TWICE UPON A TIME: COMPLEMENTARY REPRESENTATIONS FOR TIME-SERIES REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Time Series Language Models (TSLMs) enable language-based understanding and reasoning over temporal data, yet a fundamental question remains: what information should their time-series representations preserve, and what variation should they factor out? Existing TSLMs typically expose the language model to a single representation, imposing a fixed inductive bias across downstream tasks. We instead introduce complementary representations as a design principle for time-series reasoning. Our model combines two visual views: line plots preserve absolute measurements and the observed coordinate frame, while normalized delay embeddings factor out positive amplitude rescaling and offsets while exposing temporal structure. Dedicated visual encoders integrate both views into a shared language model through visual tokens and multi-level feature fusion. Evaluation-time interventions verify that the two views exhibit distinct invariance profiles, while single-view ablations show that their combination is particularly beneficial beyond basic perception, with the large gains in prediction and reasoning. Built on Qwen3-VL-8B, our dual-view model improves overall TSRBench accuracy from 44.5% with line plots alone to 53.2%, substantially exceeding the strongest evaluated dedicated TSLM baseline at 36.7%. It further achieves 89.9% accuracy on TSExam and 86.8% numerical accuracy on multivariate ChatTS alignment. Together, these results demonstrate that complementary representations provide an effective design principle for general-purpose TSLMs, enabling models to exploit temporal structure without sacrificing access to the original measurement frame.

1 INTRODUCTION

Time Series Language Models (TSLMs) seek to extend the reasoning and interaction capabilities of large language models to sequential data, enabling models to answer questions, infer temporal relationships, forecast future behavior, and support decisions over time series (Xie et al., 2025; Langer et al., 2026; Guan et al., 2026; Yu et al., 2026a). A central challenge in this emerging paradigm is how temporal signals should be represented to the language model. Existing approaches serialize numerical values into text, learn dedicated time-series embeddings, or project features from specialized temporal encoders into the language-model representation space (Xie et al., 2025; Langer et al., 2026; Guan et al., 2026). As summarized in Table 1, these approaches typically commit to a single time-series representation, with absolute measurement scale either retained directly or provided separately as auxiliary metadata. Yet these representational choices impose fundamentally different inductive biases: representations that retain absolute amplitudes and coordinate information support quantitative readout, whereas normalized representations can suppress such information while emphasizing relative structure and temporal dynamics. Recent evaluations further show that simply rendering a time series as a line plot can already provide strong temporal understanding to general-purpose vision-language models (Yu et al., 2026a; Cai et al., 2024), suggesting that representation—not only model scale or training strategy—is a key determinant of time-series reasoning ability. This motivates a more fundamental question: *Can complementary representations improve general-purpose time-series reasoning?*

We hypothesize that general-purpose time-series reasoning benefits from exposing the model to *complementary views* of the same signal, each preserving different information and inducing different invariances. Rather than requiring a single representation to simultaneously retain all information relevant to downstream reasoning while suppressing nuisance variation, this formulation allows dis-

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

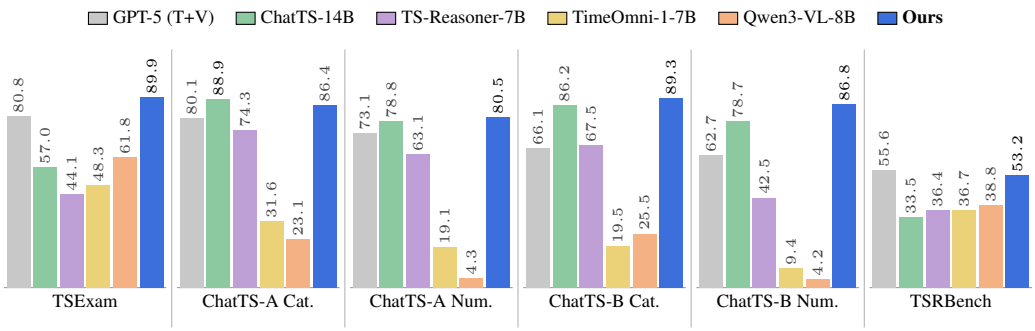


Figure 1: Cross-benchmark accuracy (%). ChatTS-A and ChatTS-B report categorical (Cat.) and numerical (Num.) alignment on the univariate and multivariate datasets, respectively. Bold marks the best result in each panel, excluding GPT-5 (T+V), shown in gray, which is a large proprietary model included as a reference rather than a size-matched comparison.

tinct views to provide different inductive biases. We instantiate this principle with two visual representations: a standard line plot and a normalized delay embedding. The line plot *preserves* absolute amplitudes, axis ranges, and measurement scales, providing the information required for questions grounded in observed values. In contrast, delay embeddings reconstruct a time series as a trajectory in phase space using time-lagged observations, revealing geometric structure associated with the underlying dynamics (Takens, 2006). We normalize this representation before rendering it as a two-dimensional view, thereby *factoring out* affine changes in amplitude while retaining the geometry of the temporal trajectory. Our central hypothesis is that combining these complementary views is particularly useful beyond basic perception, where reasoning, prediction, and decision-making require models to exploit temporal structure while remaining grounded in the observed measurements.

Building on this representational principle, we design a dual-tower TSLM in which each visual view is processed by a dedicated encoder. We retain Qwen3-VL's native vision encoder, initialized from SigLIP-2 (Tschannen et al., 2025), to process line plots, and introduce a dedicated DINOv3 encoder (Siméoni et al., 2026) for delay embeddings, allowing each view to develop a specialized visual representation before interacting with the language model. The resulting visual features are compressed into a compact set of tokens, 230 for the line plot and 64 for the delay embedding, and concatenated as input to the language model. Beyond this input-level fusion, we adopt DeepStack (Meng et al., 2024) to inject multi-level visual features into the early language-model layers, enabling the complementary representations to interact with language processing throughout the network. This architecture provides a compact realization of our central hypothesis while retaining the distinct inductive biases of the two visual views.

