BRIDGING THE MODALITY GAP BETWEEN TIME SERIES AND LANGUAGE FOR COGNITIVE REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series analysis has recently been coupled with Large Language Models (LLMs) to leverage their reasoning and world knowledge capabilities, yet gains remain limited. We attribute this to a fundamental mismatch between existing task formulations and LLM strengths: most settings reduce time series understanding to curve-fitting systems, focusing on low-level prediction while ignoring the semantic, contextual, and reasoning-intensive nature of real-world temporal decision-making. To address these limitations, we introduce ***TSCognition***, a multimodal benchmark for multi-dimensional time series reasoning. It collects real-world time series and textual information from ***15*** public sources and constructs ***∼41K*** QA samples around five cognitive reasoning tasks: Decoding, Grounding, Inferring, Extrapolating, and Acting. Building on this, we further propose ***TSAlign***, a unified framework that encodes time series into compact patch-level representations and aligns them with semantic directions in the LLM embedding space via gated residual injection and multivariate fusion. Experiments show that TSAlign ***outperforms*** existing LLM, VLM, and time series QA baselines on our TSCognition and the publicly available TimerBed while substantially reduce computational cost. **Code** and **Datasets** are available at the Anonymous Repository.

1 INTRODUCTION

Time series analysis models temporal dependencies in sequential data to uncover latent patterns and forecast future behavior, playing a critical role across diverse real-world domains (Brown et al., 2020; Tong et al., 2022; Qiu et al., 2025; Jiang et al., 2025b). Recognizing the profound success of Large Language Models (LLMs), researchers have increasingly sought to integrate them into time series analysis, hoping to leverage their vast pre-trained knowledge to enhance scenario understanding (Kong et al., 2025; Wang et al., 2025; Qiu et al., 2026).

However, despite efforts across tasks such as forecasting, classification, and anomaly detection (Liu et al., 2025a; Jiang et al., 2025a; Jia et al., 2026), introducing LLMs to time series domains has yielded limited improvements. Emerging evidence suggests this bottleneck stems from a fundamental mismatch between current task formulations and the intrinsic capabilities of LLMs (Tan et al., 2024; Xie et al., 2025; Zheng et al., 2026). Specifically, traditional tasks mainly focus on shallow pattern mining, such as predicting future points, imputing missing values, or detecting anomalies. In practice, however, real-world time series are inherently stochastic and heavily influenced by unobserved external factors, making precise trajectory prediction fundamentally ill-posed. Ultimately, these setups reduce LLMs to sophisticated curve-fitters, while their true strengths of extensive world knowledge, contextual understanding, and complex reasoning remain completely unutilized (Liu et al., 2024; Wu et al., 2025).

To better align time series analysis with LLM capabilities, recent work advocates a shift from isolated prediction toward language-conditioned temporal reasoning (Chen et al., 2025; Yu et al., 2026; Guan et al., 2026; Wu et al., 2026). This paradigm integrates temporal signals with natural language queries and external context, requiring models to perform evidence extraction, knowledge grounding, and logical inference over time-evolving systems. However, this transition is fundamentally constrained by the lack of suitable data. Most existing datasets remain designed for forecasting or anomaly detection (Wu et al., 2025; Guan et al., 2026; Yu et al., 2026), while early time series QA benchmarks such as TimeOmni-1 (Guan et al., 2025) and Time-MQA (Kong et al., 2025) short

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.

univariate time series, shallow questions, and narrow domains, lacking semantic depth and compositional reasoning, as shown in Figure 1. This limits their ability to evaluate higher-order cognition.

To address this gap, we introduce ***TSCognition***, a multimodal dataset designed for multi-dimensional time series reasoning. Constructed from 15 public sources, it pairs real-world time series with rich contextual text and contains 41,086 samples spanning five reasoning tasks. Compared to prior datasets, TSCognition evaluates finer-grained temporal cognition, covering pattern decoding, context grounding, relational inference, future extrapolation, and action decision.

Beyond the data bottleneck, existing methodologies for integrating time series with LLMs face severe representational limitations. Existing methods usually convert time series into text, images, or projected features, which may fragment temporal structures, distort numerical dynamics, and lack language-space alignment (Nie et al., 2022; Liu et al., 2025b; Nagrath & Panigrahy, 2026). To overcome this representational flaw and efficiently bridge continuous time series representations with the discrete language reasoning space, we propose ***TSAlign***. TSAlign is motivated by a simple intuition: time series tokens should preserve temporal dynamics while remaining understandable to the LLM's pretrained language space. It learns compact patch-level representations and explicitly aligns them with principal semantic directions in the LLM embedding space. A gated residual mechanism then injects semantic guidance without overwriting the original temporal information, reducing the information loss caused by direct feature replacement. Compared with directly feeding numerical values into LLMs, our method substantially reduces the modality gap.

Extensive experiments show that TSAlign achieves consistent advantages in both QA reasoning and traditional pattern analysis. On TSCognition, TSAlign outperforms existing LLM, VLM, and time series QA baselines while drastically reducing token overhead. It uses ***3.0***$\times$ fewer tokens than vision-based inputs and ***16.8***$\times$ fewer than textual inputs. Furthermore, on the established TimerBed benchmark, TSAlign achieves state-of-the-art performance in pattern analysis, proving that *when properly aligned and tasked, LLMs significantly outperform traditional time series models*.

In summary, our main contributions are as follows:

1. *A Cognitively Structured Dataset:* We curate ***TSCognition***, integrating real-world time series with contextual text to evaluate LLMs across five hierarchical reasoning tasks, moving beyond shallow pattern mining.
2. *A Unified Alignment Framework:* We propose ***TSAlign***, which bridges time series and language reasoning spaces through semantic alignment, gated residual preservation, and multi-dimensional fusion, avoiding the pitfalls of numerical serialization.
3. *State-of-the-Art Performance & Efficiency:* We demonstrate that TSAlign delivers superior reasoning accuracy and highly efficient token utilization on both the novel TSCognition dataset and the public TimerBed benchmark.

2 Related Work

Language Models for Time Series QA. Existing attempts to connect language with time series data have mostly used text as a supplementary signal for conventional tasks, such as forecasting and classification, while its role as a core component of temporal reasoning remains underexplored (Jin et al., 2024; Liu et al., 2025c; Meunier et al., 2025). In many cases, these methods are built upon handcrafted prompts or overly simplified experimental settings, which weakens their applicability to realistic scenarios. Existing language-based time series studies mostly focus on surface descriptions or basic pattern interpretation (Guan et al., 2025; Wu et al., 2025; Chen et al., 2025; Yu et al., 2026). Moreover, textualizing time series for LLMs is computationally expensive and poorly suited to real-time, multivariate, or long-horizon analysis (Luo et al., 2025; Cheng et al., 2026). More fundamentally, such formulations overlook the intrinsic modality gap between time series and natural language, thereby limiting the model's ability to capture the underlying temporal structures effectively (Schumacher et al., 2026).

Time Series–Language Modality Gap. A key challenge in integrating time series with LLMs lies in the substantial modality gap between continuous numerical signals and discrete language representations (Jiang et al., 2025b; Yu et al., 2026). Some existing methods serialize time series values into text and directly feed them into LLMs (Guan et al., 2025; Liu et al., 2025b; Kong et al., 2025).

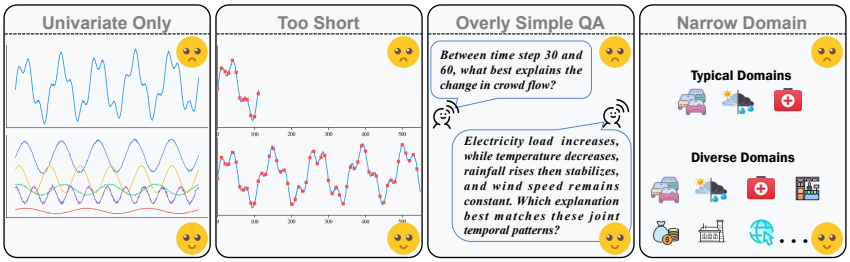


Figure 1: Limitations of existing time series QA datasets (upper) vs. our TSCognition (bottom). Our TSCognition introduces compositional reasoning over realistic real-world scenarios, better testing whether models truly understand over temporal patterns rather than relying on superficial cues.

This strategy can disrupt local continuity and numerical relationships while introducing considerable token overhead (Nie et al., 2022; Wang et al., 2025; Liu et al., 2025b). More importantly, numerical text still differs substantially from the pretrained language space of LLMs, limiting effective reuse of their prior knowledge (Wang et al., 2025). A more suitable approach is to first learn compact representations that preserve temporal dynamics and then align them with the LLM representation space (Zhou et al., 2023; Zhong et al., 2025; Liu et al., 2025c). This can reduce information distortion, input redundancy, and the modality gap between time series and language.

3 Background

3.1 Problem Formulation

Given a multivariate time series $X = \{x_1, x_2, \ldots, x_T\}, x_t \in \mathbb{R}^d$, where T denotes the sequence length and d denotes the variable dimension. In this context, evaluations are instantiated in two forms: ***QA reasoning tasks*** and ***pattern analysis tasks***.

For **QA reasoning tasks**, let the natural language query be denoted as $q \in \mathcal{Q}$, where q may contain task instructions, contextual information, background descriptions, or reasoning objectives. The model is required to generate an answer $a \in \mathcal{A}$ conditioned on both the time series and the query. This can be formulated as $f_\theta : (X, q) \to a$, or equivalently as the conditional distribution $p_\theta(a \mid X, q)$. To characterize this process, we introduce a latent reasoning state z, which represents the intermediate understanding induced jointly by temporal evidence and query semantics: $p_\theta(a \mid X, q) = \int p_\theta(a \mid z, q)\, p_\theta(z \mid X, q)\, dz$. $p_\theta(z \mid X, q)$ models the construction of a latent reasoning state from the time series and the query, while $p_\theta(a \mid z, q)$ generates the final answer conditioned on this state. The training objective is:

$$\min_\theta \ \mathbb{E}_{(X,q,a)\sim\mathcal{D}_{\text{QA}}} \left[-\log p_\theta(a \mid X, q)\right]. \tag{1}$$

For **pattern analysis tasks**, the model is expected to directly identify and analyze temporal patterns from the raw time series, without relying on an explicit natural language query. Let $y \in \mathcal{Y}$ denote the pattern label or analysis target associated with X. The task can be formulated as $f_\theta : X \to y$, or probabilistically as $p_\theta(y \mid X)$. Similarly, we introduce a latent pattern representation z to denote the structured temporal information extracted from the input sequence: $p_\theta(y \mid X) = \int p_\theta(y \mid z)\, p_\theta(z \mid X)\, dz$. $p_\theta(z \mid X)$ extracts task-relevant temporal structures from the raw time series, while $p_\theta(y \mid z)$ maps the inferred pattern representation to the final analysis target. The training objective is:

$$\min_\theta \ \mathbb{E}_{(X,y)\sim\mathcal{D}_{\text{PA}}} \left[-\log p_\theta(y \mid X)\right]. \tag{2}$$

Therefore, pattern analysis evaluates the extraction of discriminative temporal structures (Yue et al., 2022; Qiao et al., 2026), while QA reasoning evaluates their alignment with language semantics and task objectives (Chen et al., 2025; Jing et al., 2026).

3.2 Revisiting the Limitations of TS Reasoning Dataset

Despite recent efforts to incorporate natural language into time series analysis, existing time series reasoning settings remain fundamentally limited (Kong et al., 2025; Wang et al., 2025; Chen et al.,

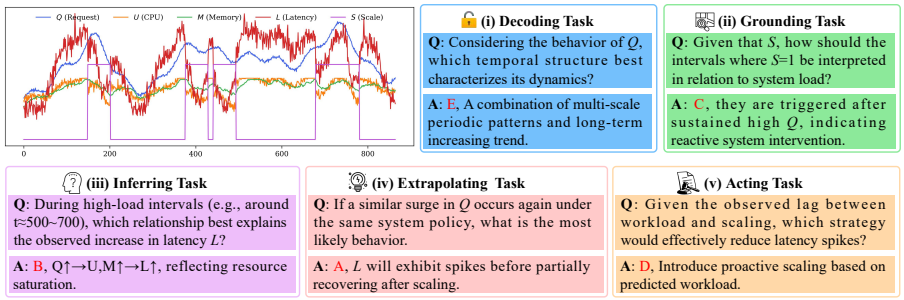


Figure 2: Using a cloud workload scenario as an example, we construct five hierarchical tasks based on time series (Q, U, M, L, S): Decoding, Grounding, Inferring, Extrapolating, and Acting.

2025; Guan et al., 2025). We analyze these issues in Figure 1. ***(1) Most existing approaches are primarily built upon univariate time series, where each sample contains only a single variable evolving over time. (2) The time series used in existing datasets are typically short, often limited to around 50–125 time steps. (3) The design of existing time series QA tasks is often overly simplistic and surface-level. (4) The range of application domains covered by existing benchmarks remains relatively narrow.*** **In summary**, existing time series reasoning tasks remain limited in structure, length, complexity, and domain coverage. More cases are provided in Appendix G.1.

3.3 TSCognition: A Multimodal Dataset for Multi-Dimensional TS Reasoning

To address these limitations, we introduce ***TSCognition***. It enables more realistic and rigorous evaluation of model generalization and reasoning capabilities. *To the best of our knowledge, this is the first multimodal dataset for time series that unifies multi-dimensional data, long-horizon dynamics, and cognitively structured reasoning tasks.* A comparison with benchmarks is provided in Table 1.

Table 1: Comparison of time series reasoning benchmarks.

Benchmark	Multivariate Signals	Long Temporal Context	Context Grounding	Multi-stage Reasoning	Action
Time-MQA	–	✓	✓	✓	–
MTBench	–	✓	✓	✓	–
EngineMT-QA	✓	✓	–	✓	✓
TSAQA	–	✓	–	✓	–
TSRBench	✓	–	–	✓	✓
SciTS	✓	✓	–	✓	–
TSCognition	✓	✓	✓	✓	✓

We collect raw data from 15 public sources. TSCognition covers 8 domains, including industry, web, finance, climate, transportation, healthcare, retail, and energy. To characterize the cognitive structure of time series reasoning, we decompose it into five hierarchical tasks: ***Decoding***, ***Grounding***, ***Inferring***, ***Extrapolating***, and ***Acting***, as shown in Figure 2. ***(i) Decoding task*** extracts fundamental temporal patterns from raw series, including trends, periodicity, and local structures, providing the basis for subsequent reasoning. ***(ii) Grounding task*** aligns time series with contextual information or task descriptions, enabling context-aware interpretation rather than isolated analysis. ***(iii) Inferring task*** performs relational modeling and multi-step reasoning, capturing dependencies, causal cues, and conditional relationships. ***(iv) Extrapolating task*** extends inferred structures to future dynamics under uncertainty. ***(v) Acting task*** integrates preceding results to produce decisions, supporting adaptive actions in complex environments. Dataset details are provided in Appendix G.3.

