalyan Veeramachaneni. Large language models can be zero-shot anomaly detectors for time series? *arXiv preprint arXiv:2405.14755*, 2024.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A Zuluaga. Usad: Unsupervised anomaly detection on multivariate time series. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 3395–3404, 2020.

Amine Bechar, Adel Oulefki, Abbes Amira, Fatih Kurogollu, and Yassine Himeur. Extracting actionable insights from building energy data using vision llms on wavelet and 3d recurrence representations. In *2025 IEEE International Conference on Data Mining (ICDM)*, 2025.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 93–104, 2000.

Chris U. Carmona, François-Xavier Aubet, Valentin Flunkert, and Jan Gasthaus. Neural contextual anomaly detection for time series. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, pp. 2843–2851, 2022. doi: 10.24963/ijcai.2022/394.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

Akash Chandrayan, Amir Zidi, Matthew Reimherr, Anis Mjirda, and Abhinav Pradhan. Lead-framework for efficient time-series anomaly detection on large scale data using llms. In *1st ICML Workshop on Foundation Models for Structured Data*, 2025.

Mingyue Cheng, Xiaoyu Tao, Qi Liu, Ze Guo, and Enhong Chen. Position: Beyond model-centric prediction–agentic time series forecasting. *arXiv preprint arXiv:2602.01776*, 2026.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4027–4035, 2021. doi: 10.1609/aaai.v35i5.16523.

Kamil Faber, Marcin Pietron, Dominik Zurek, and Roberto Corizzo. xlstmad: A powerful xlstm-based method for anomaly detection. In *2025 IEEE International Conference on Data Mining (ICDM)*, pp. 247–256, 2025. doi: 10.1109/ICDM65498.2025.00032.

Mononito Goswami, Cristian Ignacio Challu, Laurent Callot, Lenon Minorics, and Andrey Kan. Unsupervised Model Selection for Time Series Anomaly Detection. In *International Conference on Learning Representations*, 2023.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. MOMENT: A family of open time-series foundation models. In *Forty-first International Conference on Machine Learning*, 2024.

Yile Gu, Yifan Xiong, Jonathan Mace, Yuting Jiang, Yigong Hu, Baris Kasikci, and Peng Cheng. Argos: Agentic time-series anomaly detection with autonomous rule generation via large language models. *arXiv preprint arXiv:2501.14170*, 2025.

Zhaopeng Gu, Bingke Zhu, Guibo Zhu, Yingying Chen, Ming Tang, and Jinqiao Wang. Anomalygpt: Detecting industrial anomalies using large vision-language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 1932–1940, 2024.

Tong Guan, Zijie Meng, Dianqi Li, Shiyu Wang, Chao-Han Huck Yang, Qingsong Wen, Zuozhu Liu, Sabato Siniscalchi, Ming Jin, and Shirui Pan. TimeOmni-1: Incentivizing complex reasoning with time series in large language models. In *International Conference on Learning Representations*, 2026.

Manish Gupta, Jing Gao, Charu C Aggarwal, and Jiawei Han. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and data Engineering*, 26(9):2250–2267, 2013.

Malgorzata Gwiazda, Yifu Cai, Mononito Goswami, Arjun Choudhry, and Artur Dubrawski. TimeSeriesExamAgent: Creating time series reasoning benchmarks at scale. In *International Conference on Learning Representations*, 2026.

Yufei He, Juncheng Liu, Yue Liu, Yibo Li, Tri Cao, Zhiyuan Hu, Xinxing Xu, and Bryan Hooi. EvoTest: Evolutionary test-time learning for self-improving agentic systems. In *International Conference on Learning Representations*, 2026.

Chuang Jiang, Mingyue Cheng, Xiaoyu Tao, Qingyang Mao, Jie Ouyang, and Qi Liu. Tablemind: An autonomous programmatic agent for tool-augmented table reasoning. In *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining*, pp. 260–270, 2026.

Yushan Jiang, Zijie Pan, Xikun Zhang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. Empowering time series analysis with large language models: a survey. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 8095–8103, 2024.

Dongmin Kim, Sunghyun Park, and Jaegul Choo. When model meets new normals: Test-time adaptation for unsupervised time-series anomaly detection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 13113–13121, 2024.

Nikolay Laptev, Saeed Amizadeh, and Ian Flint. Generic and scalable framework for automated time-series anomaly detection. In *Proceedings of the 21st ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1939–1947, 2015.

Boyuan Li, Zhen Liu, Yicheng Luo, and Qianli Ma. Learning recursive multi-scale representations for irregular multivariate time series forecasting. In *International Conference on Learning Representations*, 2026.

Chen Liu, Shibo He, Qihang Zhou, Shizhong Li, and Wenchao Meng. Large language model guided knowledge distillation for time series anomaly detection. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 2162–2170, 2024.

Jun Liu, Chaoyun Zhang, Jiaxu Qian, Minghua Ma, Si Qin, Chetan Bansal, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. Large language models can deliver accurate and interpretable time series anomaly detection. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 4623–4634, 2025a.