To test our representational hypothesis and evaluate the resulting dual-tower TSLM, we conduct experiments on TSRBench (Yu et al., 2026a), TSExam (Cai et al., 2024), and ChatTS (Xie et al., 2025). Built on Qwen3-VL-8B (Bai et al., 2025), our model achieves 53.2% overall accuracy on TSRBench, substantially narrowing the gap to GPT-5 with text and visual input (55.6%) while surpassing the strongest evaluated dedicated TSLM (36.7%). It also outperforms the compared general-purpose and specialized baselines on TSExam and ChatTS numerical alignment. To assess the contribution of the complementary view, we compare against a plot-only variant trained under the same setup. Adding the delay embedding raises overall TSRBench accuracy from 45.2% to 53.2%, an 8.0-percentage-point improvement, with gains in reasoning, prediction, and decision-making while leaving perception nearly unchanged. Controlled evaluation-time interventions further reveal distinct invariance profiles: the delay-only model maintains nearly unchanged accuracy under positive rescaling and offset shifts, whereas the plot-only model, whose representation retains the measurement frame, is substantially more sensitive to these transformations. Our contributions are threefold:

- **Complementary representations for time-series reasoning.** We establish complementary representation as a design principle for TSLMs, motivated by the observation that quantitative readout requires the original measurement frame, whereas reasoning about temporal dynamics benefits from factoring it out.

- **A dual-view time-series language model.** We propose a dual-tower architecture that combines line plots and normalized delay embeddings within a shared language model. Dedicated visual encoders, compact visual tokens, and multi-level feature fusion integrate temporal structure with the original measurement frame.
- **State-of-the-art TSLM performance.** With an 8B backbone, our model outperforms all evaluated dedicated TSLMs on TSRBench, achieving 53.2% accuracy versus 36.7% for the strongest baseline. It also leads the evaluated models on TSExam and multivariate ChatTS numerical alignment, achieving 89.9% and 86.8% accuracy, respectively.

Table 1: Comparison of time-series representations in existing TSLMs. The methods differ in how the time series is represented, whether the primary representation is normalized, and how absolute measurement-scale information is provided to the language model. Forecasting-oriented LLM methods are included for completeness; unlike TSLMs, they target numerical forecasting rather than general question answering and reasoning.

	Model	TS Representation	Normalization	Scale Information	Representation Views	Task Scope
Forecast.	PromptCast (Xue & Salim, 2023)	Text	None	In representation	Single	Forecasting
	LLMTime (Gruver et al., 2023)	Text	Per-series rescaling	Not provided	Single	Forecasting
	Time-LLM (Jin et al., 2024)	Reprogrammed patches	Per-series	Not provided	Single	Forecasting
TSLMs	TimeOmni-1 (Guan et al., 2026)	Text	None	In representation	Single	QA & reasoning
	ChatTS (Xie et al., 2025)	1D patches	Per-series	Auxiliary metadata	Single	QA & reasoning
	OpenTSLM (Langer et al., 2026)	1D patches	Per-series	Auxiliary metadata	Single	QA & reasoning
	TS-Reasoner (Yu et al., 2026b)	TSFM features	Per-series	Auxiliary metadata	Single	QA & reasoning
	Ours	Plot + delay embedding	View-dependent	Line plot	Complementary	QA & reasoning

2 Related Work

Time-Series Representations. Time series admit multiple representations that expose different properties of the underlying signal. Beyond the raw sequential domain, classical approaches transform temporal observations into alternative coordinate systems, including frequency and time-frequency representations, recurrence-based representations, and delay embeddings that reconstruct trajectories in phase space (Takens, 2006). A related line of work represents time series as images, enabling spatial models to operate on temporal structure. Wang & Oates (2015) introduced Gramian Angular Fields and Markov Transition Fields for encoding temporal sequences as two-dimensional images, demonstrating their utility for classification and imputation. More recently, Naiman et al. (2024) showed that image transformations based on delay embeddings and short-time Fourier transforms can provide effective interfaces between time series and modern vision-based diffusion models, supporting generation across both short and long sequences. Collectively, these works demonstrate that alternative representations can expose useful and distinct structure in temporal signals. Our work builds on this perspective but shifts the focus from selecting a representation for a particular predictive or generative task to *combining complementary representations for general-purpose time-series reasoning*: we jointly expose the model to views designed to retain different information and induce different invariances.

Time-Series Language Models. A growing body of work investigates how pretrained language models can be adapted to temporal data. Early approaches primarily focused on forecasting, either by serializing numerical observations as text (Gruver et al., 2023), prompting language models with textual descriptions of temporal sequences (Xue & Salim, 2023), or aligning and reprogramming time-series representations to exploit pretrained language-model representations (Zhou et al., 2023; Sun et al., 2024; Jin et al., 2024). More recent work has broadened this paradigm from task-specific prediction toward *Time Series Language Models* that accept time series as a modality and support natural-language understanding and reasoning. ChatTS (Xie et al., 2025) aligns learned time-series embeddings with an LLM through instruction tuning, while OpenTSLM (Langer et al., 2026) investigates both soft-prompt and cross-attention interfaces for reasoning over multivariate temporal signals. TimeOmni-1 (Guan et al., 2026) instead represents numerical sequences directly within the language-model token space and targets complex temporal reasoning, whereas TS-Reasoner (Yu et al., 2026b) aligns representations from a pretrained time-series foundation model with an LLM. In parallel, benchmark studies have shown that general-purpose vision-language models can reason surprisingly effectively over time series presented as line plots (Cai et al., 2024; Yu et al., 2026a),

providing an alternative to explicitly learned time-series interfaces. Despite these diverse strategies, existing TSLMs predominantly expose the language model to a single primary representation of each temporal signal. Our work instead treats representation as a source of complementary inductive biases, jointly providing views that deliberately preserve different information and exhibit different invariances.

3 METHOD

3.1 PROBLEM FORMULATION AND OVERVIEW

Let Q denote a natural-language query and $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ a collection of n input time series, where $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,m_i}) \in \mathbb{R}^{m_i}$ may have a different length and numerical range. Given $(Q, \mathcal{X})$, a Time Series Language Model (TSLM) generates a natural-language answer $A = (a_1, \ldots, a_{|A|})$ autoregressively:

$$P(A \mid Q, \mathcal{X}) = \prod_{j=1}^{|A|} P(a_j \mid a_{<j}, Q, \mathcal{X}) \,. \tag{1}$$

Rather than encoding each $\mathbf{x}_i$ through a single representation, we construct two complementary visual views, $\mathcal{V}(\mathbf{x}_i) = \{V_i^{\text{plot}}, V_i^{\text{delay}}\}$. The line-plot view V_i^{plot} retains the original numerical scale and measurement frame, whereas the normalized delay view V_i^{delay} factors out affine amplitude variation and represents the signal through its local temporal trajectories. As illustrated in Figure 2, the two views are processed by dedicated visual encoders and projected into a shared language-model representation space. Their visual tokens are jointly provided to a single Qwen3-VL-8B language-model backbone, with multi-level visual features additionally injected through DeepStack (Meng et al., 2024). The model is trained in two stages, first aligning the delay-embedding stream with the language model and subsequently instruction-tuning the language model for time-series reasoning.