Data Statistics. TSCognition contains a total of 41,086 QA pairs, with a train–evaluation split ratio of 0.8:0.2. Specifically, 32,869 samples are used for training, while 8,217 samples are reserved for evaluation, including 6,163 in-distribution (ID) samples and 2,054 out-of-distribution (OOD) samples. In terms of data complexity, each question is associated with multivariate time series whose dimensionality ranges from 3 to 6 variables. The length of each individual time series dimension varies from 437 to 1059 time steps, with an average length of 785. In addition, each question is formulated as a multiple-choice problem with 4 or 5 options, among which only one is correct.

Pipeline. We first perform ***Temporal Subset Construction*** to organize time series by temporal characteristics and remove low-quality samples. We then conduct ***Context and Template Construction***, where variable information, contextual labels, answer candidates, and distractors are generated and independently verified. Finally, ***QA Construction*** uses the verified labels and templates for rule-based matching to produce the final samples. More details are in Appendix H.

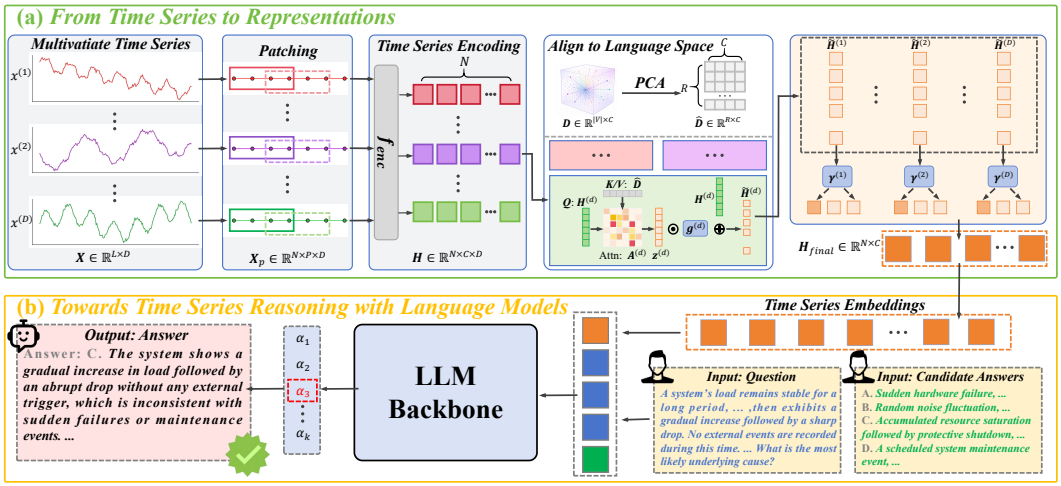


Figure 3: **Overview of TSAlign.** TSAlign encodes multivariate time series into patch-level representations, aligns them with the language space, and fuses variables into unified tokens for reasoning.

4 TSAlign: A Framework for Bridging TS and Language

Figure 3 provides an overview of TSAlign, which first derives language-compatible time series representations and then integrates them with textual inputs for LLM-based reasoning.

4.1 From Time Series to Representations

Given a multivariate time series $\mathbf{X} \in \mathbb{R}^{L \times D}$, where L denotes the number of time steps and D the number of variables, the objective is to derive a unified representation for downstream reasoning.

Patch-based Temporal Encoding. Modeling long time series is challenging due to varying numbers of time steps and complex temporal dependencies. To address this, we decompose each variable $\mathbf{x}^{(d)} \in \mathbb{R}^{L}$ into a sequence of overlapping patches with length P and stride S (Nie et al., 2022), yielding $\mathbf{x}_p^{(d)} \in \mathbb{R}^{N \times P}, N = \lfloor \frac{L-P}{S} \rfloor + 1$. Each patch is then encoded by a time series encoder $f_{\text{enc}}(\cdot)$: $\mathbf{H}^{(d)} = f_{\text{enc}}(\mathbf{x}_p^{(d)}), \mathbf{H}^{(d)} \in \mathbb{R}^{N \times C}$, where C denotes the embedding dimension. This results in a sequence of patch-level embeddings for each variable.

Alignment to Language Space. Given the discrepancy between time series and language embedding spaces, we align the former to the semantic structure of the pre-trained language embedding space to improve representation compatibility. Let $\mathbf{D} \in \mathbb{R}^{|\mathcal{V}| \times C}$ denote the language embedding matrix, where $|\mathcal{V}|$ is the vocabulary size. Due to its large scale, we apply principal component analysis (PCA) to obtain a compact set of principal semantic directions: $\hat{\mathbf{D}} = \text{PCA}(\mathbf{D}), \hat{\mathbf{D}} \in \mathbb{R}^{R \times C}, R \ll |\mathcal{V}|$. We then perform attention-based alignment by treating the time series embeddings as queries and the principal language embeddings as keys and values:

$$\mathbf{A}^{(d)} = \text{softmax}\left(\mathbf{H}^{(d)}\hat{\mathbf{D}}^{\top}/\sqrt{C}\right), \quad \mathbf{Z}^{(d)} = \mathbf{A}^{(d)}\hat{\mathbf{D}}. \tag{3}$$

To preserve patch-level temporal information while incorporating semantic guidance, we formulate alignment as a gated residual update. The original time series embedding is kept as the backbone representation, and the language-embedding-induced semantic representation is added as a controlled correction whose strength is determined by a learnable gate:

$$\mathbf{g}^{(d)} = \sigma(\mathbf{H}^{(d)}\mathbf{w}_g), \quad \tilde{\mathbf{H}}^{(d)} = \mathbf{H}^{(d)} + \mathbf{g}^{(d)} \odot \mathbf{Z}^{(d)}. \tag{4}$$

The resulting $\tilde{\mathbf{H}}^{(d)} \in \mathbb{R}^{N \times C}$ preserves sequence length while aligning with the language embedding space, enabling segment-wise semantic refinement without information loss.

Multi-dimensional Fusion. Following alignment, each variable is represented as a sequence of embeddings $\tilde{\mathbf{H}}^{(d)}$. Since the number of variables D may vary across samples, a unified rep-

resentation must be constructed. To this end, we modulate the contribution of each dimension through a gating mechanism. Specifically, we first compute a summary representation for each dimension: $\bar{\mathbf{h}}^{(d)} = \frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{H}}_i^{(d)}$, and assign an importance weight to the d-th dimension: $g^{(d)} = \sigma(w_g^\top \bar{\mathbf{h}}^{(d)} + b_g)$. The final representation is obtained by weighted fusion across dimensions at each patch position:

$$\mathbf{H}_{\text{final},i} = \frac{1}{\sum_{d=1}^{D} g^{(d)}} \sum_{d=1}^{D} g^{(d)} \tilde{\mathbf{H}}_i^{(d)}, \quad i = 1, \ldots, N. \tag{5}$$

Thus, the fused representation is $\mathbf{H}_{\text{final}} \in \mathbb{R}^{N\times C}$. This formulation preserves patch-level structure while enabling flexible multi-dimensional integration. Independent gating avoids competitive normalization, allowing multiple informative dimensions to contribute. The $\mathbf{H}_{\text{final}}$ serves as a consistent representation for downstream reasoning.

4.2 Towards Time Series Reasoning with Language Models

Given the unified time series representation $\mathbf{H}_{\text{final}} \in \mathbb{R}^{N\times C}$ obtained in Sec. 4.1, we combine it with textual inputs to perform reasoning via a LLM. Each sample consists of a question q and a set of candidate answers $\mathcal{A} = \{a_1, \ldots, a_K\}$, where the question may further include task instructions and contextual descriptions. To bridge numerical time series and language tokens, we first project the time series representation into the embedding space of the LLM: $\tilde{\mathbf{z}} = W_p\mathbf{z}$, where $W_p \in \mathbb{R}^{d\times C}$ and d denotes the hidden size of the LLM. The projected vector $\tilde{\mathbf{z}} \in \mathbb{R}^d$ is treated as a special temporal token that summarizes the global dynamics of the input series. We then concatenate this temporal token with the token embeddings of the textual question and candidate answers to form a unified input sequence: $\mathbf{E}_k = \mathcal{F}\big(\tilde{\mathbf{z}},\ \mathrm{Emb}(q),\ \mathrm{Emb}(a_k)\big)$, where $\mathrm{Emb}(\cdot)$ denotes the token embedding function of the LLM, and $\mathbf{E}_k$ is the resulting multimodal representation for the candidate answer a_k.

The resulting embedding sequence is fed into the LLM, which induces a scoring function over candidate answers: $s_k = f_\theta(\mathbf{E}_k)$, where f_θ denotes the LLM parameterized by θ. The final prediction is given by: $\hat{y} = \arg\max_k s_k$. This formulation enables the model to condition explicitly on temporal representations rather than relying on textualized or visualized surrogates of time series, allowing the LLM to jointly model temporal structures and linguistic semantics for more effective reasoning.

We train the model in a supervised manner using the multiple-choice question answering formulation. Given a question q and a set of candidate answers $\mathcal{A} = \{a_1, \ldots, a_K\}$, the model produces a score s_k for each candidate a_k based on the joint reasoning over time series and textual inputs. During training, we adopt an end-to-end optimization strategy.

5 Experiment

5.1 Comparison on QA Reasoning Tasks

Baselines & Evaluation. We compare our method with several representative baselines, covering general-purpose LLMs, VLMs, and existing time series QA models. For general LLMs and VLMs, we consider ***GPT-5.1***, ***Qwen2.5-Instruct*** (Qwen et al., 2025), and ***Qwen2.5-VL*** (Qwen et al., 2025) as representative models. For these models, we evaluate two typical input paradigms: (1) representing time series as text and combining them with background information (TS-Text), and (2) converting time series into line charts and providing them together with background information (Vision-Text). For time series QA methods, we include several recent approaches, including ***ITFormer*** (Wang et al., 2025), ***Time-MQA*** (Kong et al., 2025), ***Time-LLM*** (Jin et al., 2024), ***ChatTS*** (Xie et al., 2025), and ***GPT4TS*** (Zhou et al., 2023). For fair comparison, all methods are implemented using Qwen2.5-Instruct-7B as the backbone LLM. In our ***TSAlign*** family, the 3B & 7B variants use Qwen2.5-Instruct-3B & 7B as the backbone. See Appendix A for baseline details.

Main Results. As shown in Table 2, the proposed TSAlign outperforms existing methods across all five reasoning task categories. TSAlign-7B achieves the strongest overall performance, with particularly clear advantages on higher-level reasoning tasks such as Inferring and Acting. TSAlign-3B also performs competitively and surpasses the corresponding TS-Text baselines on most tasks. Overall, both model scales consistently outperform their backbone-matched TS-Text counterparts, and

Table 2: Accuracy (%) on five QA reasoning task categories under ID and OOD settings.

Methods	Types	Decoding		Grounding		Inferring		Extrapolating		Acting	
		ID	OOD	ID	OOD	ID	OOD	ID	OOD	ID	OOD
Random Guessing	–	22.1	23.5	22.4	21.9	24.2	21.8	23.5	22.7	24.1	23.9
GPT-5.1*	Proprietary API	71.4	68.5	74.6	70.1	68.3	72.5	57.4	63.9	71.5	77.0
Qwen2.5-Instruct-32B*	TS-Text	43.2	43.2	38.3	39.5	40.7	36.8	35.2	33.6	37.2	34.9
Qwen2.5-Instruct-7B*	TS-Text	34.6	37.1	29.4	32.7	36.5	33.3	28.3	30.2	32.6	29.7
Qwen2.5-VL-7B*	Vision-Text	37.5	41.2	34.1	34.7	30.7	28.4	33.0	34.8	26.9	25.3
Qwen2.5-Instruct-7B	TS-Text	51.7	62.3	44.0	48.6	45.2	41.7	39.8	36.5	41.5	45.6
Qwen2.5-VL-7B	Vision-Text	48.5	52.6	40.2	46.7	39.5	35.4	38.1	41.5	38.2	32.5
ITFormer	TS-Text	42.4	46.8	47.4	50.6	42.1	43.6	46.5	42.7	40.8	42.3
Time-MQA	TS-Text	45.5	44.1	42.5	41.4	36.5	38.4	34.8	31.4	36.8	32.7
Time-LLM	TS-Text	36.0	38.2	32.5	36.1	34.1	32.8	35.4	39.3	31.6	33.4
ChatTS	TS-Text	39.1	43.5	36.8	42.3	37.4	30.8	32.0	36.8	28.5	30.6
GPT4TS	TS-Text	34.8	37.9	33.2	38.4	32.4	29.7	31.2	33.5	35.6	38.1
TSAlign-3B	Ours	74.9	76.5	72.7	76.6	65.8	74.3	74.5	81.3	82.4	86.0
TSAlign-7B	Ours	83.1	87.6	85.4	87.4	91.7	88.7	77.9	84.6	90.4	95.3

*denotes results obtained through direct inference with closed-source or non-finetuned models, serving as references for comparison with the trained models.

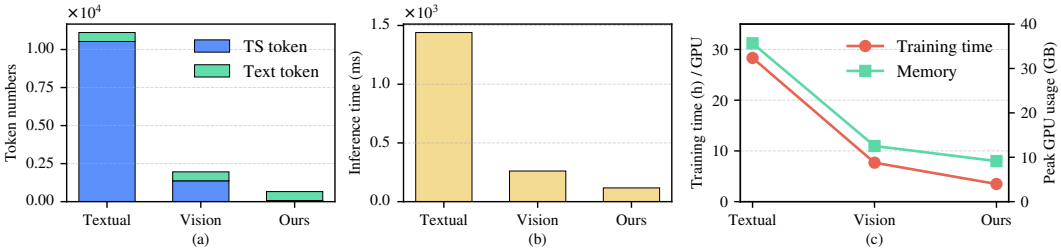


Figure 4: **Efficiency analysis.** (a) Token composition (TS token vs. Text token); (b) Average inference time per sample; (c) Training time normalized to a single A800 GPU and peak memory.