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

Yang Liu, Yixing Luo, Xiaofeng Li, Xiaogang Dong, Bin Gu, and Zhi Jin. Evaluating large language models for time series anomaly detection in aerospace software. In *2025 40th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 3522–3533. IEEE, 2025b.

Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, Puneet Agarwal, et al. Long short term memory networks for anomaly detection in time series. In *Proceedings*, volume 89, pp. 94, 2015.

Mohsin Munir, Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. Deepant: A deep learning approach for unsupervised anomaly detection in time series. *Ieee Access*, 7:1991–2005, 2018.

Junwoo Park, Kyudan Jung, Dohyun Lee, Hyuck Lee, Daehoon Gwak, ChaeHun Park, Jaegul Choo, and Jaewoong Cho. Delving into large language models for effective time-series anomaly detection. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-3444.

Xiangfei Qiu, Zhe Li, Wanghui Qiu, Shiyan Hu, Lekui Zhou, Xingjian Wu, Zhengyu Li, Chenjuan Guo, Aoying Zhou, Zhenli Sheng, Jilin Hu, Christian S. Jensen, and Bin Yang. Tab: Unified benchmarking of time series anomaly detection methods. *Proceedings of the VLDB Endowment*, 18:2775–2789, 2025.

Hansheng Ren, Bixiong Xu, Yaoliang Wang, Qi Yi, et al. Time-series anomaly detection service at microsoft. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3009–3017, 2019. doi: 10.1145/3292500.3330680.

Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. Anomaly detection in time series: a comprehensive evaluation. *Proceedings of the VLDB Endowment*, 15(9):1779–1797, 2022.

Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

Lifeng Shen, Weiyu Chen, and James T. Kwok. Multi-resolution diffusion models for time series forecasting. In *International Conference on Learning Representations*, 2024.

Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. A novel anomaly detection scheme based on principal component classifier. In *Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop*, pp. 172–179, 2003.

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 *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2828–2837, 2019.

Xiaoyu Tao, Mingyue Cheng, Ze Guo, Shuo Yu, Yaguo Liu, Qi Liu, and Shijin Wang. Memcast: Memory-driven time series forecasting with experience-conditioned reasoning. *arXiv preprint arXiv:2602.03164*, 2026a.

Xiaoyu Tao, Mingyue Cheng, Chuang Jiang, Tian Gao, Huanjian Zhang, and Yaguo Liu. Cast-r1: Learning tool-augmented sequential decision policies for time series forecasting. *arXiv preprint arXiv:2602.13802*, 2026b.

Markus Thill, Wolfgang Konen, and Thomas Bäck. Time series anomaly detection with discrete wavelet transforms and maximum likelihood estimation. In *Proceedings of the International Conference on Time Series and Forecasting*, volume 2, pp. 11–23, 2017.

Shreshth Tuli, Giuliano Casale, and Nicholas R Jennings. Tranad: deep transformer networks for anomaly detection in multivariate time series data. *Proceedings of the VLDB Endowment*, 15(6): 1201–1214, 2022.

Zexin Wang, Changhua Pei, Yang Liu, Hengyue Jiang, Quan Zhou, Haotian Si, Hang Cui, Jianhui Li, Gaogang Xie, Jingjing Li, et al. Vits: Teaching machines to see time series anomalies like human experts. In *Proceedings of the ACM Web Conference 2026*, pp. 5286–5297, 2026.

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

Xingjian Wu, Junkai Lu, Zhengyu Li, Xiangfei Qiu, Jilin Hu, Chenjuan Guo, Christian S Jensen, and Bin Yang. Timeart: Towards agentic time series reasoning via tool-augmentation. *arXiv preprint arXiv:2601.13653*, 2026.

Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference*, pp. 187–196, 2018.

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

Alan Yang, Yulin Chen, Sean Lee, and Venus Montes. Refining time series anomaly detectors using large language models. *arXiv preprint arXiv:2503.21833*, 2025a.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025b. doi: 10.48550/arXiv.2505.09388.

Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. 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*, pp. 1317–1322. IEEE, 2016.

Shenglin Zhang, Zhenyu Zhong, Dongwen Li, Qiliang Fan, Yongqian Sun, Man Zhu, Yuzhi Zhang, Dan Pei, Jiyan Sun, Yinlong Liu, et al. Efficient kpi anomaly detection through transfer learning for large-scale web services. *IEEE Journal on Selected Areas in Communications*, 40(8):2440–2455, 2022.

Qihang Zhou, Guansong Pang, Yu Tian, Shibo He, and Jiming Chen. AnomalyCLIP: Object-agnostic prompt learning for zero-shot anomaly detection. In *International Conference on Learning Representations*, 2024.

Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36:43322–43355, 2023.

Zihao Zhou and Rose Yu. Can llms understand time series anomalies? In *International Conference on Learning Representations*, volume 2025, pp. 1858–1896, 2025.