3.2 COMPLEMENTARY VISUAL REPRESENTATIONS

Our representation design exposes each time series through two complementary views with different information-preserving properties. The first retains the original measurement frame through a standard line plot, while the second represents the series through overlapping temporal windows after removing its absolute amplitude and offset.

Line-Plot View. For each time series $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,m_i})$, we render the raw observations as a two-dimensional line plot without amplitude normalization (Figure 3a). Consequently, the plot retains the original value range and its position within the coordinate system, including the numerical values displayed along the vertical axis. This representation therefore preserves information about absolute amplitudes and differences in scale that would be removed by per-series normalization. For multivariate inputs, each series is rendered as its own plot rather than overlaid, so series with different ranges keep their own measurement frames.

Normalized Delay-Embedding View. Delay embeddings (Takens, 2006) describe a time series through its local histories: overlapping windows, each shifted slightly forward in time, are stacked side by side into an image (Naiman et al., 2024). Because windows overlap, each feature of the signal traces a diagonal line across the image: periodicity appears as parallel stripes, trends as smooth gradients, and local events as isolated streaks (Figure 3b). Temporal structure thus becomes visual texture that pretrained vision encoders can readily perceive.

Concretely, for each variate $\mathbf{x}$ of length m, we extract W overlapping windows of length ℓ with evenly spaced starts covering the whole series. The c-th window is the array starting at index $p_c = \text{round}\big(c\,\frac{m-\ell}{W-1}\big)$:

$$V_c = \big(x_{p_c},\, x_{p_c+1},\, \ldots,\, x_{p_c+\ell-1}\big), \qquad c = 0, \ldots, W-1, \tag{2}$$

where $\ell = \min(\max(\lfloor \rho m \rfloor, \ell_{\min}), \ell_{\max})$ is a fixed fraction ρ of the series length, so windows cover a comparable portion of short and long signals, with bounds $\ell_{\min}$ and $\ell_{\max}$ guarding against

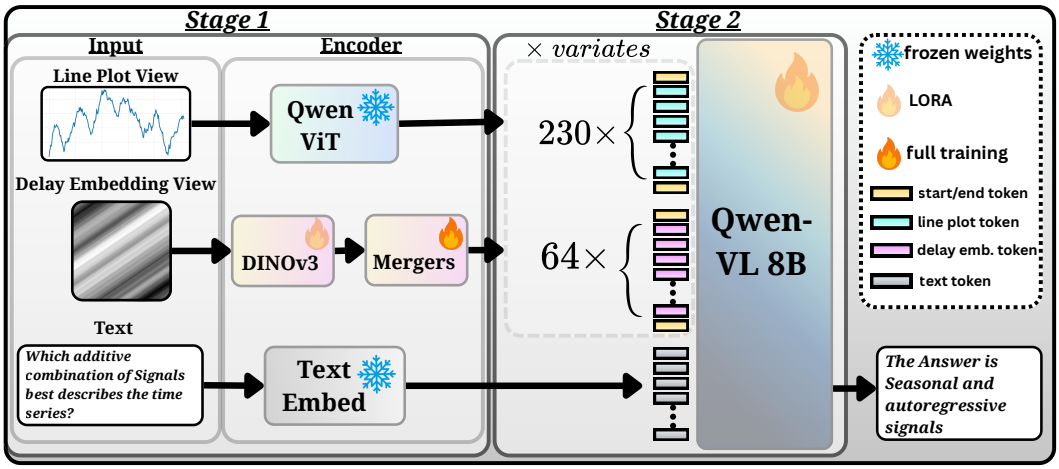


Figure 2: Overview of the Dual-View TSLM. Each visual span is wrapped by `vision_start` and `vision_end`. Stage I trains the DINOv3 LoRA and the merger. Stage II trains the LM LoRA. The Qwen vision encoder, the text embedding, and the LLM backbone stay frozen.

extreme lengths. For $m < \ell_{\min}$, we set $\ell = m$. Each window is linearly interpolated to H values, and the W resampled windows form the columns of the min–max normalized delay map $V^{\text{delay}} \in [0,1]^{H \times W}$. Every variate, regardless of its length, is thus represented by an image of the same size (hyperparameters in Appendix A.2).

Since resampling is linear and the map is min–max normalized, any transformation $a\mathbf{x} + b$ with $a > 0$ yields the same delay map. The line plot therefore retains the measurement frame, whereas the delay map exposes temporal structure independently of positive affine changes in amplitude.

3.3 Dual-View Architecture

Our architecture integrates the two views into a single Qwen3-VL-8B language-model backbone (Bai et al., 2025) through dedicated visual encoders (Figure 2).

Line-Plot Encoder. Line plots are processed by the native Qwen3-VL vision pipeline without modification, which merges spatial features over 2×2 neighboring patches and projects them to the language-model hidden dimension. This yields 230 plot tokens $\mathbf{Z}_i^{\text{plot}}$ per series, together with three intermediate feature maps for DeepStack fusion. The encoder remains frozen throughout training.

Delay-Embedding Encoder. For each time-series variate, we process its normalized delay map V^{delay} with a dedicated DINOv3 (Siméoni et al., 2026) ViT-L/16 encoder denoted by E_{delay}. Its spatial patch features are merged in 2×2 groups and mapped to the language-model hidden dimension by a two-layer projection module $M^{(0)}$, yielding 64 delay tokens $\mathbf{Z}^{\text{delay}} = M^{(0)}\big(E_{\text{delay}}(V^{\text{delay}})\big)$ per variate. To match the multi-level DeepStack interface of the plot stream, we additionally extract intermediate features $E_{\text{delay}}^{(k_i)}(V^{\text{delay}})$ after blocks k_i, where $(k_1, k_2, k_3) = (6, 12, 18)$, and merge and project each with a dedicated module $M^{(i)}$ of the same form as $M^{(0)}$, giving multi-level delay features $\mathbf{D}_i = M^{(i)}\big(E_{\text{delay}}^{(k_i)}(V^{\text{delay}})\big)$, $i \in \{1,2,3\}$, aligned one-to-one with the 64 delay tokens and injected into the language model as described below.