TSAlign-7B further exceeds the proprietary GPT-5.1 baseline. Our model can sufficiently leverage the structured knowledge encoded in LLMs to perform time series reasoning.

Efficiency Analysis. As shown in Figure 4, we compare the efficiency of TSAlign-7B with two common paradigms that convert time series into textual or visual inputs for QA reasoning under a unified experimental setup. Specifically, all methods are evaluated with the same hyperparameter configuration, using Qwen2.5-Instruct-7B and Qwen2.5-VL-7B as the base LLMs for textual and visual paradigms. TSAlign substantially reduces token complexity and consistently improves both training and inference efficiency. It also requires the lowest peak GPU memory among the compared methods, demonstrating a clear computational advantage over textual and vision-based paradigms.

5.2 Comparison on Pattern Analysis Tasks

Baselines & Evaluation. To evaluate performance on time series pattern analysis tasks, we compare ***TSAlign*** with representative methods from three paradigms. (1) General LLMs, including the proprietary ***GPT-5.1*** and the open-source ***Qwen2.5-Instruct-7B***, to assess pattern reasoning ability without explicit temporal modeling; (2) Specialized time series models, such as ***FEDformer*** (Zhou et al., 2022), ***PatchTST*** (Nie et al., 2022), ***TimesNet*** (Wu et al., 2022), and ***iTransformer*** (Liu et al., 2023), which focus on modeling deterministic and probabilistic temporal patterns; (3) TS-Text methods, including ***GPT4TS*** (Zhou et al., 2023), ***Time-LLM*** (Jin et al., 2024), ***ChatTS*** (Xie et al., 2025), and ***S^{2}IP*** (Pan et al., 2024), which convert time series into text for pattern analysis and reasoning.

We conduct experiments on the TimerBed (Liu et al., 2025b), which consists of six datasets spanning three categories. ***Simple tasks (RCW, TEE)*** involve one-to-one pattern-to-label mappings, ***Complex tasks (ECG, EMG)*** require integrating multiple patterns, and ***Probabilistic tasks (CTU, HAR)*** exhibit uncertain relationships between patterns and labels. See Appendix G.2 for details.

Main Results. As shown in Table 3, TSAlign achieves the best overall performance across the six time series pattern analysis tasks. Specifically, for Simple and Complex tasks, TSAlign-7B attains the best results on all datasets, significantly outperforming both traditional time series models and existing TS-Text methods. For the more challenging Probabilistic tasks, TSAlign also exhibits

Table 3: Accuracy (%) on three pattern analysis task categories across six datasets.

Methods	Types	Simple		Complex		Probabilistic		Weighted Avg.
		RCW	TEE	ECG	EMG	CTU	HAR	
Random Guessing	–	50.0	14.3	25.0	33.3	50.0	16.7	32.98
GPT-5.1*	Proprietary API	60.9	24.8	27.1	39.5	52.7	36.2	40.34
Qwen2.5-Instruct-7B*	TS-Text	52.1	16.9	22.5	35.9	46.3	28.4	33.85
Qwen2.5-Instruct-7B	TS-Text	67.8	40.5	28.8	44.9	59.1	52.0	45.65
FEDformer	TS	76.6	42.9	26.4	73.3	51.6	89.9	52.16
PatchTST	TS	82.1	57.1	24.8	60.0	64.0	79.6	52.09
TimesNet	TS	80.2	61.9	26.2	73.3	64.0	88.7	53.28
iTransformer	TS	76.9	21.4	24.5	46.7	46.4	89.5	51.10
GPT4TS	TS-Text	78.2	62.3	21.4	75.5	69.9	87.6	50.01
Time-LLM	TS-Text	62.5	47.3	25.6	63.8	60.2	88.4	46.61
ChatTS	TS-Text	71.3	50.8	28.4	69.4	52.6	77.5	49.81
S^2IP	TS-Text	81.5	59.2	27.5	82.7	64.2	75.2	52.79
TSAlign-3B	Ours	84.2	71.4	42.1	76.8	68.7	91.5	63.28
TSAlign-7B	Ours	86.0	74.3	42.9	81.7	77.4	93.2	64.60

*denotes results obtained through direct inference with closed-source or non-finetuned models, serving as references for comparison with the trained models.

stronger robustness under uncertainty. In contrast, text-based methods generally underperform, suggesting that converting time series into discrete textual representations leads to information loss.

5.3 Ablation Study

To evaluate the effectiveness of each component, we conduct a series of ablation studies: ***representation granularity***, ***alignment strategy***, and ***fusion strategy***. Ablation details are in Appendix E.

Representation Granularity. We analyze the effect of representation granularity by considering three strategies: point-level, patch-level, and variable-level. As shown in Table 4, patch-level achieves the best performance across model scales and evaluation. QA denotes the QA task, and PA denotes the pattern analysis task.

Table 4: Comparison of performance under different granularity levels.

Granularity	TSAlign-3B		TSAlign-7B	
	QA (%)	PA (%)	QA (%)	PA (%)
Patch-level (Ours)	**79.4**	**63.3**	**85.1**	**64.6**
Point-level	65.7	54.3	69.2	58.2
Variable-level	46.9	40.2	48.6	37.4

Alignment Strategy. As shown in Table 5, alignment improves performance over the no-alignment baseline, highlighting the mismatch between time series and language representations. While prototype and attention-based methods leverage semantic structure, directly replacing the original features leads to information loss. Residual alignment mitigates this issue, and our formulation further improves performance by controlling semantic integration. Effective alignment requires semantic guidance while preserving and selectively integrating temporal information.

Fusion Strategy. As shown in Table 6, the fusion strategy plays a crucial role in integrating multi-dimensional information. Uniform fusion yields the lowest accuracy, as it treats all dimensions equally. Scalar-weight fusion provides moderate gains but remains limited in expressiveness. Softmax-based fusion further improves performance with adaptive weighting, yet its competitive normalization suppresses useful signals. In contrast, our gated fusion achieves the best results by allowing multiple informative dimensions to be preserved and combined.

Table 5: Ablation on alignment strategies.

Module	Strategy	TSAlign-3B		TSAlign-7B	
		QA	PA	QA	PA
Alignment Strategy	Ours	**79.4**	**63.3**	**85.1**	**64.6**
	Residual Alignment	75.3	56.4	81.2	59.7
	Attention Alignment	65.8	52.1	69.3	55.2
	Prototype Matching	63.8	54.4	65.4	56.7
	Linear Alignment	57.5	50.2	60.9	51.8
	No Alignment	52.3	46.1	55.7	48.0

Table 6: Ablation on multi-dimensional fusion.

Module	Strategy	TSAlign-3B		TSAlign-7B	
		QA	PA	QA	PA
Fusion Strategy	Ours	**79.4**	**63.3**	**85.1**	**64.6**
	Softmax Fusion	66.3	54.8	73.5	56.0
	Scalar-weight Fusion	63.5	51.9	66.4	54.2
	Uniform Fusion	58.1	49.7	62.5	52.3

5.4 Addition Analysis

Shortcut Analysis. To examine whether the benchmark can be solved through superficial cues, we conduct several controlled shortcut tests with TSAlign-7B, as shown in Figure 5. We remove time series inputs, randomly shuffle 30% of timestamps, swap answer options, and replace the original context with mismatched context from other samples. Performance drops substantially under text-only, shuffled, and wrong-context settings, while option swapping has only a limited effect. This

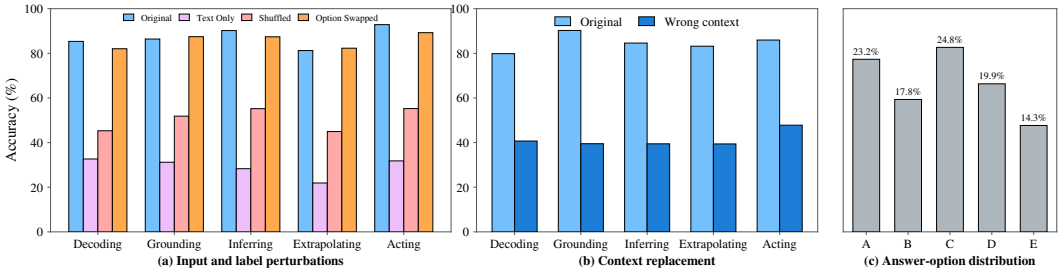


Figure 5: **Shortcut-control analysis of TSCognition.** We evaluate text-only input, timestamp shuffling, option swapping and context replacement to assess potential shortcut reliance.

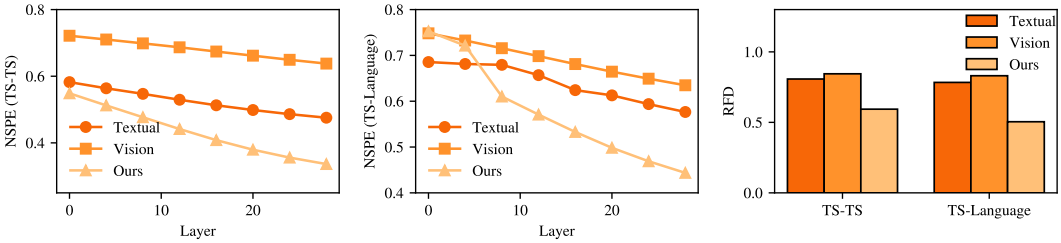


Figure 6: NSPE and RFD analysis of modality alignment. The left and middle plots show NSPE for TS–TS and TS–Language settings, respectively, while the right plot shows RFD.

indicates that no evident shortcut leakage is present in the benchmark. In addition, the answer choices are balanced across the benchmark, reducing the risk of exploiting option-frequency biases.

General Reasoning Capability. To examine whether fine-tuning affects the original reasoning ability of the LLM, we evaluate the Qwen2.5-3B-Instruct and the fine-tuned TSAlign-3B on GPQA (5-shot) (Rein et al., 2023). The performance gap is small (**32.4%** vs. **29.5%**), suggesting that fine-tuning does not substantially degrade general capability. Furthermore, to investigate whether TSAlign relies solely on LLM fine-tuning, we freeze the Qwen2.5-3B-Instruct backbone and only optimize the alignment components in Table 7. Fine-tuning the LLM brings additional benefits without sacrificing general capability.

Table 7: Effect of LLM fine-tuning on ID and OOD reasoning.

Method	ID	OOD
TSAlign-3B (Frozen LLM)	70.4	76.6
TSAlign-3B	74.1	78.9

6 Reducing the Modality Gap between Time Series and Language

Normalized Subspace Projection Error (NSPE). As shown in Figure 6, NSPE (Mu & Viswanath, 2018; Ethayarajh, 2019) measures the projection error of time series representations onto the principal subspace of language, where lower values indicate better alignment. TS–TS reflects intra-modal alignment within time series representations, while TS–Language measures cross-modal alignment. Under the TS–TS setting (left), as the LLM depth increases, our method achieves the lowest NSPE. Under the TS–Language setting (middle), our method reduces NSPE across all layers and continues to improve with depth, demonstrating its effectiveness in mitigating the modality gap.

Relative Fréchet Distance (RFD). At the distribution level (right), we compute the Fréchet Distance (Dowson & Landau, 1982; Heusel et al., 2017) based on the mean and covariance of representations and normalize it to obtain RFD, where lower values indicate better alignment. The results show that our method achieves the lowest RFD under both TS–TS and TS–Language settings.

7 Conclusion & Limitations

In our work, we study LLM-based time series understanding beyond conventional state modeling. We introduce TSCognition, a multimodal benchmark for time series reasoning, which integrates time series with textual information and covers five cognitive reasoning tasks. Building on this, we propose TSAlign, a unified framework that aligns time series representations with the language reasoning space. Experiments on TSCognition and the TimerBed show that TSAlign achieves strong performance and efficiency, showing the value of language-conditioned time series reasoning. This work still has several limitations. ***First***, TSCognition mainly adopts a multiple-choice format, which may not fully capture open-ended reasoning. ***Second***, although the dataset covers diverse domains, further expansion to more real-world scenarios and expert-level tasks would improve its coverage.

AI USE STATEMENT

Generative AI tools were used solely for language polishing, including improving grammar, clarity, fluency, and overall readability of the manuscript. They were also used to assist in constructing candidate answer and distractor templates, with the detailed generation and verification procedure described in the Appendix. They were not relied upon to determine the scientific claims, experimental results, or conclusions of this work. All AI-assisted content was carefully reviewed, verified, and, when necessary, further modified by the authors to ensure accuracy and consistency with the original technical content. We take full responsibility for the final manuscript and all of its contents.

REPRODUCIBILITY STATEMENT

We provide detailed information to facilitate the reproduction of our experiments. The data sources, construction pipeline, hierarchical task definitions, dataset statistics, and train/evaluation and ID/OOD splitting protocols of TSCognition are provided in Appendix H and Appendix G.3. Detailed descriptions and configurations of all comparison methods are reported in Appendix A. The formulations of the QA reasoning and pattern analysis tasks are provided in Appendix B, while the prompt templates for TS-Text, Vision-Text, and TSAlign are given in Appendix C. Training configurations and implementation details are reported in Appendix F. Detailed ablation settings for representation granularity, alignment strategies, and multivariate fusion are provided in Appendix E. Additional representation analyses based on NSPE and RFD are presented in Appendix D. Together, these materials provide the necessary details for reproducing the data construction, model training, evaluation, and analysis in this work.

ETHICS STATEMENT

Our work focuses on methodological and scientific research and does not involve human subjects, animals, personal or sensitive data, or environmentally sensitive materials. We do not identify any specific ethical risks or conflicts of interest associated with this study. All experiments are conducted following standard research practices, and the authors take responsibility for ensuring the integrity, validity, and reliability of the reported results.

REFERENCES

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Yifu Cai, Arjun Choudhry, Mononito Goswami, and Artur Dubrawski. Timeseriesexam: A time series understanding exam. *arXiv preprint arXiv:2410.14752*, 2024.

Jialin Chen, Aosong Feng, Ziyu Zhao, Juan Garza, Gaukhar Nurbek, Cheng Qin, Ali Maatouk, Leandros Tassiulas, Yifeng Gao, and Rex Ying. Mtbench: A multimodal time series benchmark for temporal reasoning and question answering. *arXiv preprint arXiv:2503.16858*, 2025.