A Dataset Details

We evaluate our method on four widely used time series anomaly detection benchmarks covering production traffic, service KPIs, and system-level monitoring measurements, with both point-level and range-level anomaly annotations. Details are provided below:

- **YAHOO**: The Yahoo S5 benchmark contains one real-world subset from Yahoo production traffic and three synthetic subsets. It covers diverse anomaly types, including point, contextual, and collective anomalies, and is widely used for univariate time series anomaly detection.
- **KPI**: The KPI benchmark contains operational metrics from large-scale web services, with segment-level anomaly labels. Its anomalies often appear as short abnormal intervals within noisy and periodic service behavior.
- **IOPS**: The IOPS benchmark contains real-world system-level monitoring metrics from cloud and web services. It includes labeled segment anomalies such as spikes, drops, and drifts, making it suitable for evaluating anomaly detection in complex IT environments.
- **WSD**: The WSD benchmark consists of KPIs from large-scale internet services, including traffic, response time, and error rate metrics. It features production noise, strong periodic patterns, and low anomaly density ($\sim$0.6%), with labeled spikes, contextual shifts, and long-term drifts.

B Baseline Details

To comprehensively evaluate our method, we benchmark against competitive approaches covering statistical and classical machine learning methods, deep learning models, foundation models, and LLM-based anomaly detectors.

B.1 Statistical and Classical Approaches

DWT-MLEAD applies wavelet decomposition to model time series at multiple resolutions and detects anomalies from coefficient- or reconstruction-level deviations. **FFT-AD** decomposes time series into frequency components and identifies anomalies through large high-frequency residuals after low-frequency reconstruction. **MatrixProfile** computes subsequence similarity profiles and flags subsequences with unusually large nearest-neighbor distances. **SR** (Spectral Residual) uses Fourier-based log-amplitude deviations to construct a saliency map for point-level anomaly detection. **Sub-PCA** applies PCA to subsequence representations and detects anomalies from reconstruction deviations from normal subspaces.

B.2 Deep Learning Baselines

CNN employs 1D convolutional layers to extract local temporal features and forecasts future values via fully connected layers, detecting anomalies based on prediction error thresholds. **M2N2** is a test-time adaptation method that handles distribution shifts. It detrends sequences via exponential moving averages and selectively updates parameters on low-error instances through self-supervision. **LSTMAD** utilizes an LSTM-based encoder-decoder architecture to learn latent representations of normal patterns. It identifies anomalies by thresholding reconstruction errors, as deviations yield significantly higher losses. **TranAD** leverages a Transformer encoder-decoder with multi-head self-attention to capture global temporal dependencies, using reconstruction error for anomaly detection. **OmniAnomaly** models stochastic latent representations of multivariate temporal dynamics and detects anomalies through reconstruction likelihoods.

B.3 Foundation Models

TimesFM is a decoder-only foundation model that patches sequences into fixed-length tokens. It autoregressively generates future values, detecting anomalies via point-wise prediction errors.

Chronos is a language-model-based forecaster that tokenizes time series. Operating in a zero-shot setup, it produces probabilistic forecasts via a transformer encoder-decoder, flagging anomalies based on prediction deviations.

B.4 LLM-BASED APPROACHES

LLM-TSAD decomposes time series into trend, seasonal, and residual components, converting them into text descriptions. Using few-shot prompts, the LLM reasons over the decomposed signals to identify anomalies. **LLM-AD** employs LLM reasoning for anomaly detection through textualized temporal observations and task-specific prompts. **OFA** (One Fits All) utilizes a frozen pretrained transformer backbone with lightweight task-specific adapters. It processes sequences through the frozen encoder and uses adapter heads to reconstruct inputs for anomaly identification. **ARGOS** uses collaborative LLM agents to generate executable detection rules, validate their accuracy on labeled data, and repair execution errors (Gu et al., 2025).

B.5 ADDITIONAL SUPERVISED BASELINES

NCAD (sup.) uses the supervised setting of Neural Contextual Anomaly Detection, a framework that learns contextual representations for time-series anomaly detection (Carmona et al., 2022). **Qwen3-8B-SFT** is a non-agentic, supervised-fine-tuned Qwen3-8B baseline that does not use the AnomaMind workflow. In contrast, the *w/o RL* ablation retains the agentic workflow and the SFT-trained detector but omits reinforcement learning. The two settings therefore differ in whether agentic inference is used, rather than representing duplicate SFT-only configurations. These supervised baselines use the same labeled training splits as AnomaMind, providing direct comparisons under the supervised setting, although their model architectures and computational costs differ.

C ADDITIONAL IMPLEMENTATION DETAILS

This section provides additional details on training, inference, data preprocessing, and evaluation.

C.1 TRAINING AND INFERENCE CONFIGURATION

The detection policy is optimized with verl using a learning rate of 5×10^{-6} and a batch size of 16 on four NVIDIA A800 GPUs (80GB).

SFT and RL each use 3,013 training instances constructed exclusively from the labeled training splits. This specifies the training-instance count, not the number of distinct labeled windows or optimization steps. The multi-domain setting uses one training epoch. Unless otherwise specified, ablation and exploratory experiments use multi-domain joint training. The parsing-reward weight is 1 for all samples; the F1 and false-positive-penalty weights are respectively $(0, 1)$ for anomaly-free samples and $(1, 0)$ for anomalous samples, using training labels to select the applicable case.