Dual-View Fusion. Both streams enter the language model in the same way. For each time series, $\mathbf{Z}_i^{\text{plot}}$ and $\mathbf{Z}_i^{\text{delay}}$ occupy separate visual spans, each framed by `<|vision_start|>` and `<|vision_end|>` tokens, alongside the textual query Q, so the two views interact within a single autoregressive language model rather than through separate heads or an ensemble. Beyond this input-level fusion, we extend DeepStack to the delay stream. Following the DeepStack mechanism of Qwen3-VL (Meng et al., 2024; Bai et al., 2025), the three multi-level features of each stream are added directly to the hidden states of the first three language-model layers, one feature per layer,

at the positions of that stream's own visual tokens. The plot and delay streams thus inject into the same layers but at disjoint token positions. Rendering and tokenization details are provided in Appendix B.1.

3.4 TRAINING

Following the staged training paradigm common in multimodal language models and recent TSLMs (Xie et al., 2025; Langer et al., 2026), we train our model, initialized from Qwen3-VL-8B (Bai et al., 2025), in two stages that separate the alignment of the new delay stream from task-level instruction tuning. In **Stage I (modality alignment)**, the Qwen3-VL vision encoder and the language model remain frozen, and we train LoRA adapters (Hu et al., 2022) on DINOv3 together with the projection $M^{(0)}$, which maps the final DINOv3 patch features to delay tokens, and the projections $\{M^{(i)}\}_{i=1}^{3}$, which map the intermediate DINOv3 features to the multi-level DeepStack features. This stage is trained on captioning data, encouraging the delay stream to produce representations that the frozen language model can interpret. In **Stage II (instruction tuning)**, the DINOv3 adapters are merged, the entire visual pathway is frozen, and LoRA is applied to the language model, which is trained on time-series question answering to reason jointly over both views. Both stages use the standard next-token prediction loss computed over the answer tokens only. The objective and hyperparameters are given in Appendix B.2.

4 TRAINING DATA

We construct a dedicated training corpus for each of the two stages described in Section 3.4. Stage I uses caption supervision to align the new delay-embedding stream with the language model, whereas Stage II uses a heterogeneous instruction corpus to train perception, numerical readout, forecasting, and temporal reasoning. Most time series are generated procedurally, with labels obtained directly from generator parameters or exact functions of the realized signal rather than from a teacher model. The only real-world time-series sources are CaTS-Bench in Stage I and a small subset of sports win-probability series from GAMETime (Tan et al., 2025) used for abductive reasoning in Stage II. Table 2 summarizes the data composition.

Stage I: Caption Alignment. The alignment corpus contains 55,696 examples from three sources. CaTS-Bench (Zhou et al., 2026) contributes real-world short time series from diverse physical and socioeconomic domains together with reference descriptions. To provide broader attribute coverage, we additionally construct 32,000 ChatTS-style captions from the ChatTS time-series generator (Xie et al., 2025). Trend, seasonality, noise, and local events are sampled from the generator's attribute space, and captions are rendered directly from those attributes. Finally, we construct 7,701 structural captions by generating series with our TimeSeriesExam fork (Cai et al., 2024) and deterministically rendering one-sentence descriptions from the underlying generative components, parameters, noise processes, and injected anomalies. The three sources are sampled with equal probability during Stage I, balancing real-world diversity, free-form synthetic descriptions, and tightly controlled constructive supervision.

Table 2: Training-data composition. Rows denote distinct examples available to the corresponding training stage. Stage I sources are sampled with equal probability; Stage II uses a balanced mixture described in the text.

Stage	Source	Origin	Rows	Supervision
I	CaTS-Bench	Real (Zhou et al., 2026)	15,995	Caption
I	ChatTS-style captions	Synthetic (Xie et al., 2025)	32,000	Caption
I	Structural captions	Synthetic, in-house	7,701	Caption
II	ChatTS `ift`	Synthetic (Xie et al., 2025)	6,373	Free-form QA
II	ChatTS `sft`	Synthetic (Xie et al., 2025)	44,802	Free-form QA
II	TSExam perception	Synthetic, in-house	20,640	MCQ
II	Numerical corpus	Synthetic, in-house	11,000	Scalar
II	Reasoning corpus	Synthetic + real sports	100,000	MCQ / rationale+MCQ

Stage II: Instruction Tuning. The instruction corpus contains 182,815 examples spanning complementary task families. We include the released ChatTS `ift` and `sft` training sets (Xie et al., 2025) for open-ended univariate and multivariate question answering. Using our TimeSeriesExam fork, we further generate the TSExam perception set, 20,640 multiple-choice questions covering pattern recognition, anomaly detection, similarity, noise, and causality, together with an 11,000-example numerical corpus of open-ended questions whose answers are exact statistics or dynamical quantities computed from the realized series. The largest component is an in-house 100,000-example reasoning corpus designed to develop broad reasoning capabilities over time series. It spans numerical, deductive, inductive, abductive, causal, etiological, temporal-relation, forecasting, and event-prediction tasks, with examples generated from procedural rules, dynamical systems, and domain-specific simulators. Because instruction-tuned models are sensitive to the task interface, we align the mixture of reasoning tasks and their question–answer formats with the capabilities evaluated by TSRBench (Yu et al., 2026a); the underlying time series, questions, and answers are generated independently and no benchmark instances are used for training. For tasks requiring explicit derivation, supervision additionally includes deterministic rationales constructed from the simulator state, while the remaining tasks use direct multiple-choice supervision.

Gold-by-Construction Supervision and Benchmark Separation. For all in-house synthetic data, supervision is derived from known generative parameters or exact computations on the realized signal. Captions describe the sampled components, multiple-choice labels follow directly from the generating mechanism, numerical targets are exact signal reductions, and reasoning rationales are rendered from structured intermediate facts. This design avoids using an external language model to infer labels for the synthetic data. Importantly, no TSRBench or TimeSeriesExam evaluation examples are included in training. The TSExam perception set, structural captions, and numerical corpus are generated with our fork of the public generator and contain freshly sampled series whose lengths differ from those in the benchmark. Similarly, although the reasoning corpus covers the reasoning task families and question–answer formats evaluated by TSRBench, all training instances are generated independently and no benchmark examples are used for training. We additionally verified the reasoning corpus against the benchmark for exact series overlap, shared sports-game instances, and exact question–answer matches. ChatTS evaluation data are likewise excluded; only the released training splits are used. Additional generator details, per-task statistics, target formats, and held-out splits are provided in Appendix D.