Mingyue Cheng, Jiahao Wang, Daoyu Wang, Xiaoyu Tao, Qi Liu, and Enhong Chen. Can slow-thinking llms reason over time? empirical studies in time series forecasting. In *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining*, pp. 99–110, 2026.

David C Dowson and BV666017 Landau. The fréchet distance between multivariate normal distributions. *Journal of multivariate analysis*, 12(3):450–455, 1982.

Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 55–65, 2019.

Tong Guan, Zijie Meng, Dianqi Li, Shiyu Wang, Chao-Han Huck Yang, Qingsong Wen, Zuozhu Liu, Sabato Marco Siniscalchi, Ming Jin, and Shirui Pan. Timeomni-1: Incentivizing complex reasoning with time series in large language models. *arXiv preprint arXiv:2509.24803*, 2025.

Tong Guan, Sheng Pan, Johan Barthelemy, Zhao Li, Yujun Cai, Cesare Alippi, Ming Jin, and Shirui Pan. Timeomni-vl: Unified models for time series understanding and generation. *arXiv preprint arXiv:2602.17149*, 2026.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Shuning Jia, Baijun Song, Canming Ye, and Chun Yuan. M3time: Llm-enhanced multi-modal, multi-scale, and multi-frequency multivariate time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 22265–22273, 2026.

Yue Jiang, Yile Chen, Xiucheng Li, Qin Chao, Shuai Liu, and Gao Cong. Fstllm: Spatio-temporal llm for few shot time series forecasting. In *Forty-second International Conference on Machine Learning*, 2025a.

Yushan Jiang, Kanghui Ning, Zijie Pan, Xuyang Shen, Jingchao Ni, Wenchao Yu, Anderson Schneider, Haifeng Chen, Yuriy Nevmyvaka, and Dongjin Song. Multi-modal time series analysis: A tutorial and survey. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 6043–6053, 2025b.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-llm: Time series forecasting by reprogramming large language models. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 23857–23880, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/680b2a8135b9c71278a09cafb605869e-Paper-Conference.pdf.

Baoyu Jing, Sanhorn Chen, Lecheng Zheng, Boyu Liu, Zihao Li, Jiaru Zou, Tianxin Wei, Zhining Liu, Zhichen Zeng, Ruizhong Qiu, et al. Tsaqa: Time series analysis question and answering benchmark. *arXiv preprint arXiv:2601.23204*, 2026.

Yaxuan Kong, Yiyuan Yang, Yoontae Hwang, Wenjie Du, Stefan Zohren, Zhangyang Wang, Ming Jin, and Qingsong Wen. Time-mqa: Time series multi-task question answering with context enhancement. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 29736–29753, 2025.

Chenxi Liu, Qianxiong Xu, Hao Miao, Sun Yang, Lingzheng Zhang, Cheng Long, Ziyue Li, and Rui Zhao. Timecma: Towards llm-empowered multivariate time series forecasting via cross-modality alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 18780–18788, 2025a.

Haoxin Liu, Shangqing Xu, Zhiyuan Zhao, Lingkai Kong, Harshavardhan Kamarthi, Aditya B Sasanur, Megha Sharma, Jiaming Cui, Qingsong Wen, Chao Zhang, et al. Time-mmd: Multi-domain multimodal dataset for time series analysis. *Advances in Neural Information Processing Systems*, 37:77888–77933, 2024.

Haoxin Liu, Chenghao Liu, and B Aditya Prakash. A picture is worth a thousand numbers: Enabling llms reason about time series via visualization. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7486–7518, 2025b.

Peiyuan Liu, Hang Guo, Tao Dai, Naiqi Li, Jigang Bao, Xudong Ren, Yong Jiang, and Shu-Tao Xia. Calf: Aligning llms for time series forecasting via cross-modal fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 18915–18923, 2025c.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.

Yucong Luo, Yitong Zhou, Mingyue Cheng, Jiahao Wang, Daoyu Wang, Tingyue Pan, and Jintao Zhang. Time series forecasting as reasoning: A slow-thinking approach with reinforced llms. *arXiv preprint arXiv:2506.10630*, 2025.

Romain Meunier, Farah Benamara, Véronique Moriceau, Zhongzheng Qiao, and Savitha Ramasamy. Crisists: Coupling social media textual data and meteorological time series for urgency classification. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16082–16099, 2025.

Jiaqi Mu and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations*, 2018.

Saurish Nagrath and Saroj Kumar Panigrahy. Patch-level tokenization with cnn encoders and attention for improved transformer time-series forecasting. *arXiv preprint arXiv:2601.12467*, 2026.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

Zijie Pan, Yushan Jiang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. $\mathbf{S}^2$ip-llm: Semantic space informed prompt learning with llm for time series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

Zhongzheng Qiao, Sheng Pan, Anni Wang, Viktoriya Zhukova, Yong Liu, Xudong Jiang, Qingsong Wen, Mingsheng Long, Ming Jin, and Chenghao Liu. It's time: Towards the next generation of time series forecasting benchmarks. *arXiv preprint arXiv:2602.12147*, 2026.

Xin Qiu, Junlong Tong, Yirong Sun, Yunpu Ma, and Xiaoyu Shen. The few govern the many: Unveiling few-layer dominance for time series models. *arXiv preprint arXiv:2511.07237*, 2025.

Xin Qiu, Junlong Tong, Yirong Sun, Yunpu Ma, Wei Zhang, and Xiaoyu Shen. Rethinking the role of llms in time series forecasting. *arXiv preprint arXiv:2602.14744*, 2026.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Dan Schumacher, Erfan Nourbakhsh, Rocky Slavin, and Anthony Rios. Prompting underestimates llm capability for time series classification. *arXiv preprint arXiv:2601.03464*, 2026.

Mingtian Tan, Mike A Merrill, Vinayak Gupta, Tim Althoff, and Thomas Hartvigsen. Are language models actually useful for time series forecasting? *Advances in Neural Information Processing Systems*, 37:60162–60191, 2024.

Junlong Tong, Liping Xie, Shixiong Fang, Wankou Yang, and Kanjian Zhang. Hourly solar irradiance forecasting based on encoder–decoder model using series decomposition and dynamic error compensation. *Energy Conversion and Management*, 270:116049, 2022. ISSN 0196-8904. doi: https://doi.org/10.1016/j.enconman.2022.116049. URL https://www.sciencedirect.com/science/article/pii/S019689042200838X.

Yilin Wang, Peixuan Lei, Jie Song, Yuzhe Hao, Tao Chen, Yuxuan Zhang, Lei Jia, Yuanxiang Li, and Zhongyu Wei. Itformer: Bridging time series and natural language for multi-modal qa with large-scale multitask dataset. In *International Conference on Machine Learning*, pp. 63324–63344. PMLR, 2025.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations*, 2022.

Wen Wu, Ziyang Zhang, Liwei Liu, Xuenan Xu, Jimin Zhuang, Ke Fan, Qitan Lv, Junlin Liu, Chen Zhang, Zheqi Yuan, et al. Scits: Scientific time series understanding and generation with llms. *arXiv preprint arXiv:2510.03255*, 2025.

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.

Zhe Xie, Zeyan Li, Xiao He, Longlong Xu, Xidao Wen, Tieying Zhang, Jianjun Chen, Rui Shi, and Dan Pei. Chatts: Aligning time series with llms via synthetic data for enhanced understanding and reasoning. *Proceedings of the VLDB Endowment*, 18(8):2385–2398, 2025.

Fangxu Yu, Xingang Guo, Lingzhi Yuan, Haoqiang Kang, Hongyu Zhao, Lianhui Qin, Furong Huang, Bin Hu, and Tianyi Zhou. Tsrbench: A comprehensive multi-task multi-modal time series reasoning benchmark for generalist models. *arXiv preprint arXiv:2601.18744*, 2026.

Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 8980–8987, 2022.

Liangwei Nathan Zheng, Wenhao Liang, Wei Emma Zhang, Miao Xu, Olaf Maennel, and Weitong Chen. Lifting manifolds to mitigate pseudo-alignment in llm4ts. In *Proceedings of the ACM Web Conference 2026*, pp. 3764–3775, 2026.

Siru Zhong, Weilin Ruan, Ming Jin, Huan Li, Qingsong Wen, and Yuxuan Liang. Time-vlm: Exploring multimodal vision-language models for augmented time series forecasting. In *International Conference on Machine Learning*, pp. 78478–78497. PMLR, 2025.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pp. 27268–27286. PMLR, 2022.

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.

A BASELINES

GPT-5.1. GPT-5.1 is the proprietary API baseline used in this work, representing a strong general-purpose LLM. We evaluate it under the zero-shot setting by organizing the time series information, textual background, question, and candidate answers into a unified natural language input. This baseline is used to examine the capability of a strong general LLM without task-specific training.

Qwen2.5-Instruct (Qwen et al., 2025). Qwen2.5-Instruct is the large-scale open-source LLM baseline. We adopt the TS-Text input paradigm, where numerical time series values are textualized and then concatenated with the question, background description, and candidate answers as model input. This baseline is used to evaluate the reasoning ability of a large open-source LLM under textualized time series inputs, and to analyze whether increasing the LLM scale can alleviate the structural fragmentation and context burden caused by time series textualization.

Qwen2.5-VL (Qwen et al., 2025). Qwen2.5-VL is the vision-language baseline. We convert time series into line plots and feed the resulting images, together with the textual background, question, and candidate answers, into the VLM. This baseline is used to evaluate the feasibility of reasoning over visualized time series with VLMs, and to examine whether visual representations can effectively preserve temporal structures and key dynamic patterns.

ITFormer (Wang et al., 2025). ITFormer is a baseline designed for time series and language modeling, and is used to evaluate the performance of existing time series QA methods on TSCognition. This method typically connects time series information to LLMs through textualization or language interfaces, enabling the model to make predictions based on task descriptions and candidate answers. We include ITFormer to examine whether methods specifically designed for time series language tasks can adapt to more complex multi-level QA reasoning settings.

Time-MQA (Kong et al., 2025). Time-MQA is a representative method for multimodal time series question answering. It combines time series with natural language questions to support QA tasks based on temporal dynamics and textual information. We use Time-MQA as a directly related time series QA baseline to compare existing QA construction paradigms with the more complex cognitive reasoning tasks in TSCognition.

Time-LLM (Jin et al., 2024). Time-LLM is a representative LLM-based time series method, which leverages LLM capabilities by converting or aligning time series into input forms that language models can process. We use it as one of the TS-Text baselines to evaluate the transferability of existing LLM4TS methods to QA reasoning scenarios. This comparison reflects whether LLM4TS methods originally designed for forecasting or representation learning remain effective under language-conditioned reasoning tasks.

ChatTS (Xie et al., 2025). ChatTS is a baseline for time series understanding and conversational analysis. It typically represents time series in textual or LLM-processable forms, allowing the LLM to generate answers according to user questions. We include ChatTS to evaluate the performance of conversational time series analysis methods on standardized multiple-choice QA reasoning tasks.

GPT4TS (Zhou et al., 2023). GPT4TS is an early representative method that introduces pretrained language models into time series modeling. It mainly leverages language-model architectures or pretrained knowledge to enhance time series representation learning, but its original design is more oriented toward conventional time series tasks rather than complex language-conditioned reasoning. We use GPT4TS as a baseline to analyze the applicability of such LLM-based time series methods to QA reasoning benchmarks such as TSCognition.

S^2IP (Pan et al., 2024). S^2IP is adopted as a baseline for TimerBed, i.e., the pattern analysis task in our evaluation. It is designed to recognize characteristic temporal patterns from time series inputs rather than perform language-conditioned QA reasoning. Therefore, we use S^2IP to assess the capability of existing pattern analysis methods on the Simple Deterministic, Complex Deterministic, and Probabilistic categories.

FEDformer (Zhou et al., 2022). FEDformer is a frequency-enhanced time series modeling method that captures long-term dependencies and periodic structures through frequency-domain representations and decomposition mechanisms. In this work, we use FEDformer as a traditional TS baseline to evaluate the performance of non-LLM time series methods on the TimerBed pattern analysis tasks. This comparison helps determine whether the advantage of TSAlign comes only from LLM-based reasoning, or whether it also preserves effective modeling of raw temporal patterns.

PatchTST (Nie et al., 2022). PatchTST is a patch-based Transformer model for time series modeling. It divides time series into local patches and performs representation learning at the patch level. Since TSAlign also adopts patch-level time series representations, PatchTST serves as an important reference for comparing pure time series patch modeling with patch representations aligned to the language space. It is mainly used to examine the ability of patch-based TS models to recognize different types of temporal patterns in TimerBed without textual semantics or LLM reasoning.

TimesNet (Wu et al., 2022). TimesNet is a general-purpose time series analysis model that captures complex temporal variations through multi-period structure modeling. We use TimesNet as a strong time series baseline to evaluate the overall performance of traditional TS architectures on simple, complex, and probabilistic pattern analysis tasks.

iTransformer (Liu et al., 2023). iTransformer is an improved Transformer-based time series model that organizes tokens along the variable dimension to enhance multivariate time series modeling. In this work, iTransformer is used as a traditional TS baseline to evaluate the classification performance of non-textual time series models on TimerBed.

B Problem Formulation

B.1 Problem Formulation for QA Reasoning Tasks

Given a multivariate time series sample $X = \{x_1, x_2, \ldots, x_T\}$, where $x_t \in \mathbb{R}^d$, T denotes the sequence length, and d denotes the variable dimension. Unlike conventional time series tasks that predict fixed outputs such as future values, class labels, or anomaly indicators, QA reasoning re-

quires the model to extract task-relevant temporal evidence and connect it with natural language semantics, scenario conditions, and decision objectives.

Let $q \in \mathcal{Q}$ denote a natural language query, which may contain task instructions, variable descriptions, contextual information, background events, or reasoning objectives. Let $\mathcal{A} = \{a_1, a_2, \ldots, a_K\}$ denote the set of candidate answers. The model is required to select the answer that is best supported by both the temporal evidence in X and the semantic condition in q. This can be formulated as estimating the conditional distribution $p_\theta(a_k \mid X, q)$, or equivalently learning a scoring function $s_k = S_\theta(X, q, a_k)$, where s_k measures the compatibility among the time series, the question, and the k-th candidate answer. The final prediction is given by $\hat{a} = \arg\max_{a_k \in \mathcal{A}} s_k$.