The default inference temperature is 0.6, except in temperature-sensitivity experiments. Each input sample has a budget of at most six tool invocations, which is distinct from the number of model responses or refinement turns. The Grok configuration in Table 8 uses the API model identifier `grok-4-1-fast`.

C.2 DATA PREPROCESSING

Time series data vary in numerical magnitude and timestamp formats. We apply the following preprocessing steps to standardize inputs for anomaly analysis:

C.2.1 NORMALIZATION

We normalize numerical values to the range $[0, 1]$ via min-max scaling:

$$x' = \frac{x - a}{b - a}$$

where a and b represent the minimum and maximum values of the time series, respectively.

C.2.2 INDEXING

To mitigate the inconsistencies caused by varying timestamp formats and sampling frequencies across datasets, we abstract absolute time into a unified zero-based integer index. Formally, we map the raw time series into an ordered sequence of index-value pairs $\mathcal{X} = \{(t, v_t)\}_{t=0}^{T-1}$, where t represents the relative temporal step and T denotes the sequence length. This standardized representation ensures that the agents focus on relative temporal patterns and output consistent interval coordinates (e.g., $[t_{start}, t_{end}]$) regardless of the original time format.

C.2.3 DETRENDING

To mitigate the influence of global trends that are irrelevant to local anomalies, and improve local pattern sensitivity, we remove the linear component:

$$x' = x - \hat{x}_{\text{trend}}$$

Here, $\hat{x}_{\text{trend}}$ denotes the linear trend estimated via least squares regression.

C.3 EVALUATION METRICS

We use official baseline implementations for modeling and score generation, with the thresholding procedures specified below. Without run-level dispersion, small differences should be interpreted descriptively rather than as statistically significant improvements.

C.3.1 F1-SCORE

The primary goal of time series anomaly detection is to identify anomalous points while minimizing false alarms. We adopt the standard definitions for Recall, Precision, and F1-score:

- **Recall**: Measures the proportion of actual anomalies that are correctly detected. It assesses the model's ability to capture all relevant instances.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **Precision**: Quantifies the reliability of the reported anomalies. It indicates the ratio of true anomalies among all positive predictions, reflecting the model's ability to avoid false alarms during practical anomaly detection scenarios.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- **F1-score**: Provides a balanced metric of overall detection performance. It is calculated as the harmonic mean of Precision and Recall, making it suitable for scenarios with imbalanced data in practical anomaly detection settings.

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

For traditional models that output continuous anomaly scores, the reported F1 uses a threshold of $\mu + 3\sigma$, where μ and σ are estimated from anomaly scores on the training split. Points exceeding this threshold are classified as anomalous. This shared scoring protocol should be distinguished from each model's original default thresholding strategy. For models that directly generate anomaly intervals, points inside the reported intervals are classified as anomalous. To assess robustness beyond a single operating threshold, we additionally report Best-F1 and AUC-PR in Table 7; Best-F1 is used only as an oracle diagnostic.

C.3.2 BEST-F1

- **Traditional methods**: Best-F1 represents the maximum F1-score obtained by sweeping over possible anomaly score thresholds.

Algorithm 1 Anomaly Pattern Mining

Require: Labeled training series $\mathcal{D}_{tr}$, number of clusters per length group H
Ensure: Visual anomaly patterns $\mathcal{K}_{vap}$

```
Extract anomalous segments U = {u_i}_{i=1}^{Q} from D_tr
Partition U into {U^short, U^medium, U^long} by segment length
for each segment u_i ∈ U do
    Compute descriptor φ(u_i) ∈ R^18
end for
Initialize K_vap ← ∅
for each length group g ∈ {short, medium, long} do
    Cluster {φ(u_i) | u_i ∈ U^g} into {C_{g,h}}_{h=1}^{H} by KMeans
    for each cluster C_{g,h} do
        Select u*_{g,h} = arg min_{u_i ∈ C_{g,h}} ||φ(u_i) − μ_{g,h}||_2
        Add u*_{g,h} to K_vap
    end for
end for

return K_vap
```

- **Our method**: We retain predicted intervals whose confidence meets a threshold and report the maximum F1 across the resulting predictions. The detector emits ordinal confidence scores in $\{1, 2, 3\}$, so this sweep has only a few distinct operating points; it is not a sweep over calibrated continuous probabilities.
- **Other LLM-based methods**: We adhere to the Best-F1 calculation procedures specified in their respective implementations.

Best-F1 selects the best threshold using evaluation labels and is therefore an oracle diagnostic, not a deployable threshold-selection result.

C.3.3 AUC-PR AND RANGE-F1

AUC-PR summarizes the precision-recall curve induced by anomaly-score thresholds. For AnomaMind, the ranking uses interval confidence rather than continuous point-wise model scores. Discrete scores permit a precision-recall curve, but introduce ties and fewer operating points; these values should not be interpreted as evidence of calibrated confidence. Complete Best-F1 and AUC-PR results are reported in Table 7, while the main text focuses on F1 and Range-F1.

Range-F1. We adopt the R-based-F1 metric implemented in the TSB-AD benchmark (Liu & Paparrizos, 2024), and refer to it as Range-F1 throughout this paper. It evaluates range-level anomaly detection by accounting for the overlap between predicted and ground-truth anomalous intervals.