5 Experiments and Results

5.1 Experimental Setup

We evaluate on three benchmarks that together probe time-series understanding from low-level perception to high-level reasoning. **TSRBench** (Yu et al., 2026a), our primary benchmark, comprises 4,125 multiple-choice questions over 15 tasks spanning Perception, Reasoning, Prediction, and Decision-Making. **TSExam** (Cai et al., 2024) contains 746 multiple-choice questions on pattern recognition, noise understanding, anomaly detection, similarity, and causality, and **ChatTS** (Xie et al., 2025) measures both fine-grained categorical and numerical alignment and higher-level reasoning over univariate and multivariate series. We compare against two families of models. General-purpose vision-language models receive the series as line plots: Qwen3-VL-8B (Bai et al., 2025), our own backbone, isolates the effect of our design and training, while GPT-5 serves as a substantially larger proprietary reference rather than a size-matched competitor. We further compare against representative dedicated TSLMs: ChatTS-14B (Xie et al., 2025), TS-Reasoner-7B (Yu et al., 2026b), and TimeOmni-1-7B (Guan et al., 2026). Table 3 reports task-level TSRBench accuracy together with our representation ablations, and Figure 1 summarizes performance across all three benchmarks. Complete task-level results and evaluation procedures, including the source of each baseline result, are provided in Appendix C.

5.2 Do Complementary Representations Improve Temporal Reasoning?

Our central hypothesis is that exposing the language model to complementary representations provides benefits that cannot be obtained from either view alone. We therefore compare the full dual-view model against *Plot-only* and *Delay-only* variants, as well as two same-view controls that retain

Table 3: TSRBench accuracy (%) across all 15 tasks. Rows are grouped into general-purpose models, dedicated TSLMs, and representation ablations. In the Overall column, **bold** and underline mark the best and second-best results, respectively. Gray rows denote large proprietary models included as reference points rather than size-matched comparisons and are excluded from this ranking. Full task descriptions and results are provided in Appendix C.

		Perception				Reasoning							Prediction		Decision		
	Model	PR	NU	AD	CA	ER	CD	AR	TR	NR	DR	IR	TSF	EP	QuaDM	QuanDM	**Overall**
General	GPT-5 (V)	83.6	72.4	69.8	73.5	34.6	37.0	81.3	68.1	64.0	48.0	57.0	38.8	71.9	26.9	30.0	52.4
	GPT-5 (T+V)	84.9	72.4	61.2	70.8	38.9	50.3	93.3	68.8	72.0	44.0	58.0	37.8	78.1	31.9	34.3	55.6
	Qwen3-VL-8B (V)	54.7	57.5	62.8	65.5	24.3	31.7	52.0	23.1	33.2	19.2	29.0	51.7	30.6	36.1	29.0	38.9
TSLMs	ChatTS-14B	50.7	50.6	46.5	55.8	21.7	34.3	51.3	24.4	30.5	23.6	25.0	19.2	52.2	29.3	27.7	33.5
	TS-Reasoner-7B	53.1	56.3	48.1	50.4	28.9	24.3	57.3	26.9	37.2	23.6	24.0	31.7	55.0	32.8	21.7	36.4
	TimeOmni-1-7B	55.0	59.8	41.9	64.6	28.0	35.3	46.7	24.4	31.5	22.8	34.0	30.0	49.4	34.0	30.3	36.7
Ablations	Delay only	57.4	65.5	69.8	76.1	30.3	51.7	55.3	24.4	38.5	26.8	21.0	49.0	20.3	34.9	19.3	40.5
	Plot only	87.3	94.3	77.5	93.8	29.7	52.0	57.3	18.1	36.5	27.2	23.0	46.2	21.7	39.1	33.3	45.2
	Delay+Delay	67.9	62.1	79.1	82.3	32.3	40.3	64.7	34.4	40.2	26.8	19.0	54.6	15.8	33.1	21.0	42.6
	Plot+Plot	84.1	94.3	80.6	87.6	40.0	43.0	60.7	24.4	32.2	26.4	27.0	46.2	28.6	32.8	28.3	44.8
	Ours	85.4	94.3	78.3	92.9	43.1	51.7	56.0	28.7	35.5	29.2	34.0	50.4	84.2	32.8	43.0	**53.2**

the complete dual-tower architecture but feed both towers the same representation. In *Plot+Plot*, the DINOv3 tower receives the line plot in place of the delay map; in *Delay+Delay*, the Qwen3-VL tower receives the delay map in place of the line plot. Both controls keep the same encoders, adapters, per-tower token budgets, and two-stage training procedure as our model, so any difference from the full model reflects whether the two streams carry distinct or identical views, rather than differences in architecture or capacity. Table 3 reports results across all 15 TSRBench tasks, together with representative dedicated TSLMs and general-purpose models.

The results show that the benefit of complementary representations is concentrated beyond basic perception. Plot only already achieves 87.4% on Perception, and adding the delay view leaves this essentially unchanged at 86.4%. In contrast, the dual-view model improves Reasoning from 35.8% to 40.1%, Prediction from 38.0% to 61.7%, and Decision-Making from 36.4% to 37.6%, raising overall accuracy from 45.2% to **53.2%**. The task-level results further show that the contribution of the complementary view varies across tasks, rather than producing a uniform improvement across the benchmark. Importantly, simply duplicating a view does not reproduce the gain: Plot+Plot reaches 44.8% overall and Delay+Delay reaches 42.6%, compared with our 53.2%. This suggests that the improvement is not explained merely by providing a second visual stream, but is associated with combining representations that preserve different properties of the underlying signal.

Our model also exceeds the evaluated dedicated TSLMs, whose strongest overall TSRBench result is 36.7%. For context, Table 3 additionally reports GPT-5 results; these models are substantially larger proprietary systems and are therefore not size-matched comparisons. They nevertheless provide a useful reference point: our model reaches 53.2% overall, surpassing GPT-5 with visual input (52.4%) and approaching GPT-5 with text and visual input (55.6%), while exhibiting a markedly different task-level performance profile.