To characterize the internal reasoning process, we introduce a candidate-conditioned latent state z_k, which represents the task-relevant temporal evidence induced by the triplet (X, q, a_k). This state is not a generic time series embedding, but a candidate-specific representation that may contain local patterns, long-term trends, abrupt changes, cross-variable dependencies, and context-related temporal dynamics that support or refute the candidate answer. The conditional probability of each candidate can then be written as:

$$p_\theta(a_k \mid X, q) = \int p_\theta(a_k \mid z_k, q)\, p_\theta(z_k \mid X, q, a_k)\, dz_k. \tag{6}$$

Here, $p_\theta(z_k \mid X, q, a_k)$ describes how the model extracts candidate-specific temporal evidence from the input time series under the question condition, while $p_\theta(a_k \mid z_k, q)$ evaluates whether the candidate answer is semantically and temporally consistent with the inferred evidence.

This formulation highlights the conditional nature of QA reasoning. The same time series may require different evidence under different questions or candidate answers. For example, one question may focus on the global trend, another may focus on a local anomaly, while another may require reasoning over variable dependencies or future risks. Therefore, QA reasoning is not equivalent to directly classifying a time series into a fixed label space. Instead, it can be viewed as a task-conditioned evidence aggregation problem, where the model must dynamically select and organize temporal evidence according to the language query and candidate hypothesis.

During training, we adopt a multiple-choice negative log-likelihood objective. For each sample $(X, q, \mathcal{A}, a^*)$, where a^* denotes the correct answer, the model first computes scores $\{s_1, s_2, \ldots, s_K\}$ for all candidate answers. The conditional probability of each candidate is obtained:

$$P_\theta(a_k \mid X, q, \mathcal{A}) = \frac{\exp(s_k)}{\sum_{j=1}^{K} \exp(s_j)}. \tag{7}$$

The training objective is $\mathcal{L}_{\mathrm{QA}} = -\log P_\theta(a^* \mid X, q, \mathcal{A})$. It requires the model to distinguish the correct answer from distractors by modeling the relative compatibility among temporal dynamics, textual context, and answer semantics. Since the candidate answer set varies across questions, the answer space is not fixed, making semantic alignment and conditional reasoning essential.

B.2 Problem Formulation for Pattern Analysis Tasks

We formalize pattern analysis tasks, which are used to evaluate whether a model preserves fundamental temporal pattern modeling capability. Different from QA reasoning tasks, pattern analysis does not rely on an natural language query. Given the time series $X = \{x_1, x_2, \ldots, x_T\}, x_t \in \mathbb{R}^d$, the model is expected to directly identify discriminative temporal structures from the raw time series and map them to a class label or pattern analysis target $y \in \mathcal{Y}$. This can be formulated as learning the conditional distribution $p_\theta(y \mid X)$, or equivalently a mapping function $f_\theta : X \to y$.

To describe the temporal pattern extraction process, we introduce a latent pattern representation z, which summarizes the task-relevant temporal structures contained in X. Such structures may include local peaks, periodic patterns, shape variations, temporal transitions, cross-variable coordination, or statistical dynamics. The prediction process can then be written as a marginalization over the latent pattern representation:

$$p_\theta(y \mid X) = \int p_\theta(y \mid z)\, p_\theta(z \mid X)\, dz. \tag{8}$$

$p_\theta(z \mid X)$ extracts discriminative temporal patterns from the raw input, while $p_\theta(y \mid z)$ maps the inferred pattern representation to the target class.

The corresponding training objective is the standard supervised classification loss. For each sample (X, y), the model maximizes the likelihood of the ground-truth label, or equivalently minimizes the negative log-likelihood: $\mathcal{L}_{\mathrm{PA}} = -\log p_\theta(y \mid X)$. The core of pattern analysis is thus to learn the discriminative relationship between time series patterns and labels. It evaluates whether the model can capture useful temporal structures from raw signals, but does not require the model to interpret natural language questions or select evidence according to different semantic conditions.

C Prompt Template

To ensure comparability across different models and input paradigms, we use a unified prompt template to organize the time series, textual context, question, and candidate answers. For all LLM-based baselines, the prompt only provides the information required to complete the task, including the time-series content, variable descriptions, scenario background, question statement, and candidate options. It does not include any additional answer hints or manually designed reasoning chains. We organize each sample in a multiple-choice question answering format. The input first provides a task instruction, asking the model to select the most appropriate answer based on the time series and textual context. It then presents the variable descriptions, time-series segments, scenario background, and question, followed by the candidate answers. The model is required to output the corresponding option rather than generate open-ended long-form text. This design reduces evaluation instability caused by differences in generation style across models.

TS-Text Baseline. Time series values are converted into textual sequences using a unified format. For a univariate time series, the prompt presents the variable name together with its value list. For a multivariate time series, the observations of each variable are listed separately. This template preserves the temporal order and variable names, allowing the LLM to receive time-series information in textual form. The detailed prompt template is shown in Figure 7.

Vision-Text Baseline. We render the time series as line plots and provide the same textual context, question, and candidate answers as in the QA reasoning setting. The model is required to answer by jointly considering the temporal patterns shown in the image and the accompanying textual information. To ensure fairness, all plots are generated using a unified visualization setup, including the same image size, axis format, variable annotation scheme, and color rules. The detailed prompt template is shown in Figure 8.

TSAlign. TSAlign does not fully textualize or visualize the time series. Instead, it feeds the temporal representation obtained after encoding, alignment, and fusion into the LLM together with the textual prompt. Therefore, at the prompt level, TSAlign uses the same question, context, and candidate-answer template as other QA methods; the only difference lies in how time-series information is incorporated. This ensures that the comparison focuses on the representation paradigm of time series itself. The detailed prompt template is shown in Figure 9.

Prompt Template for TS-Text Baseline

Instruction: You are an expert in time series analysis and temporal reasoning. Please answer the multiple-choice question based on the textualized time series, context, and candidate answers.
Time Series:
Variable 1: $[v_1^1, v_2^1, \ldots, v_T^1]$
Variable 2: $[v_1^2, v_2^2, \ldots, v_T^2]$
...
Variable D: $[v_1^D, v_2^D, \ldots, v_T^D]$
Variable Information: [Variable names and descriptions]
Context: [Scenario background or task context]
Question: [Question]
Candidate Answers:
A. [Candidate answer A]
B. [Candidate answer B]
C. [Candidate answer C]
D. [Candidate answer D]

Figure 7: Prompt template for TS-Text baseline.

Prompt Template for Vision-Text Baseline

Instruction: You are an expert in time series analysis and visual reasoning. Please answer the multiple-choice question based on the time series plot, context, and candidate answers.
Time Series Image: [Line plot of the time series]
Variable Information: [Variable names and descriptions]
Context: [Scenario background or task context]
Question: [Question]
Candidate Answers:
A. [Candidate answer A]
B. [Candidate answer B]
C. [Candidate answer C]
D. [Candidate answer D]

Figure 8: Prompt template for Vision-Text baseline.

Prompt Template for TSAlign

Instruction: You are an expert in time series analysis and temporal reasoning. A temporal token representing the input time series is provided to the model. Please answer the multiple-choice question based on the temporal evidence, context, and candidate answers.
Temporal Token: [Encoded time series representation inserted at the embedding level]
Variable Information: [Variable names and descriptions]
Context: [Scenario background or task context]
Question: [Question]
Candidate Answers:
A. [Candidate answer A]
B. [Candidate answer B]
C. [Candidate answer C]
D. [Candidate answer D]

Figure 9: Prompt template for TSAlign.

D REPRESENTATION ANALYSIS FOR TIME SERIES–LANGUAGE SPACE

To assess whether TSAlign mitigates the modality gap between time series and language representations, we use Normalized Subspace Projection Error and Relative Fréchet Distance. These metrics measure geometric subspace alignment and distributional discrepancy, respectively. Since pretrained LLM embeddings form a structured semantic space, matching dimensionality alone is insufficient; time series representations should also align with its geometry and distribution.

Normalized Subspace Projection Error (NSPE). NSPE (Mu & Viswanath, 2018; Ethayarajh, 2019) measures how well time series representations align with the principal subspace of the LLM embedding space. We apply PCA to the pretrained word embedding matrix to obtain dominant semantic directions, then compute the projection error of time series representations onto this subspace. A lower NSPE indicates that more representation energy lies within the language subspace, implying better geometric compatibility with the LLM input space.

Formally, let $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$ denote the word embedding matrix of the pretrained LLM, where $|\mathcal{V}|$ is the vocabulary size and d is the embedding dimension. We perform PCA on $\mathbf{E}$ and take the top-k principal components to form an orthonormal basis $\mathbf{U}_k \in \mathbb{R}^{d \times k}$. Given a time series representation $\mathbf{z} \in \mathbb{R}^d$, its projection onto the dominant language subspace is given by $\mathbf{U}_k \mathbf{U}_k^\top \mathbf{z}$, and the residual component not explained by this subspace is $\mathbf{z} - \mathbf{U}_k \mathbf{U}_k^\top \mathbf{z}$. We define NSPE as the ratio between the residual norm and the original representation norm:

$$\mathrm{NSPE}(\mathbf{z}) = \frac{\left\|\mathbf{z} - \mathbf{U}_k \mathbf{U}_k^\top \mathbf{z}\right\|_2}{\|\mathbf{z}\|_2}. \tag{9}$$

This normalized formulation removes the influence of representation scale, making comparisons across layers and methods more stable. A smaller $\mathrm{NSPE}(\mathbf{z})$ indicates that a larger proportion of $\mathbf{z}$ lies within the principal language subspace, whereas a larger value suggests that $\mathbf{z}$ contains more directions deviating from the language embedding space, which may make it harder for the LLM to effectively utilize the time series information. For a set of time series representations $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^n$, we compute the average NSPE over samples: $\mathrm{NSPE}(\mathbf{Z}) = \frac{1}{n} \sum_{i=1}^n \mathrm{NSPE}(\mathbf{z}_i)$, which serves as an overall measure of geometric alignment at the corresponding layer.

We compute NSPE under TS–TS and TS–Language settings. TS–TS evaluates the internal consistency of time series representations, while TS–Language measures their geometric alignment with the principal language subspace and directly reflects the modality gap. As shown in Table 8, TSAlign yields progressively lower NSPE across layers in both settings, indicating more compact temporal representations and improved geometric compatibility with the language space.

Table 8: **NSPE under two settings across LLM layers.** Lower values indicate better alignment.

Layer	TS–TS			TS–Language		
	Textual	Vision	Ours	Textual	Vision	Ours
0	0.582	0.722	0.549	0.686	0.748	0.753
4	0.564	0.710	0.512	0.681	0.733	0.721
8	0.547	0.699	0.477	0.679	0.716	0.610
12	0.529	0.687	0.442	0.657	0.698	0.571
16	0.513	0.674	0.408	0.624	0.681	0.533
20	0.499	0.662	0.380	0.613	0.665	0.498
24	0.486	0.649	0.356	0.594	0.649	0.469
28	0.476	0.638	0.336	0.577	0.635	0.444

Relative Fréchet Distance (RFD). RFD (Heusel et al., 2017; Dowson & Landau, 1982) measures the overall discrepancy between time series and language representations from a distributional perspective. Unlike NSPE, RFD further examines whether a set of time series representations is close to the language embedding distribution in terms of both mean and covariance structure. Therefore, NSPE focuses more on whether the representation directions are geometrically aligned, whereas RFD characterizes whether the overall representation distribution is distributionally compatible with the language space. We first use FD to measure the discrepancy between two representation distributions. Given two sets of representations with means $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, and covariance matrices $\boldsymbol{\Sigma}_1$ and

$\mathbf{\Sigma}_2$, the FD is defined as:

$$\mathrm{FD} = \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2 + \mathrm{Tr}\left(\mathbf{\Sigma}_1 + \mathbf{\Sigma}_2 - 2\left(\mathbf{\Sigma}_1\mathbf{\Sigma}_2\right)^{\frac{1}{2}}\right). \tag{10}$$

The first term measures the distance between the centers of the two distributions, while the second term captures the discrepancy between their covariance structures. Thus, a smaller FD indicates that the two representation sets are closer in both global location and distributional shape. To enable relative comparison across different methods, we further define RFD. Let $\mathbf{Z}_m$ denote the set of time series representations produced by method m, and let $\mathbf{E}$ denote the set of language embeddings. The RFD of method m with respect to a reference baseline is defined as:

$$\mathrm{RFD}(\mathbf{Z}_m, \mathbf{E}) = \frac{\mathrm{FD}(\mathbf{Z}_m, \mathbf{E})}{\mathrm{FD}(\mathbf{Z}_{\mathrm{ref}}, \mathbf{E})}. \tag{11}$$

$\mathbf{Z}_{\mathrm{ref}}$ can be chosen as the unaligned time series representations or the representations produced by a basic baseline. Under this definition, $\mathrm{RFD} < 1$ indicates that the current method is closer to the language embedding distribution than the reference baseline, whereas $\mathrm{RFD} > 1$ indicates a larger distributional discrepancy. In our experiments, we mainly compare the relative distributional distances of different methods under the same setting; therefore, a lower RFD indicates better distribution-level alignment.

For each method, we extract time series representations from different LLM layers and compute their Fréchet Distance to the language embedding distribution. The TS–TS setting measures the relative distance among time series representation distributions, reflecting intra-modal consistency, while the TS–Language setting measures the distributional gap between time series and language embeddings. As shown in Table 9, TSAlign achieves the lowest RFD in both settings, with 0.594 for TS–TS and 0.504 for TS–Language, clearly outperforming Textual and Vision. This indicates that TSAlign forms more stable time series representations and brings their global distribution closer to the language embedding space, demonstrating stronger modality alignment.

Table 9: **RFD comparison under two settings.** Lower values indicate better alignment.

Setting	Textual	Vision	Ours
TS–TS	0.807	0.844	0.594
TS–Language	0.783	0.831	0.504

E Detailed Ablation Setting

E.1 Representation Granularity

Point-level. Raw timestamps are directly used as modeling units. Given $\mathbf{X} \in \mathbb{R}^{L \times D}$, each timestamp is independently projected to the LLM hidden space. This preserves maximal temporal resolution but produces long token sequences and high computational cost, while each point contains only limited local numerical information.

Variable-level. Each variable is globally aggregated into a single representation before fusion. This greatly reduces input length and computation, but compresses fine-grained temporal structures such as abrupt changes, local trends, short-period patterns, and stage-wise variations. It therefore serves as a global-summary baseline.

Patch-level (Ours). Each variable is divided into local patches, which are encoded into compact patch-level representations. This preserves local dynamics while substantially reducing sequence length compared with point-level modeling. The resulting tokens are further processed by language-space alignment and multivariate fusion before LLM reasoning.