D ADDITIONAL DETAILS OF TOOLKIT BOX CONSTRUCTION

This section provides additional details on the construction of the toolkit box, with a focus on the knowledge components used to support anomaly reasoning. We first summarize representative anomaly types that organize domain-level anomaly knowledge, and then describe the mining procedure used to extract visual anomaly prototypes from training data. Together, these components provide reusable prior knowledge for subsequent diagnostic reasoning and tool interaction.

D.1 REPRESENTATIVE ANOMALY TYPES

Figure 4 summarizes the anomaly-type names, descriptions, and representative examples used to organize the domain knowledge in AnomaMind. These categories cover common local, global, burst, plateau, and level-shift patterns, providing structured semantic references for interpreting different forms of anomalous behavior during subsequent diagnostic reasoning.

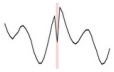
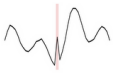
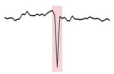
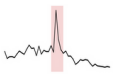
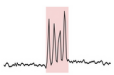
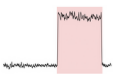
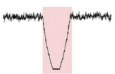
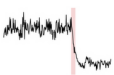
Anomaly Type	Description	Example
Local Dip	A brief, sharp dip at one or a few points that breaks the local pattern, but stays within the normal overall range.	
Local Spike	A brief, sharp spike at one or a few points that breaks the local pattern, but stays within the normal overall range.	
Global Dip	A sharp downward dip that falls far below the rest of the segment, is an extreme outlier on the global scale, and then returns to baseline.	
Global Spike	A sharp upward spike that rises far above the rest of the segment, is an extreme outlier on the global scale, and then returns to baseline.	
Spike Burst	Several sharp upward spikes occur close together in a short interval, while the series before and after stays near a quiet baseline.	
Transient Plateau Up	The series jumps to a higher level, stays on a plateau for a while, then returns to the original baseline.	
Transient Plateau Down	The series drops to a lower level, stays on a plateau for a while, then returns to the original baseline.	
Level Shift Down	A sustained downward shift in the series to a new, lower mean level, without reverting to the previous baseline.	

Figure 4: Representative time-series anomaly types.

D.2 Anomaly Pattern Mining

Algorithm 1 constructs the visual anomaly prototypes stored in the knowledge memory from training data. It first extracts all annotated anomalous segments and partitions them by duration into short, medium, and long groups, allowing patterns at different temporal scales to be modeled separately. Each anomalous segment is then represented by an 18-dimensional descriptor $\phi(u_i)$, which summarizes its characteristic pattern for subsequent comparison. Within each length group, the descriptors are clustered into H groups using KMeans to capture recurring anomaly structures. For each cluster, the segment whose descriptor is closest to the corresponding cluster centroid is selected as the representative prototype and added to $\mathcal{K}_{vap}$. This procedure compresses the diverse anomaly patterns observed in the training data into a compact set of representative examples that can be reused as visual evidence during subsequent anomaly reasoning.

E Extended Analysis and Experimental Results

In this section, we provide additional analyses to further validate the effectiveness of AnomaMind. We report training convergence and sampling-temperature sensitivity, present qualitative visualization comparisons against baseline methods, and provide the prompts used in AnomaMind to clarify the reasoning and detection process.

E.1 Main Results

Table 7 reports F1, Best-F1, AUC-PR, and Range-F1, including supervised NCAD and Qwen3-8B-SFT. The in-domain AnomaMind variant performs best on YAHOO and KPI, while multi-domain joint training performs best on WSD and IOPS. The effect of joint training is not uniformly positive: for example, KPI AUC-PR decreases from 0.519 to 0.403, below OmniAnomaly's 0.417. These results support the effectiveness of the framework on the evaluated domains while showing dataset- and metric-dependent trade-offs. Because all four domains contribute training data, the compari-

Table 7: Complete comparison of anomaly detection performance on multiple benchmark datasets. **Bold** and underlined values indicate the best and second-best performance, respectively.