5.3 How Do the Representations Respond to Distribution Shifts?

To directly probe the inductive biases induced by the visual representations, we apply controlled transformations to TSRBench inputs at evaluation time. We consider affine changes to amplitude and offset, observational noise and smoothing, and changes in sampling horizon. Tasks whose gold targets depend intrinsically on absolute coordinate values or sampling rates are excluded from this analysis, leaving 2,020 items across invariant reasoning, perception, and event prediction tasks. Table 4 reports the resulting accuracy under each transformation, together with clean performance for reference.

The interventions reveal distinct rather than uniformly superior robustness profiles. Consistent with its per-series normalization, the delay-only representation is nearly invariant to the measured affine transformations: its clean accuracy of 44.7% changes only slightly under rescaling and offset shifts. Our model remains sensitive to affine transformations because its line-plot stream continues to en-

Table 4: Evaluation-time interventions on scale- and sampling-invariant TSRBench subsets (2,020 items; excludes NR, DR, IR, TSF, and decision tasks). Each entry reports accuracy (%).

		Affine					Noise / Smoothing				Sampling	
Model	Clean	$\times 0.1$	$\times 10$	$+5\sigma$	-5σ	z-score	20 dB	10 dB	0 dB	Smooth	256	2048
Plot only	52.7	58.2	49.6	48.9	49.2	47.5	50.6	47.4	43.2	50.1	51.6	52.9
Delay only	44.7	43.4	44.9	44.9	44.1	44.3	42.3	38.3	35.3	40.7	45.1	42.8
Ours	**66.5**	**63.4**	**64.8**	**62.9**	**65.0**	**60.7**	**65.4**	**62.4**	**56.6**	**60.7**	**64.8**	**66.8**

code the observed measurement frame, while maintaining substantially higher absolute accuracy than Plot only across all reported interventions. Its sensitivity also depends on the type of shift: noise and smoothing degrade both Plot only and our model, whereas our model changes relatively little under changes in sampling horizon. Thus, combining the two views does not impose a single invariance profile or provide uniform robustness to every perturbation. Instead, the results support the intended representational distinction: the views exhibit different sensitivities, allowing the model to retain access to the original measurement frame while also incorporating a representation that factors out selected amplitude variation.

5.4 Cross-Benchmark Generalization

Finally, we evaluate whether the gains observed on TSRBench transfer to benchmarks emphasizing different aspects of time-series understanding. Figure 1 summarizes overall TSExam accuracy, categorical and numerical alignment on the univariate (A) and multivariate (B) ChatTS datasets, and TSRBench overall accuracy; per-task results and ChatTS reasoning evaluations are in Appendix C. Our model attains 89.9% accuracy on TSExam, exceeding the compared general-purpose and dedicated TSLM baselines. On ChatTS, it achieves 80.5% numerical accuracy on univariate sequences and 86.8% on multivariate sequences, with the largest advantage appearing in the multivariate setting. Together with the TSRBench results, these evaluations indicate that the benefit of the dual representation is not confined to a single benchmark: the model remains strong across perceptual, numerical, and broader temporal reasoning tasks, despite using a Qwen3-VL-8B backbone.

6 Discussion

Time-series reasoning requires both numerical readout and understanding of temporal dynamics, yet these capabilities favor different representational properties. This tension motivated our central question: can complementary representations improve general-purpose time-series reasoning? We establish complementary representation as a design principle for TSLMs and instantiate it in a dual-view model that integrates measurement-preserving line plots and normalized delay embeddings through dedicated visual encoders and a shared language model. The resulting model raises TSRBench accuracy from 45.2% for the plot-only variant to 53.2%, surpassing all evaluated dedicated TSLMs. The gains are concentrated beyond basic perception and are not reproduced when both encoders receive line plots or both receive delay embeddings. The model also leads the evaluated baselines on TSExam and multivariate ChatTS numerical alignment, achieving 89.9% and 86.8% accuracy, respectively, demonstrating strong performance across perceptual, numerical, and broader temporal reasoning tasks.

The STFT comparison in Appendix C.5 further shows that the preferred secondary representation varies across evaluations: delay embeddings perform better on TSRBench, while STFT provides small advantages on ChatTS numerical alignment and average reasoning accuracy. This suggests that representation choice should reflect the capabilities a model must support. The intervention analysis likewise distinguishes access to an invariant view from invariance of the combined model, which retains both normalized structure and absolute measurements. Future work could explore additional representations and encoder combinations, including more than two views and query-dependent mechanisms for combining them.

AI USE STATEMENT

In this work, we used generative AI tools for generating synthetic datasets, specifically for drafting the question templates of our in-house training corpora. We have not used generative AI tools for developing theoretical models or conceptual frameworks, formulating or proving mathematical claims, proposing or refining hypotheses, designing research methodology or experiments, implementing methods, cleaning or reformatting datasets, qualitative data analysis, or interpreting results, and translation is not applicable to this work. Additionally, we used generative AI tools for aiding and polishing the writing of the manuscript. We have reviewed all AI-assisted work, and all labels are computed deterministically rather than by an LLM. We take responsibility for the final content of this work.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

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

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters. In *Advances in Neural Information Processing Systems*, 2023.

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. In *The Fourteenth International Conference on Learning Representations*, 2026.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. 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 *International Conference on Learning Representations*, 2024.

Patrick Langer, Thomas Kaar, Max Rosenblattl, Maxwell A Xu, Winnie Chow, Martin Maritsch, Robert Jakob, Ning Wang, Juncheng Liu, Aradhana Verma, Brian Han, Daniel Seung Kim, Henry Chubb, Scott R. Ceresnak, Aydin Zahedivash, Alexander T Sandhu, Fatima Rodriguez, Daniel McDuff, Elgar Fleisch, Oliver Oppers Aalami, Filipe Barata, and Paul Schmiedmayer. OpenTSLM: Time-series language models for reasoning over multivariate medical text- and time-series data. In *Forty-third International Conference on Machine Learning*, 2026.

Lingchen Meng, Jianwei Yang, Rui Tian, Xiyang Dai, Zuxuan Wu, Jianfeng Gao, and Yu-Gang Jiang. Deepstack: Deeply stacking visual tokens is surprisingly simple and effective for lmms. *Advances in Neural Information Processing Systems*, 37:23464–23487, 2024.