E.2 Alignment Strategy

No Alignment. The explicit alignment module is removed. Time-series features are only linearly projected to match the LLM hidden size and directly used as temporal tokens. This setting tests whether dimensional compatibility alone is sufficient for effective LLM reasoning.

Linear Alignment. A learnable linear mapping projects time-series representations into the LLM embedding space. Although it provides additional transformation capacity, it does not explicitly exploit the semantic structure of pretrained language embeddings.

Prototype Matching. Semantic prototypes extracted from pretrained word embeddings are used as alignment targets. Time-series representations are mapped into the space spanned by these prototypes. This explicitly introduces language-space structure, but discrete prototypes may restrict continuous temporal information.

Attention Alignment. Time-series representations are used as queries, while language embeddings or compressed semantic bases serve as keys and values. Attention adaptively selects relevant semantic directions, providing more flexible alignment than prototype matching.

Residual Alignment. A residual connection is added to Attention Alignment. Instead of replacing the original time-series features, the language-induced representation is treated as a semantic correction term. This helps preserve temporal information during alignment.

Ours Alignment. TSAlign further introduces a learnable gate on top of Residual Alignment. The gate adaptively controls semantic injection, strengthening language guidance when useful while preserving original temporal dynamics when necessary.

E.3 Fusion Strategy

Uniform Fusion. Representations from different variables are simply averaged with equal weights. This baseline tests whether multivariate information can be effectively integrated without modeling variable importance.

Scalar-weight Fusion. A learnable scalar weight is assigned to each variable. This captures global variable importance, but the weights remain static across samples and temporal conditions.

Softmax Fusion. Variable contributions are dynamically normalized with softmax. Although sample-adaptive, the competitive normalization forces variables to share a fixed probability mass and may suppress simultaneously informative variables.

Our Fusion. TSAlign independently gates each variable in a non-competitive manner. Multiple informative variables can therefore contribute simultaneously, which is better suited to multivariate time series with complementary signals.

F Reproducibility

We adopt unified implementation, training, and evaluation settings whenever possible. Unless otherwise specified, all experiments are conducted on two NVIDIA A800 GPUs using PyTorch, HuggingFace Transformers, DeepSpeed, and Accelerate. We fix the random seed to 2026 and use identical data splits and ID/OOD evaluation protocols for each group of comparative experiments. All models are trained with supervised learning using AdamW, with consistent hyperparameters within the same experimental group. For LLM-based methods, we use the official tokenizer and pretrained weights of each backbone and maintain consistent input length, answer format, and prompt organization. Mixed-precision training is used to reduce memory consumption and improve efficiency.

Table 10: Experimental hyperparameters.

GPUs	$2 \times$ NVIDIA A800 GPUs
Framework	PyTorch + HuggingFace Transformers
Distributed training	DeepSpeed / Accelerate
Precision	bfloat16
Optimizer	AdamW
Learning rate	2×10^{-5}
Weight decay	1×10^{-2}
Batch size per GPU	4
Gradient accumulation	4
Effective batch size	32
Gradient clipping	1.0
Learning rate scheduler	Cosine decay
Warmup ratio	0.0
Random seed	2026

G Dataset Details

G.1 Case Study: Limitations of Existing TS Reasoning Datasets

To more concretely illustrate the limitations of existing reasoning datasets, we conduct case studies on several representative question formats. ***Cases 1–6 are collected from prior benchmarks***, including Time-MQA (Kong et al., 2025) and Time-Omni-1 (Guan et al., 2025), and are used here only for qualitative analysis. These examples highlight common limitations of existing datasets, such as simplified question formulations, limited use of textual context, weak multi-step reasoning requirements, and insufficient coverage of decision-oriented temporal tasks.

Overall, these cases suggest that existing time series reasoning datasets still have several limitations. Many questions mainly focus on shallow pattern recognition, such as detecting volatility, seasonality, or simple scenario matching, rather than requiring deeper reasoning over temporal dynamics. Some examples provide limited or ambiguous textual context, making it difficult to evaluate whether the model truly performs semantic grounding or merely relies on surface-level statistical cues. In addition, several tasks ask for causal, future-oriented, or decision-related judgments from time series data alone, but do not provide sufficient contextual evidence, domain constraints, or task objectives to support reliable reasoning. As a result, these benchmarks are useful for evaluating basic temporal pattern understanding, but remain limited in assessing language-conditioned reasoning that requires joint modeling of signals, context, relations, and decisions.

Case 1

Question:
Which answer is the most comprehensive in relation to the question? Considering the data points [47.0, 51.0, 51.0, 20.0, 48.0, 44.0, 41.0, 28.0, 48.0, 38.0, 67.0, 66.0, 70.0, 54.0, 63.0, 56.0, 76.0, 58.0, 60.0, 58.0, 69.0, 56.0, 39.0, 50.0], which statement best describes the volatility of this time series?

Candidate answers:
A) The volatility is low with consistent data points.
B) There is moderate volatility with regular patterns.
C) High volatility with sudden peaks and drops.
D) The time series exhibits a perfectly linear trend.

Case 2

Question:
How clear or understandable is the reasoning provided in the answer? Considering the data points [60.0, 28.0, 46.0, 51.0, 37.0, 46.0, 41.0, 35.0, 32.0, 66.0, 10.0, 21.0, 67.0, 72.0, 83.0, 61.0, 11.0, 31.0, 62.0, 50.0, 56.0, 76.0, 53.0, 86.0], do you see any seasonal patterns?

Candidate answers:
A) Yes, there is a seasonal pattern.
B) No, there is no identifiable seasonal pattern.

Case 3

Question:
How clearly does the response explain its reasoning? (1 = Not Clear At All, 5 = Really Clear) What is the volatility level in the dataset [0.83, 0.8, 0.76, 0.75, 0.75, 0.76, 0.78, 0.8, 0.8, 0.8, 0.81, 0.8, 0.8, 0.8, 0.8, 0.79, 0.8, 0.82, 0.83, 0.82, 0.82, 0.82, 0.82, 0.81]?

Candidate answers:
A) High
B) Moderate
C) Low
D) No volatility

Case 4

Question:
You manage a home energy storage system with battery capacity 18 kWh, current state-of-charge (SoC) 5 kWh, max charging power 5 kW, and max discharging power 10 kW. Historical 48-hour hourly load (kWh) from 2023-07-10 00:00 to 2023-07-11 23:00 is given. Tomorrow's 24-hour pricing is off-peak \$0.22/kWh for hours 0–14 and 20–23, and peak \$0.54/kWh for hours 15–19. Based on historical usage, predict tomorrow's load and select the optimal 24-hour battery strategy from A–D, where hour indices range from 0 to 23.

Candidate answers:
A) Charge: {5, 11}; Discharge: {2, 18}
B) Charge: {1, 2}; Discharge: {15, 17}
C) Charge: {2, 10}; Discharge: {18, 19}
D) Charge: {11, 14}; Discharge: {16, 18}

Case 5

Question:
You are given two time series related to river discharge measurements, expressed in m^3/s. Through causal discovery methods, we aim to identify potential causal relationships between different measuring stations from data alone.
The time series of A463 is: [4.12, 3.8, 3.44, 3.11, 2.97, 2.85, 2.79, 2.76, 2.63, 2.74, 2.69, 2.77, 2.75, 2.79, 2.97, 3, 3.09, 3.18, 3.24, 3.16, 3.16, 3.37, 3.37, 3.54, 3.56, 3.51, 3.43, 3.37, 3.37, 3.21, 3.12, 3.07, 3.16, 3.12, 3.16, 3.15, 2.95, 3, 2.95, 2.84, 2.75, 2.74, 2.74, 2.74, 2.74, 2.74, 2.6, 2.56, 2.56, 2.53, 2.54, 2.52, 2.33, 2.33, 2.33, 2.4, 2.61, 2.9, 3.22, 3.45, 3.58, 3.95]
The time series of 4PRY is: [7.55, 7.31, 7, 6.78, 6.5, 6.2, 6.08, 6, 6, 5.91, 5.82, 5.78, 5.84, 5.92, 5.88, 5.89, 5.82, 6, 6.25, 6, 5.94, 5.9, 5.9, 5.89, 5.89, 6, 6, 5.9, 5.74, 5.81, 5.67, 5.55, 5.35, 5.29, 5.14, 5.11, 5.18, 5.12, 5.11, 5.07, 4.89, 4.81, 4.82, 4.7, 4.69, 4.61, 4.67, 4.58, 4.45, 4.55, 4.42, 4.44, 4.32, 4.28, 4.37, 4.53, 4.5, 4.62, 4.84, 5, 5.43, 5.74] Please identify the causal relationships between the two measurement stations. The data is collected every 12 hours from 2020-01-01 to 2020-01-31, with 62 points for each series.

Candidate answers:
A) A463 is the cause and 4PRY is the effect.
B) A463 and 4PRY are not causal.
C) 4PRY is the cause and A463 is the effect.

Case 6

Question:
You are given a time series. Please identify the scenario that most likely created it. [39, 58, 51, 45, 40, 39, 57, 45, 45, 34, 40, 32.64, 62, 92, 37, 31, 29.52, 46, 69, 79, 43, 51, 50, 50, 56, 45, 57, 32, 35, 55, 38, 41, 38, 54, 33, 48, 54, 40.45, 30.31, 56, 49, 37.41, 45, 39, 49, 31.52, 47, 51, 33, 53, 38.49, 33, 30, 35.52, 41, 32.61, 52, 44, 29.68, 41, 32.44, 46.5, 49, 48, 38, 46, 38, 49, 45, 52, 47, 45, 39, 47, 44.47, 28, 36, 39, 27.67, 43, 37, 41, 52, 50, 37, 30, 30, 27.4, 33, 40, 35.51, 38, 35.43, 38, 37, 42, 28.49, 27, 47, 32.45, 30, 42, 36.39, 29.62, 27, 29.55, 32, 46, 44, 29.42, 26.72, 32, 44, 34.61, 37, 45, 45, 38.44, 42, 50, 34, 47.5, 51, 30, 31.56, 34, 42, 45, 39, 30, 42, 34, 50, 27.33, 38, 41.55, 32.43, 40, 40, 50, 39, 33.53, 26, 25.73, ..., 27, 30, 22.59, 38, 33, 26.58, 23.48, 32.62, 40, 20.33, 27.4, 33.44, 35.42, 33.34, 31, 38, 34, 21, 21, 38, 36]

Candidate answers:
A) ATM daily cash withdrawals over a year with an annual festival (365 daily samples).
B) Weekend music festival causing a surge in noise levels (96 hourly samples).
C) Hourly restaurant food orders during a 2-week local food festival (336 hourly samples).
D) NYC daily taxi pick-ups over a year with a subway strike (365 daily samples).

G.2 TIMERBED

Overview. TimerBed (Liu et al., 2025b) is a public benchmark for evaluating time series pattern analysis ability. In this work, we use TimerBed as the evaluation benchmark for the pattern analysis task. Unlike TSCognition, TimerBed does not require models to perform multi-step semantic

reasoning over natural language questions, scenario context, and task objectives. Instead, it mainly evaluates whether models can identify discriminative temporal patterns from raw time series and map them to the corresponding categories. TimerBed categorizes time series pattern analysis tasks into three types: ***simple deterministic***, ***complex deterministic***, and ***probabilistic***.

Simple Deterministic. Simple Deterministic includes ***RCW*** and ***TEE***, whose labels are determined by clear local temporal patterns. RCW detects North Atlantic right whale calls from ocean audio, where a short rising "whoop" is the key pattern. TEE classifies transient electromagnetic events from FORTE satellite observations, characterized by distinctive power-density patterns such as radiation spikes followed by noise variations.

Complex Deterministic. Complex Deterministic includes ***ECG*** and ***EMG***, where classification depends on multiple temporal features. ECG distinguishes cardiac conditions using characteristics such as rhythm irregularity, P waves, and QRS morphology. EMG classifies physiological conditions such as healthy, neuropathy, and myopathy based on signal amplitude, duration, waveform complexity, and motor-unit responses.

Probabilistic includes ***HAR*** and ***CTU***, where the mapping between temporal patterns and labels is affected by unobserved factors. HAR recognizes daily activities from smartphone acceleration signals, with variability across users. CTU distinguishes desktop and laptop usage from 24-hour electricity traces, where user habits and working schedules introduce uncertainty.

Statistics. Table 11 summarizes the statistics of the six sub-datasets. These datasets vary in sample size, sequence length, and label space, ranging from small-scale datasets such as ***TEE*** and ***EMG*** to large-scale datasets such as ***RCW*** and ***ECG***. Most datasets are univariate, while ***HAR*** contains three variables, enabling evaluation under both single-variable and multivariate settings.

Table 11: Statistics of the TimerBed datasets.

Dataset	Variables	Time Series Length	Classes	Samples
RCW	1	4,000	2	30,000
TEE	1	319	7	143
ECG	1	1,500	4	43,673
EMG	1	1,500	3	205
CTU	1	720	2	500
HAR	3	206	6	10,299

G.3 TSCOGNITION

Overview. TSCognition is a multimodal benchmark for evaluating language-conditioned time series reasoning in real-world scenarios. Unlike pattern-analysis benchmarks such as TimerBed, it requires models to jointly understand numerical signals, textual context, and task objectives. Beyond identifying trends, periodicity, and abrupt changes, models must integrate scenario descriptions and variable semantics for grounding, inference, future-oriented judgment, and decision analysis. TSCognition is constructed from 15 public sources and contains approximately 41K QA samples, each consisting of a time series segment, textual context, a question, and candidate answers.

Hierarchical Cognitive Tasks. TSCognition organizes time series reasoning into five hierarchical categories: ***Decoding, Grounding, Inferring, Extrapolating, and Acting***. They describe a progressive process from perceiving temporal patterns to understanding context, reasoning about relations, anticipating future changes, and supporting actions. Decoding extracts basic dynamics, Grounding connects them with contextual semantics, Inferring analyzes dependencies and causes, Extrapolating performs future-oriented judgment, and Acting converts evidence into goal-oriented decisions.

(1) Decoding is the most fundamental task level, evaluating whether a model can identify temporal patterns directly from the signal. Typical patterns include trends, periodicity, peaks, abrupt changes, and local anomalies. For example, a model may identify rush-hour patterns in traffic flow, sudden spikes in server load, or daily periodicity in energy consumption.