Model	YAHOO				KPI				WSD				IOPS			
	F1	Best-F1	AUC-PR	Range-F1	F1	Best-F1	AUC-PR	Range-F1	F1	Best-F1	AUC-PR	Range-F1	F1	Best-F1	AUC-PR	Range-F1
DWT_MLEAD	0.268	0.314	0.205	0.407	0.227	0.306	0.181	0.360	0.472	0.614	0.534	0.642	0.388	0.451	0.413	0.431
FFT	0.361	0.689	0.623	0.695	0.267	0.305	0.194	0.373	0.159	0.230	0.186	0.292	0.172	0.287	0.181	0.389
SR	0.573	0.727	0.657	0.759	0.293	0.316	0.273	0.573	0.296	0.539	0.461	0.625	0.127	0.419	0.290	0.542
MatrixProfile	0.459	0.641	0.542	0.658	0.396	0.407	0.364	0.529	0.324	0.468	0.347	0.460	0.369	0.453	0.381	0.486
Sub_PCA	0.249	0.364	0.219	0.385	0.219	0.475	0.376	0.557	0.381	0.475	0.416	0.589	0.453	0.523	0.492	0.538
CNN	0.408	0.620	0.573	0.661	0.214	0.313	0.254	0.512	0.308	0.528	0.432	0.651	0.175	0.415	0.323	0.537
LSTMAD	0.426	0.642	0.595	0.694	0.161	0.287	0.227	0.535	0.422	0.609	0.520	0.664	0.161	0.425	0.340	0.528
M2N2	0.379	0.523	0.476	0.578	0.184	0.316	0.255	0.498	0.216	0.483	0.402	0.597	0.178	0.496	0.384	0.493
OmniAnomaly	0.351	0.515	0.467	0.529	0.400	0.471	0.417	0.389	0.226	0.334	0.298	0.448	0.265	0.386	0.315	0.439
TranAD	0.253	0.406	0.355	0.445	0.359	0.386	0.264	0.372	0.199	0.300	0.217	0.431	0.238	0.436	0.352	0.420
Chronos	0.698	0.789	0.782	0.863	0.185	0.284	0.220	0.493	0.305	0.515	0.410	0.581	0.149	0.488	0.375	0.543
TimesFM	0.681	0.849	0.826	0.896	0.197	0.312	0.245	0.538	0.355	0.546	0.429	0.603	0.147	0.435	0.350	0.354
OFA	0.491	0.723	0.684	0.753	0.400	0.407	0.375	0.415	0.294	0.464	0.384	0.500	0.551	0.567	0.450	0.652
LLM-AD	0.352	0.371	0.220	0.362	0.381	0.417	0.327	0.538	0.293	0.294	0.193	0.372	0.277	0.354	0.256	0.533
LLM-TSAD	0.626	0.626	0.553	0.669	0.437	0.437	0.258	0.527	0.422	0.422	0.289	0.470	0.316	0.316	0.153	0.465
ARGOS	0.484	0.484	0.509	0.590	0.477	0.477	0.288	0.548	0.364	0.364	0.358	0.408	0.301	0.301	0.267	0.236
NCAD (sup.)	0.453	0.534	0.510	0.527	0.276	0.323	0.251	0.448	0.510	0.572	0.497	0.567	0.281	0.299	0.269	0.491
Qwen3-8B-SFT	0.526	0.526	0.497	0.542	0.367	0.367	0.287	0.453	0.392	0.392	0.288	0.442	0.283	0.283	0.247	0.472
Ours (In-domain)	**0.902**	**0.909**	**0.880**	**0.903**	**0.679**	**0.682**	**0.519**	**0.804**	0.697	0.702	0.587	0.721	0.801	0.814	0.672	0.840
Ours (Multi-domain)	0.891	0.896	0.856	0.892	0.606	0.618	0.403	0.800	**0.805**	**0.816**	**0.730**	**0.842**	**0.855**	**0.865**	**0.767**	**0.909**

Table 8: Experimental results with different general-purpose model variants under the proposed framework across diverse anomaly detection settings.

Model	YAHOO		KPI		WSD		IOPS	
	F1	Range-F1	F1	Range-F1	F1	Range-F1	F1	Range-F1
Gemini-3.1-flash	**0.891**	**0.892**	**0.606**	**0.800**	0.805	0.842	0.855	**0.909**
grok-4-1-fast	0.848	0.860	0.424	0.749	**0.836**	**0.856**	**0.871**	**0.909**
GPT-5-mini	0.774	0.853	0.447	0.691	0.766	0.840	0.740	0.796

son does not test unseen-domain transfer. Best-F1 is an oracle threshold diagnostic, and AUC-PR comparisons should account for the different score resolutions described above.

E.2 Impact of General-Purpose Models

As shown in Table 8, the proposed framework maintains competitive performance across the evaluated general-purpose model variants, suggesting that it can work with different backbones. Gemini-3.1-flash achieves stronger results on YAHOO and KPI, while `grok-4-1-fast` performs better on WSD and IOPS, indicating dataset-dependent performance across the evaluated models. GPT-5-mini remains effective but achieves lower overall performance under these settings.

E.2.1 Impact of Training Convergence

Figure 5 shows the displayed training-reward trajectories. SFT+RL starts with higher rewards than pure RL, consistent with a benefit from supervised initialization. Reward trends also vary across backbone scales. These curves describe the shown training runs; without run-level uncertainty and smoothing details, they do not establish optimization stability or efficiency across random seeds.

E.2.2 Sampling Temperature Sensitivity

Figure 6 reports F1 and Range-F1 at the sampled temperatures. Performance varies with temperature and dataset, with generally smaller changes in Range-F1 than in point-wise F1. The results characterize the tested settings, rather than establishing temperature-insensitive behavior across agents, model versions, or random seeds.

E.3 Visualization Case

Figure 7 compares detection outputs on one representative window. FFT, TimesFM, M2N2, LLM-TSAD, Sub-PCA, and OmniAnomaly detect both anomalous regions but also flag normal points. TranAD, ARGOS, CNN, OFA, and LSTMAD miss the second anomalous plateau, while MatrixProfile and DWT-MLEAD only partially localize it. AnomaMind identifies both anomalous intervals with no false positives in this displayed window. This is a qualitative illustration, not a dataset-wide claim of zero false positives; aggregate comparisons are given in Table 7.