Ilan Naiman, Nimrod Berman, Itai Pemper, Idan Arbiv, Gal Fadlon, and Omri Azencot. Utilizing image transforms and diffusion models for generative modeling of short and long time series. *Advances in Neural Information Processing Systems*, 37:121699–121730, 2024.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seung Eun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3. *Trans. Mach. Learn. Res.*, 2026.

Chenxi Sun, Yaliang Li, Hongyan Li, and Shenda Hong. Test: Text prototype aligned embedding to activate llm's ability for time series. In *International Conference on Learning Representations*, 2024.

Floris Takens. Detecting strange attractors in turbulence. In *Dynamical Systems and Turbulence, Warwick 1980: proceedings of a symposium held at the University of Warwick 1979/80*, pp. 366–381. Springer, 2006.

Mingtian Tan, Mike A Merrill, Zack Gottesman, Tim Althoff, David Evans, and Tom Hartvigsen. Inferring events from time series using language models. *arXiv preprint arXiv:2503.14190*, 2025.

Michael Tschannen, Alexey A. Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier J. Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Zhiguang Wang and Tim Oates. Imaging time-series to improve classification and imputation. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pp. 3939–3945. AAAI Press, 2015.

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. In *Proceedings of the VLDB Endowment, 2025*, 2025.

Hao Xue and Flora D Salim. Promptcast: A new prompt-based learning paradigm for time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):6851–6864, 2023.

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*, 2026a.

Fangxu Yu, Hongyu Zhao, and Tianyi Zhou. TS-reasoner: Aligning time series foundation models with LLM reasoning. *Transactions on Machine Learning Research*, 2026b. ISSN 2835-8856. URL https://openreview.net/forum?id=d6TD0f2xXq.

Luca Zhou, Pratham Yashwante, Marshall Fisher, Alessio Sampieri, Zihao Zhou, Fabio Galasso, and Rose Yu. CaTS-bench: Can language models describe time series? In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 34479–34519, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.1722. URL https://aclanthology.org/2026.findings-acl.1722/.

Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained lm. In *Advances in Neural Information Processing Systems*, 2023.

Table of Contents for Appendix

A Time Series Representations 13
A.1 Visualization of Complementary Representations 13
A.2 Delay-Map Construction Details 13

B Model and Training Details 14
B.1 Implementation Details 14
B.2 Training Objective and Hyperparameters. 14

C Additional Experimental Results 14
C.1 Cross-Benchmark Representation Ablations 15
C.2 TSRBench Category-Level Ablations 15
C.3 Detailed ChatTS Results 15
C.4 Complete Intervention Results 16
C.5 Alternative Secondary Representations 16
C.6 Visual-Token Budget and Same-View Controls 17
C.7 Complete Benchmark Results 18

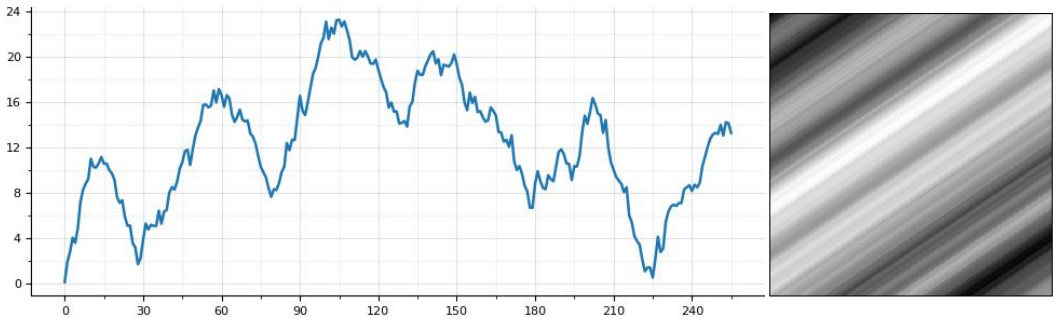
D Training Data Details 21
D.1 Training Corpus Overview 21
D.2 Stage I: Modality-Alignment Data 21
D.3 Stage II: Instruction-Tuning Data. 22
D.4 Reasoning Corpus Specification 22
D.5 Numerical Corpus Specification 24

APPENDIX

A TIME SERIES REPRESENTATIONS

A.1 VISUALIZATION OF COMPLEMENTARY REPRESENTATIONS

Figure 3 illustrates the two visual representations constructed from the same input time series. The line-plot view (left) preserves the original measurement frame, including absolute amplitudes and axis information. In contrast, the normalized delay-embedding view (right) represents overlapping temporal trajectories after per-series normalization, thereby removing offsets and positive amplitude rescaling while emphasizing temporal structure. The two views are provided to their respective visual encoders as described in Section 3.2.



(a) Line-plot view (b) Delay-embedding view

Figure 3: Dual-view visual representations of a time-series variate. (a) The line plot retains the original measurement frame and axis values. (b) The normalized delay map arranges overlapping windows as columns, so recurring structure appears as diagonal stripes.

A.2 DELAY-MAP CONSTRUCTION DETAILS

Hyperparameters. Table 5 lists the hyperparameters of the delay map (Eq. 2). With these values, the window length ℓ equals the series length for $m < 48$, is fixed at $\ell_{\min} = 48$ for $48 \leq m < 80$, scales as $\lfloor 0.6m \rfloor$ for $80 \leq m \leq 320$, and is capped at $\ell_{\max} = 192$ for longer series.

Table 5: Hyperparameters of the normalized delay-embedding view.

Symbol	Description	Value
ρ	Window length as a fraction of series length	0.6
$\ell_{\min}$	Minimum window length	48
$\ell_{\max}$	Maximum window length	192
W	Number of windows (map width)	256
H	Resampled window length (map height)	256

Interpolation. Interpolation is applied only within each window: the ℓ values of V_c are placed on a uniform grid over $[0, 1]$ and linearly interpolated at H uniformly spaced points. Window start indices p_c are rounded to integers, so no interpolation is performed along the time axis. Each entry of the map is therefore an interpolated sample of the original series rather than a resized pixel. The resulting $H \times W$ matrix is min–max normalized, replicated across three channels, and normalized with the DINOv3 preprocessing statistics as a floating-point tensor, without quantization or image resizing.