(2) Grounding introduces textual context and requires models to associate observed temporal patterns with scenario descriptions or events. For example, traffic peaks may need to be linked

to holiday demand, energy changes to production stages, or physiological variations to medical interventions.

(3) Inferring requires relational reasoning over both time series and textual information. Models must analyze dependencies, latent causes, and conditional changes among variables. For example, increased request volume, CPU usage, and latency may indicate computational resource pressure, while joint changes in climate variables may reflect a weather-state transition.

(4) Extrapolating focuses on future-oriented judgment from observed temporal dynamics. Unlike numerical forecasting, models select the most plausible future state by combining current trends, contextual information, and candidate answers. Typical examples include identifying overload risks, worsening traffic congestion, or deterioration in physiological conditions.

(5) Acting is the highest-level task, requiring models to translate temporal evidence and reasoning into goal-oriented decisions. For example, sustained increases in system load may trigger resource scaling, worsening traffic may require route diversion, and abnormal equipment signals may suggest preventive maintenance.

Statistics. We report the sample distribution from two perspectives: domain source and task type, as shown in Figure 10. Overall, TSCognition covers diverse real-world domains and maintains a relatively balanced distribution across the five hierarchical reasoning tasks. This design ensures domain diversity while preventing the benchmark from being overly biased toward a specific reasoning type, thereby providing a more comprehensive basis for evaluating reasoning ability. In addition, Table 12 reports task-level statistics of the time series.

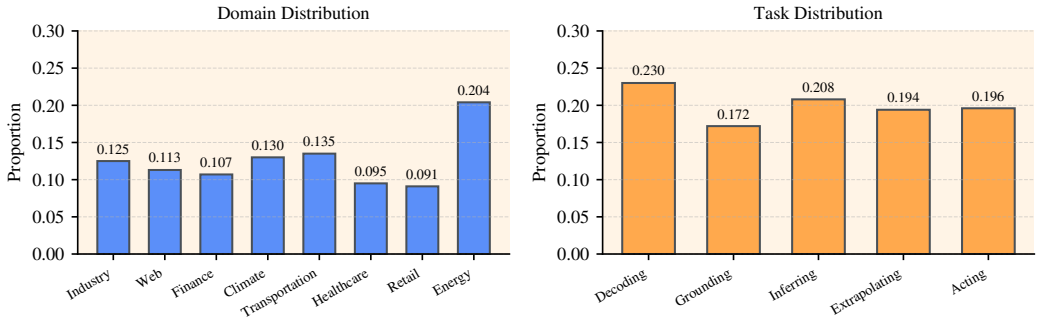


Figure 10: Distribution of TSCognition samples across eight domains and five hierarchical tasks.

Table 12: Task-level statistics of time series in TSCognition.

Tasks	Variable Numbers			Time Series Length			Choices		
	Min	Max	Avg.	Min	Max	Avg.	Min	Max	Avg.
Decoding Task	4	5	4.7	437	982	742	4	5	4.7
Grounding Task	3	6	4.4	656	1018	871	4	5	4.3
Inferring Task	3	6	5.1	882	1059	912	4	5	4.6
Extrapolating Task	3	6	5.2	601	1052	846	4	5	4.5
Acting Task	4	6	5.5	468	1047	795	4	5	4.2
Overall	3	6	4.9	437	1059	785	4	5	4.5

H TSCOGNITION CONSTRUCTION

H.1 DATA SOURCES

TSCognition is constructed from 15 public real-world datasets spanning eight domains, including Energy (Buildings-900K[1] and UMass Smart*[2]), Retail (Favorita Sales[3] and Dominick[4]), Healthcare (Project Tycho[5] and JHU CSSE COVID-19[6]), Transportation (LargeST[7] and METR-LA[8]), Climate (SubseasonalClimateUSA[9] and Oikolab Weather[10]), Finance (Bitcoin[11]), Web (Web Traffic[12]), and Industry (WADI[13], C-MAPSS[14], and Paderborn[15]).

H.2 FOUR-STAGE PIPELINE

The construction of TSCognition follows a four-stage pipeline that progressively transforms heterogeneous time series into verified reasoning-oriented QA samples, as shown in Figure 11.

(1) Temporal Pattern Subset Construction and Data Cleaning. We first organize the collected datasets into scenario-specific subsets according to their real-world application domains. Within each scenario, the original multivariate time series are further structured according to five intrinsic temporal characteristics: trend variations, periodic patterns, inter-variable dependencies, abnormal behaviors, and future dynamics. We simultaneously remove low-quality samples with severe missing values, excessive noise, invalid variable fields, or insufficient temporal structures, ensuring that the retained sequences contain meaningful evidence for subsequent reasoning tasks.

(2) Context and Answer Template Construction with Verification. For each temporal subset, we construct *Variable Information* and *Context* labels to describe variable semantics, scenario backgrounds, and task conditions. GPT-5.1 is then used to generate candidate answer templates and semantically plausible distractors conditioned on the corresponding temporal characteristics. To ensure reliability, all generated templates are independently reviewed by three annotators with at least six months of research experience in time series analysis. The annotators verify whether the answer templates are consistent with the underlying temporal evidence and whether the distractors remain plausible while contradicting that evidence. Subsets that fail verification are discarded.

(3) Time Series Selection. After obtaining verified temporal labels and answer templates, we segment the original time series into standardized input sequences ranging from 420 to 1100 time steps. This step reduces inconsistency across heterogeneous datasets while preserving sufficient temporal context for hierarchical reasoning. Each selected segment must retain the temporal characteristics associated with its corresponding label and provide adequate evidence for the QA construction.

(4) QA Generation and Answer Grounding. Finally, we generate QA samples by matching the selected time series segments with their verified temporal labels, contextual information, and answer/distractor templates through rule-based construction. TSCognition therefore does not directly generate questions and answers from scenario text alone:

$$\text{Scenario Text} \rightarrow \text{Question} \rightarrow \text{Answer}$$

[1] https://data.openei.org/submissions/5859
[2] https://traces.cs.umass.edu/docs/traces/smartstar/
[3] https://www.kaggle.com/competitions/favorita-grocery-sales-forecasting/data
[4] https://www.chicagobooth.edu/research/kilts/datasets/dominicks
[5] https://www.tycho.pitt.edu/
[6] https://github.com/CSSEGISandData/COVID-19
[7] https://github.com/liuxu77/LargeST
[8] https://github.com/liyaguang/DCRNN
[9] https://github.com/microsoft/subseasonal_data
[10] https://docs.oikolab.com/datasets/
[11] https://zenodo.org/records/5122101
[12] https://www.kaggle.com/competitions/web-traffic-time-series-forecasting/data
[13] https://www.sutd.edu.sg/itrust/itrust-labs/datasets/dataset-characteristics/wadi/
[14] https://data.nasa.gov/dataset/c-mapss-aircraft-engine-simulator-data
[15] https://mb.uni-paderborn.de/en/kat/research/bearing-datacenter

Instead, it follows a temporally grounded construction process:

Multivariate Time Series → Temporal Characteristics → Reasoning Subset →
Verified Answer Template → QA Construction

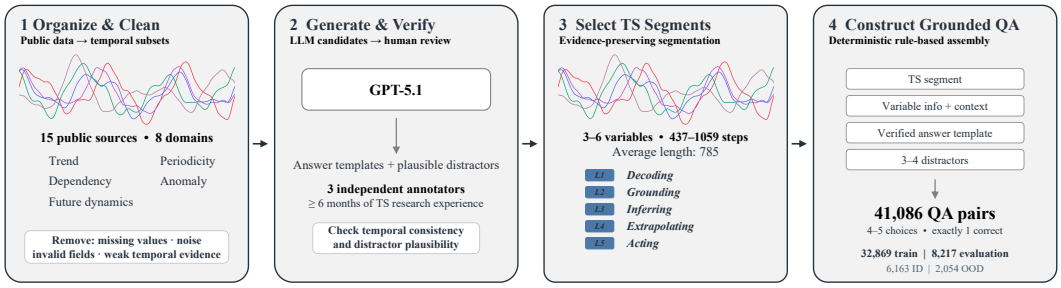


Figure 11: Four-stage construction pipeline of TSCognition.

H.3 Data Split

To avoid potential data leakage, the train/evaluation and ID/OOD splits of TSCognition are performed before QA construction and are defined at the underlying time series and scenario levels.

Train/Evaluation Split. TSCognition adopts an 80/20 train/evaluation split, using the underlying multivariate time series as the basic splitting unit. All Decoding, Grounding, Inferring, Extrapolating, and Acting questions generated from the same sequence are always assigned to the same split. Thus, the same underlying sequence never appears in both training and evaluation sets. The final split contains 32,869 training QA samples and 8,217 evaluation QA samples.

ID/OOD Split and Check. The evaluation set is further divided into 6,163 ID samples and 2,054 OOD samples. ID samples come from scenarios observed during training, while OOD samples originate from unseen scenarios whose underlying sequences are fully separated from the training set. This setting therefore evaluates generalization to new scenarios and temporal dynamics.

Cross-Split Duplicate Check. After sequence partitioning, we further inspect cross-split overlap based on data source, scenario identity, and underlying sequence content. Approximately **8.7%** of the initial candidate sequences are removed due to severe missing values, invalid variables, insufficient temporal structure, or obvious duplication. Before final QA construction, about **1.6%** of candidate sequences are identified as exact or highly overlapping duplicates and are either removed or reassigned to the same split. After filtering, the exact duplicate rate between the training and evaluation sets is **0%**, while the remaining near-duplicate rate is below **0.4%**. These cases are further manually reviewed to reduce the risk of performance gains caused by sequence memorization.

I More Results

Shortcut-Control Analysis. As shown in Table 13, removing the time-series input leads to a substantial performance drop across all five reasoning tasks under both ID and OOD settings, indicating that textual context alone is insufficient to solve the benchmark. Timestamp shuffling also causes a clear degradation, showing that the model depends not only on numerical values but also on their temporal ordering and dynamic structure. In contrast, option swapping produces only minor performance changes, suggesting limited sensitivity to answer positions. Overall, these results indicate that TSAlign primarily relies on genuine temporal evidence and contextual reasoning rather than superficial textual or option-position shortcuts.

Table 13: **Detailed shortcut-control results of TSAlign-7B on TSCognition.** T denotes text-only input, S denotes timestamp shuffling, and O denotes option swapping.

Split	Setting	Decoding	Grounding	Inferring	Extrapolating	Acting
ID	Original	83.1	85.4	91.7	77.9	90.4
	T	30.8	33.6	27.2	19.7	32.9
	S	48.9	56.5	55.8	50.7	64.4
	O	80.7	86.3	88.6	78.7	86.9
OOD	Original	87.6	87.4	88.7	84.6	95.3
	T	34.5	28.8	29.4	24.1	30.7
	S	41.7	47.2	54.6	39.2	46.1
	O	83.4	88.6	86.2	85.9	91.6

Context Replacement Analysis. Table 14 shows that replacing the original context with mismatched context consistently degrades performance across all reasoning tasks. The result indicates that correct contextual information is important for TSCognition and that the benchmark requires joint reasoning over temporal signals and scenario semantics.

Table 14: Detailed context replacement results.

Split	Setting	Decoding	Grounding	Inferring	Extrapolating	Acting
ID (1K)	Original	77.6	89.1	85.8	80.5	86.2
	Wrong Context	37.5	40.3	34.7	48.0	43.1
OOD (1K)	Original	82.1	91.4	83.4	85.9	85.7
	Wrong Context	43.9	38.7	44.2	30.8	52.5

Effect of LLM Fine-tuning. As reported in Table 15, freezing the LLM backbone leads to lower performance on most ID and OOD tasks. Fine-tuning the backbone therefore provides additional gains beyond the alignment modules alone, especially for higher-level reasoning tasks.

Table 15: Detailed results of TSAlign-3B with and without LLM fine-tuning.

Method	Decoding		Grounding		Inferring		Extrapolating		Acting	
	ID	OOD	ID	OOD	ID	OOD	ID	OOD	ID	OOD
TSAlign-3B (Frozen LLM)	67.5	73.6	67.9	73.2	64.7	75.9	75.2	76.4	76.6	83.8
TSAlign-3B	74.9	76.5	72.7	76.6	65.8	74.3	74.5	81.3	82.4	86.0

General Time-Series Reasoning. Table 16 compares TSAlign-3B with ITFormer (Wang et al., 2025) on TimeSeriesExam (Cai et al., 2024). TSAlign achieves stronger results on Anomaly and Causality while remaining competitive on Similarity, indicating that the proposed representation can retain useful general time-series reasoning ability beyond TSCognition.

Table 16: Results on TimeSeriesExam.

Method	Pattern	Anomaly	Noise	Similarity	Causality
ITFormer (Original Paper)	**0.83**	0.84	**0.95**	**0.92**	0.79
TSAlign-3B	0.7615	**0.8823**	0.8267	0.9030	**0.8584**

Human Evaluation. As shown in Table 17, both experts achieve consistently high agreement with the benchmark labels across all five reasoning tasks and both ID/OOD settings. These results support the reliability of the constructed QA samples and answer annotations.

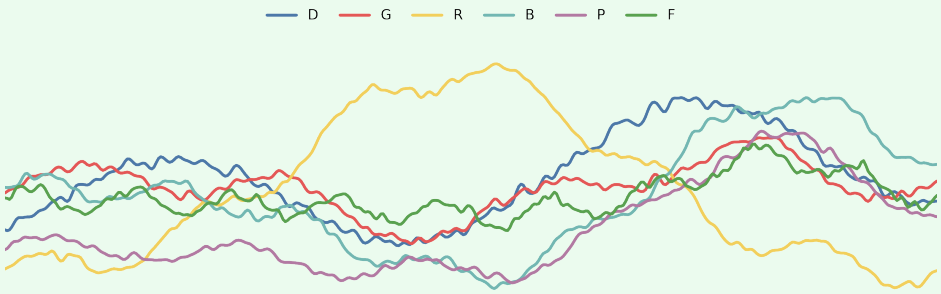
Table 17: Human evaluation results on randomly sampled ID and OOD examples.

Evaluator	Decoding ID/OOD	Grounding ID/OOD	Inferring ID/OOD	Extrapolating ID/OOD	Acting ID/OOD
Expert 1	94/98	96/99	97/98	93/99	97/95
Expert 2	97/100	94/100	96/97	97/100	98/93

J Case Study and Failure Analysis

Example of the Decoding task

Instruction: You are an expert in time series analysis and temporal reasoning. A temporal token representing the input time series is provided to the model. Please answer the multiple-choice question based on the temporal evidence, context, and candidate answers. Only one answer is correct.
Variable Information: D: Load Demand; G: Thermal Generation; R: Renewable Output; B: Battery Power; P: Electricity Price; F: Frequency Deviation.