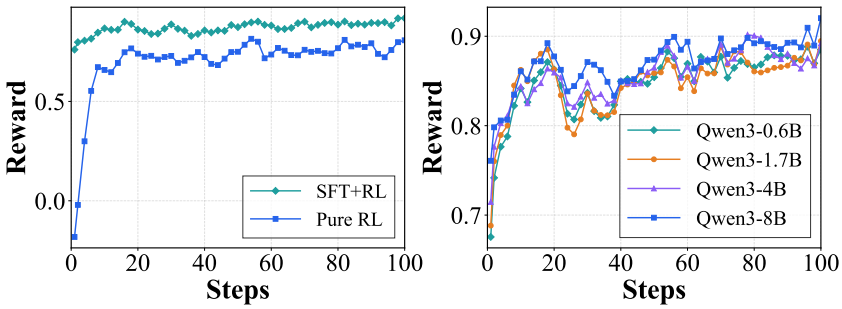


Figure 5: Training-reward trajectories for the displayed optimization strategies and backbone scales. Run-level variability is not shown.

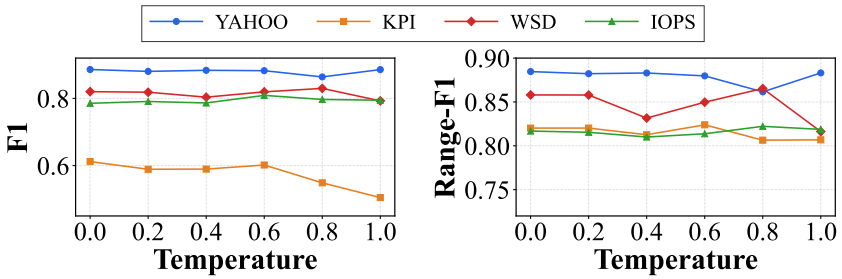


Figure 6: Sensitivity analysis of sampling temperature on anomaly detection performance across F1 and Range-F1 metrics.

E.4 Prompt Templates

We provide the role-specific prompt templates used in the reported workflow. The actor prompt encourages broad evidence collection and does not impose a cost-minimizing selection rule. The evaluator uses a lenient critical-error check, so its acceptance should not be interpreted as independent validation against ground truth.

Locator Prompt

Role: You are an anomaly detection expert.
Input Context: You will be given a time series line chart and relevant typical anomaly pattern diagrams. You need to analyze whether the line chart contains any anomalous intervals and compare them with the patterns:
- **Picture:** The time series picture, with anomaly pattern visual references.{`Picture`}.
- **Knowledge:** Domain-specific constraints and patterns provided in {`Domain Knowledge`}.

Guidelines: Adhere to the following rules:
- **Completeness:** Do not miss any suspicious intervals. Include as many suspicious intervals as possible.
- **Constraint:** Anomaly intervals should be identified based on domain-specific typical anomaly patterns and domain knowledge.

Output Protocol: Structure your response in three strict parts:
- `<Visual Description>`: Your visual summary of the line chart, including information on shape, value fluctuation range, and periodicity.
- `<Intervals>`: The Candidate Intervals detected from picture.
- `<Rationale>`: Explanation of why each candidate interval is considered an anomaly.

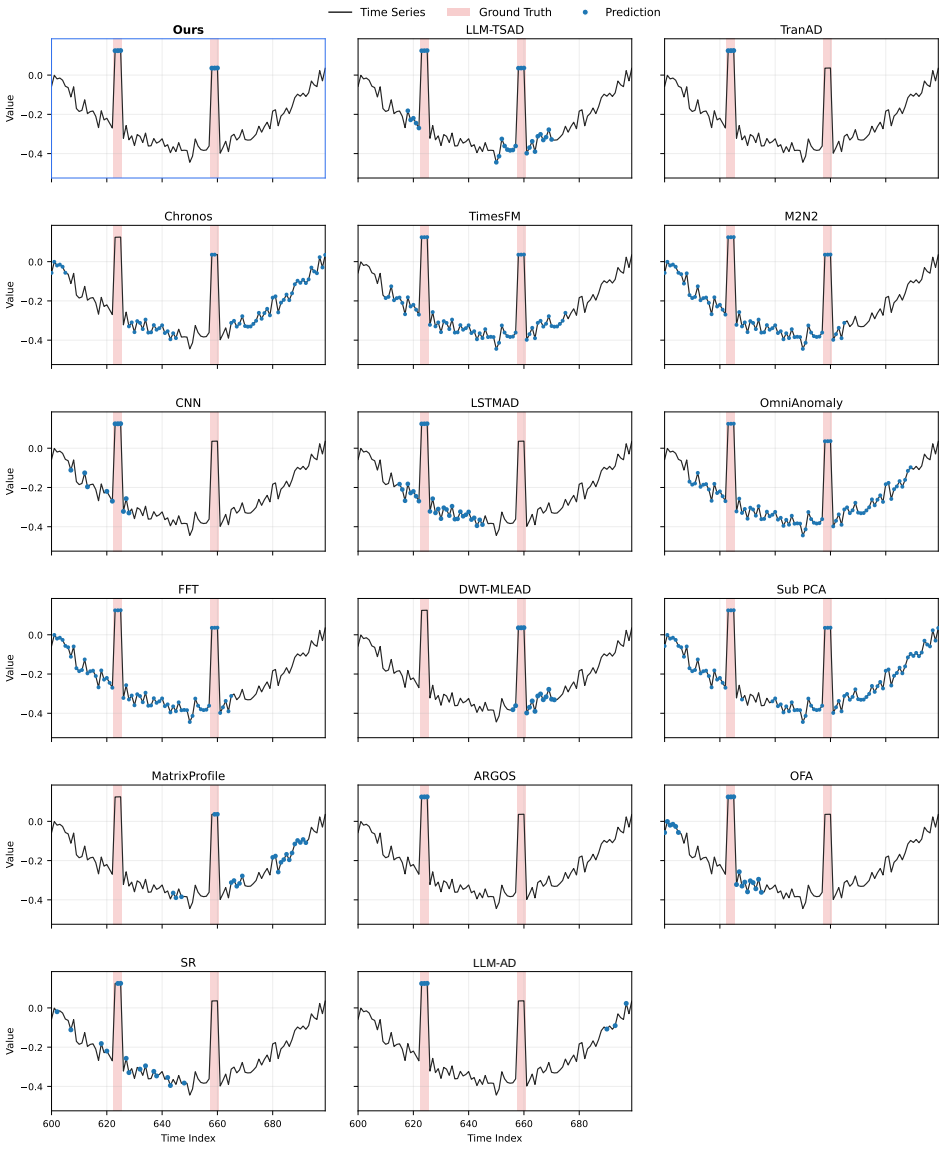


Figure 7: Qualitative comparison of detection results on one representative time-series window.

Actor Prompt

Role: You are a time series anomaly detection expert responsible for tool execution. Your primary function is to select and invoke appropriate tools based on coarse candidate intervals and domain knowledge.
Input Context:
- **Knowledge:** The description and rules of dataset. {`Domain Knowledge`}.
- **Tool Description:** The explanation, parameters, expected output format, and usage constraints for each available tool. `Tool Description`. You must strictly follow the input-output specifications defined here. Do not attempt to use a tool in a way that violates its description.
- **Candidate Intervals:** The suspicious intervals from the locator {`Candidate Intervals`}.

Execution Protocol: Adhere to the following strict rules:
- **Evidence-Driven Stopping Criterion:** Collect sufficient and relevant tool evidence to resolve the uncertainty of the current candidate intervals. Stop calling tools once the evidence is sufficient to determine all anomaly locations. Do not stop early unless every candidate interval has been sufficiently supported.
- **Batch Execution:** Efficiency is paramount. Try to invoke more independent tools in fewer turns.
- **Compliance:** Do not deviate from the domain rules; focus solely on execution logic.

Detector Prompt

Role: You are the core anomaly detector. Your task is to synthesize tool execution results, the suspicious intervals, and domain knowledge to render a final, reasoned verdict.
Input Context:
- **Execution History:** The results from previous tool calls ({`Tool_Interaction`}).
- **Knowledge:** Domain-specific rules for fine-grained reasoning ({`Domain Knowledge`}).

Response Structure: Your output must strictly follow this format:
- `<think>`: A concise analysis of how the tool results support or refute the anomaly hypothesis.
- `JSON Array`: A list of detected anomalies (or an empty array `[]` if none). Each object must include:
  - `"interval": [start_index, end_index]` (Use actual data indices).
  - `"type"`: The specific name of the anomaly type.
  - `"explanation"`: Concrete reasoning for the decision.
  - `"confidence"`: An integer score from 1 (Low) to 3 (High).

Critical Constraints:
- **Evidence-Based:** Decisions must be grounded in the provided tool results and data values.
- **Index Fidelity:** Strictly adhere to the original data index values.

Evaluator Prompt

Role: You are the final reviewer responsible for quality assurance. Your task is to evaluate the detection results for logical consistency and critical errors.
Input Context:
- **Tool Interaction:** The tool calls and tool results in Actor ({`Tool Interaction`}).
- **Result:** The detection outcome and reasoning provided by the Detector ({`Detector Result`}).

Evaluation Criteria: Adopt a **lenient** evaluation stance. Intervention is required *only* for critical failures.
- **Pass Condition:** If the analysis is generally reasonable, set `needs_refinement` to `false`.
- **Fail Condition:** Trigger refinement (`true`) *only* for major errors (e.g., critical logic flaws, hallucinated data, or complete absence of reasoning).
- **Rating Scale:** Use ”poor” strictly for severe issues. Default to ”good” or ”acceptable” for minor imperfections.

Output Protocol: Return a strict JSON object containing the following keys:
- `"issues"`: A list of strings describing critical errors (if any).
- `"suggestions"`: Actionable advice for correction.
- `"needs_refinement"`: Boolean (`true`/`false`).
- `"quality_metrics"`: Ratings for `"tool_usage"` and `"reasoning"` (Value: ”good”/”acceptable”/”poor”).