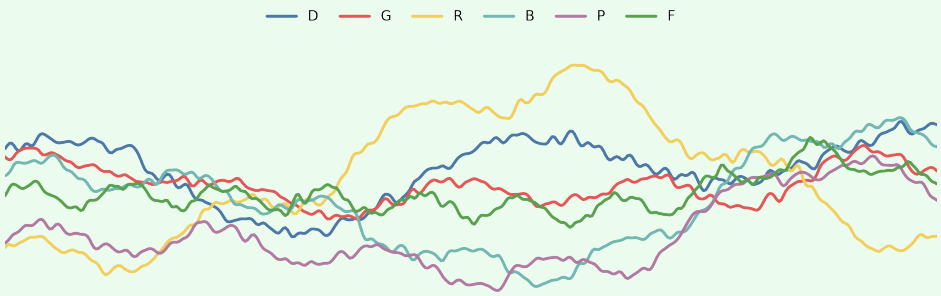


Context: The data describe the operation of a regional power system. The operator monitors load demand, thermal generation, renewable output, battery behavior, electricity price, and frequency deviation to evaluate supply-demand balance and dispatch decisions. Battery power reflects storage operation, while electricity price and frequency deviation may indicate system stress. The actual temporal state should be inferred from the provided time series representation.
Question: Which statement best characterizes the joint temporal behavior of renewable output and battery power?
Candidate Answers:
A. Renewable output stays low throughout the period, while the battery remains inactive with no clear charging or discharging phase.
B. Renewable output reaches its highest level only during the final high-demand period, while the battery mainly charges at the same time.
C. Renewable output and battery power increase monotonically together, indicating that storage output always follows renewable generation.
D. Renewable output first increases to a mid-period peak and then declines, while the battery tends to charge during the renewable-rich phase and discharge later.
E. Renewable output fluctuates randomly without a recognizable phase, while battery behavior shows no relation to the system state.
Output: Please output the correct candidate answer.
Answer: *D. Renewable output forms a clear mid-period peak before decreasing. Battery power shifts from charging during the renewable-rich phase to discharging afterward.*

Figure 12: Example of the Decoding task.

Example of the Grounding task

Instruction: You are an expert in time series analysis and temporal reasoning. A temporal token representing the input time series is provided to the model. Please answer the multiple-choice question based on the temporal evidence, context, and candidate answers. Only one answer is correct.
Variable Information: D: Load Demand; G: Thermal Generation; R: Renewable Output; B: Battery Power; P: Electricity Price; F: Frequency Deviation.

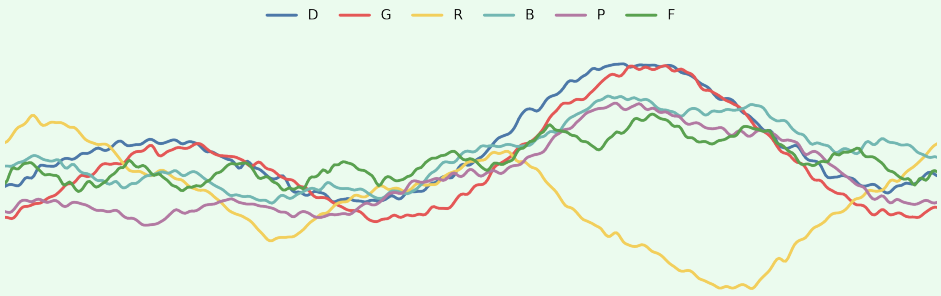


Context: The data describe the operation of a regional power system. The operator monitors load demand, thermal generation, renewable output, battery behavior, electricity price, and frequency deviation to evaluate supply-demand balance and dispatch decisions. Battery power reflects storage operation, while electricity price and frequency deviation may indicate system stress. The actual temporal state should be inferred from the provided time series representation.
Question: Which interpretation best grounds the observed temporal patterns in the operational context of power dispatch?
Candidate Answers:
A. A renewable-rich period is reflected by higher renewable output together with battery charging, suggesting that surplus renewable energy is being absorbed.
B. A renewable-rich period is reflected by high electricity price and large frequency deviation, because these always indicate abundant renewable supply.
C. A peak-load period is reflected by low demand and strong battery charging, suggesting that the system is under severe supply shortage.
D. A stable operating period is reflected by simultaneous increases in load demand, electricity price, and frequency deviation, regardless of renewable output.
E. The storage behavior cannot be grounded in the power-system context because battery power is unrelated to renewable output or demand.
Output: Please output the correct candidate answer.
Answer: *A. The high-renewable interval coincides with sustained battery charging. This pattern indicates that the battery is absorbing surplus renewable energy.*

Figure 13: Example of the Grounding task.

Example of the Inferring task

Instruction: You are an expert in time series analysis and temporal reasoning. A temporal token representing the input time series is provided to the model. Please answer the multiple-choice question based on the temporal evidence, context, and candidate answers. Only one answer is correct.
Variable Information: D: Load Demand; G: Thermal Generation; R: Renewable Output; B: Battery Power; P: Electricity Price; F: Frequency Deviation.

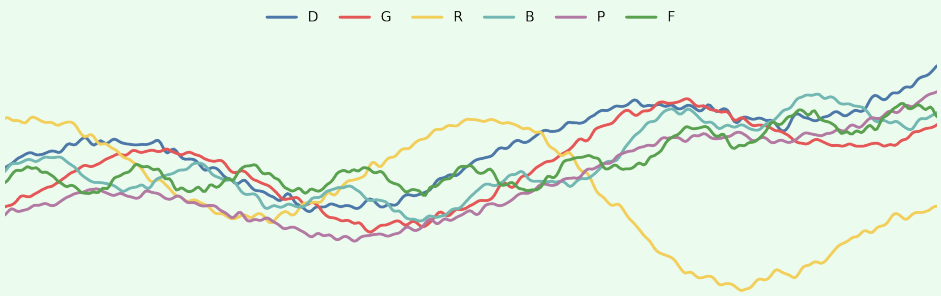


Context: The data describe the operation of a regional power system. The operator monitors load demand, thermal generation, renewable output, battery behavior, electricity price, and frequency deviation to evaluate supply-demand balance and dispatch decisions. Battery power reflects storage operation, while electricity price and frequency deviation may indicate system stress. The actual temporal state should be inferred from the provided time series representation.
Question: What is the most plausible explanation for the system pressure indicated by electricity price and frequency deviation during the high-load phase?
Candidate Answers:
A. The pressure is mainly caused by decreasing demand, since lower load usually forces the system to increase electricity price and frequency deviation.
B. The pressure is likely caused by the coincidence of high load demand, reduced renewable support, and increased dependence on dispatchable resources.
C. The pressure is mainly caused by excessive renewable generation, because abundant renewable output always increases electricity price and frequency deviation.
D. The pressure is unrelated to supply-demand balance and is only caused by random fluctuations in the battery power curve.
E. The pressure disappears because battery discharge and thermal generation fully eliminate all frequency deviation.
Output: Please output the correct candidate answer.
Answer: *B. Load demand rises while renewable output weakens during the high-pressure phase. The concurrent increases in thermal generation, electricity price, and frequency deviation indicate greater reliance on dispatchable resources.*

Figure 14: Example of the Inferring task.

Example of the Extrapolating task

Instruction: You are an expert in time series analysis and temporal reasoning. A temporal token representing the input time series is provided to the model. Please answer the multiple-choice question based on the temporal evidence, context, and candidate answers. Only one answer is correct.
Variable Information: D: Load Demand; G: Thermal Generation; R: Renewable Output; B: Battery Power; P: Electricity Price; F: Frequency Deviation.

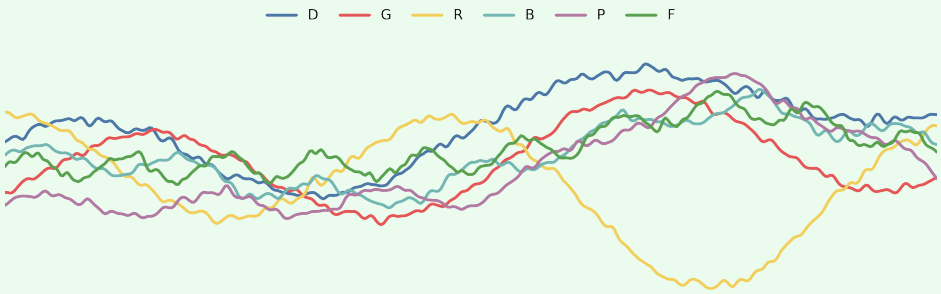


Context: The data describe the operation of a regional power system. The operator monitors load demand, thermal generation, renewable output, battery behavior, electricity price, and frequency deviation to evaluate supply-demand balance and dispatch decisions. Battery power reflects storage operation, while electricity price and frequency deviation may indicate system stress. The actual temporal state should be inferred from the provided time series representation.
Question: Assume that the current high-load phase continues for the next interval and renewable output does not recover. Which future evolution is most consistent with the observed temporal evidence?
Candidate Answers:
A. The system will remain risk-free even if demand keeps increasing, because frequency deviation is independent of supply-demand imbalance.
B. Renewable output will necessarily rebound immediately, so electricity price and frequency deviation should both drop to their minimum values.
C. Battery power is more likely to switch to strong charging, because a high-load phase implies surplus electricity in the system.
D. Thermal generation will become unnecessary, since declining renewable output reduces the need for dispatchable resources.
E. Electricity price and frequency deviation may further increase unless additional dispatchable generation or storage discharge is available.
Output: Please output the correct candidate answer.
Answer: *E. The final interval shows persistently high demand and continuously weakening renewable support. If this trend continues without additional controllable supply, electricity price and frequency deviation are likely to increase further.*

Figure 15: Example of the Extrapolating task.

Example of the Acting task

Instruction: You are an expert in time series analysis and temporal reasoning. A temporal token representing the input time series is provided to the model. Please answer the multiple-choice question based on the temporal evidence, context, and candidate answers. Only one answer is correct.
Variable Information: D: Load Demand; G: Thermal Generation; R: Renewable Output; B: Battery Power; P: Electricity Price; F: Frequency Deviation.

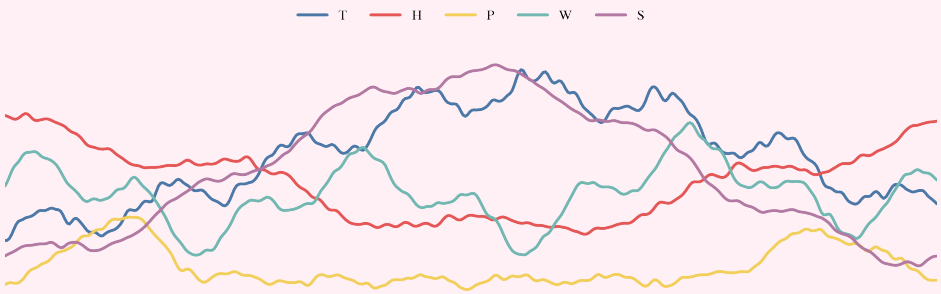


Context: The data describe the operation of a regional power system. The operator monitors load demand, thermal generation, renewable output, battery behavior, electricity price, and frequency deviation to evaluate supply-demand balance and dispatch decisions. Battery power reflects storage operation, while electricity price and frequency deviation may indicate system stress. The actual temporal state should be inferred from the provided time series representation.
Question: Which operational decision best matches the observed system state and dispatch objective?
Candidate Answers:
A. Increase battery charging and reduce dispatchable generation, because the system appears to have large surplus supply during the high-load phase.
B. Maintain the current dispatch without intervention, because electricity price and frequency deviation provide no useful signal about operating pressure.
C. Discharge the battery and increase dispatchable generation to relieve the high-load pressure while monitoring frequency stability.
D. Shut down renewable generation completely, because removing renewable output is the most direct way to stabilize price and frequency.
E. Reduce both thermal generation and battery output, because lower controllable supply will automatically reduce demand-side pressure.
Output: Please output the correct candidate answer.
Answer: *C. High demand and weak renewable output coincide with elevated electricity price and frequency deviation. Increasing dispatchable generation and battery discharge is therefore the appropriate response to relieve system pressure.*

Figure 16: Example of the Acting task.

Failure Case

Instruction: You are an expert in time series analysis and temporal reasoning. A temporal token representing the input time series is provided to the model. Please answer the multiple-choice question based on the temporal evidence, context, and candidate answers. Only one answer is correct.
Variable Information: T: Surface Temperature; H: Relative Humidity; P: Precipitation; W: Wind Speed; S: Solar Radiation.



Context: The data describe the operation of a regional weather monitoring system. The operator monitors surface temperature, relative humidity, precipitation, wind speed, and solar radiation to evaluate atmospheric conditions and weather evolution. Temperature and solar radiation reflect thermal conditions, while humidity and precipitation characterize moisture variation. The actual temporal state should be inferred from the provided time series representation.
Question: Which statement best describes the temporal relationship between surface temperature and solar radiation over the observed period?
Candidate Answers:
A. Surface temperature gradually increases throughout the period, while solar radiation decreases after an early peak, indicating a delayed thermal response to weakening radiation.
B. Solar radiation increases toward the middle of the period and then decreases, while surface temperature follows a similar pattern with a warming phase followed by cooling.
C. Surface temperature decreases during the period, while solar radiation reaches its maximum value near the end, suggesting reduced thermal input and delayed warming.
D. Solar radiation remains relatively stable with small fluctuations, while surface temperature exhibits strong variations unrelated to radiation changes.
E. Surface temperature and solar radiation both increase continuously throughout the period, showing a consistent upward relationship between thermal conditions and radiation intensity.
Answer: *E. Surface temperature and solar radiation exhibit a broadly aligned upward trend despite several local fluctuations.*

Ground Truth: *B.*
Failure Analysis: *The model captures the initial increasing trend but fails to recognize the subsequent reversal shared by surface temperature and solar radiation. It therefore overemphasizes the early local pattern and incorrectly predicts a continuous increase. This failure indicates difficulty in tracking coordinated phase transitions across variables over the full temporal horizon.*

Figure 17: Failure case of TSAlign on temporal relationship reasoning.