Relation to the standard delay embedding. The classical delay embedding (Takens, 2006; Naiman et al., 2024) arranges windows of fixed dimension n as columns with a fixed integer skip s between their start indices, yielding an $n \times \lceil (m-n)/s \rceil$ matrix that is zero-padded to the network input size. Our construction corresponds to $n = \ell$ with a length-dependent fractional skip

$(m-\ell)/(W-1)$. Instead of padding, both dimensions are fixed: the number of columns to W and the number of rows to H through resampling. When $m-\ell < W-1$, the skip is below one sample and neighboring columns may repeat. When $\ell = m$, all windows start at $p_c = 0$, and the map reduces to W identical copies of the resampled series.

B Model and Training Details

B.1 Implementation Details

Line-plot tokens. Each chart is rendered at 768×320 px and ingested by the Qwen3-VL vision encoder at 736×320 px, corresponding to 46×20 patches. After 2×2 merging, this yields $N_i^{\text{plot}} = 230$ visual tokens per series. The Qwen3-VL tokenizer frames these with two boundary tokens (<|vision_start|> and <|vision_end|>) enclosing 230 <|image_pad|> slots, for an exact budget of 232 tokens per series. The native encoder additionally provides three intermediate feature maps, which Qwen3-VL injects into the language model through DeepStack.

Delay tokens. For each time-series variate, the delay map $V^{\text{delay}} \in [0,1]^{256\times256}$ is replicated across three channels and normalized using the DINOv3 image preprocessing statistics before being passed to the encoder. We retain the spatial patch tokens and discard the class and register tokens. The resulting 16×16 patch grid is merged into an 8×8 grid of 2×2 groups and projected by the two-layer module $M^{(0)}$, giving $N^{\text{delay}} = 64$ visual tokens per variate; with the two boundary tokens, the exact budget is 66 tokens per variate.

Multi-level delay features. The intermediate delay features used for DeepStack fusion are computed as

$$\mathbf{D}_i = M^{(i)}\big(E^{(k_i)}_{\text{delay}}(V^{\text{delay}})\big), \qquad (k_1,k_2,k_3) = (6,12,18), \tag{3}$$

where $E^{(k_i)}_{\text{delay}}$ denotes the output of transformer block k_i and each $M^{(i)}$ applies the same 2×2 merge and two-layer projection as $M^{(0)}$, with separate parameters. $\mathbf{D}_1$, $\mathbf{D}_2$, and $\mathbf{D}_3$ are added to the hidden states of the first, second, and third language-model layers, respectively, at the delay-token positions.

B.2 Training Objective and Hyperparameters

For both stages, letting θ_s denote the parameters optimized at stage s, we minimize the autoregressive language-modeling loss

$$\mathcal{L}_{\text{LM}}(\theta_s) = -\sum_{j=1}^{|A|} \log P_{\theta_s}\left(a_j \mid a_{<j}, Q, \mathcal{X}\right) , \tag{4}$$

where the loss is computed only over target answer tokens; textual prompts, visual tokens, and padding positions are masked. In Stage I, θ_1 comprises the DINOv3 LoRA parameters, the delay-token projection $M^{(0)}$, and the three DeepStack projections $\{M^{(i)}\}_{i=1}^{3}$ applied to the features of blocks $(k_1,k_2,k_3) = (6,12,18)$. In Stage II, θ_2 comprises the LoRA parameters of the language model. In Stage I, we apply LoRA with rank $r = 64$ and scaling parameter $\alpha = 128$ to the DINOv3 backbone, while the projections $\{M^{(i)}\}_{i=0}^{3}$ are trained in full. In Stage II, the Stage I adapter is merged into DINOv3, and we apply LoRA with rank $r = 32$, $\alpha = 64$, and dropout 0.05 to all attention and MLP projections of the language model (q, k, v, o, gate, up, and down).

C Additional Experimental Results

This appendix provides additional results supporting the analyses in the main paper. We first summarize the representation ablations across benchmarks, followed by detailed ChatTS results, the complete intervention study, and experiments with an alternative frequency-domain representation. Together, these experiments examine whether the gains of the dual-view model extend beyond TSR-Bench, whether they can be reproduced by simply duplicating a representation, and whether the choice of the complementary representation matters.

Table 6: Cross-benchmark representation ablations. ChatTS-A and ChatTS-B report categorical (Cat.) and numerical (Num.) alignment accuracy. ChatTS Reason. reports average accuracy across the four reasoning evaluations. TSRBench reports overall accuracy. Plot+Plot and Delay+Delay feed the same representation to both towers while matching the architecture and visual-token budget of our model.

	TSExam	**ChatTS-A**		**ChatTS-B**		**ChatTS**	**TSRBench**
Model	Overall	Cat.	Num.	Cat.	Num.	Reason.	Overall
Delay only	0.749	0.782	0.290	0.806	0.308	0.596	0.405
Plot only	**0.899**	0.861	0.781	**0.895**	0.864	0.613	0.452
Delay+Delay	0.790	0.858	0.424	0.848	0.348	0.604	0.426
Plot+Plot	0.865	**0.873**	**0.805**	0.877	0.859	0.586	0.448
Ours	**0.899**	0.864	**0.805**	0.893	**0.868**	**0.615**	**0.532**

Table 7: TSRBench category-level accuracy (%) for the representation ablations. Overall accuracy is computed using the benchmark's complete task-level evaluation.

Model	Perception	Reasoning	Prediction	Decision	Overall
Delay only	63.7	36.5	39.4	27.6	40.5
Plot only	**87.4**	35.8	38.0	36.4	45.2
Delay+Delay	71.6	37.0	41.7	27.4	42.6
Plot+Plot	85.3	36.3	40.3	30.7	44.8
Ours	86.4	**40.1**	**61.7**	**37.6**	**53.2**

C.1 Cross-Benchmark Representation Ablations

Table 6 extends the representation ablations beyond TSRBench. We compare the full dual-view model with single-view variants (*Plot-only* and *Delay-only*) and duplicated-view controls (*Plot+Plot* and *Delay+Delay*). The duplicated-view controls keep the full dual-tower architecture, per-tower token budgets, and training procedure of our model, but feed both towers the same representation; they therefore test whether the improvement can be reproduced by a second visual stream alone (Appendix C.6).

The results show that the contribution of the complementary representation depends on the benchmark. Plot only is already strong on TSExam and ChatTS, whereas Delay only is substantially weaker on numerical alignment. Combining the representations provides modest improvements over Plot only on ChatTS numerical alignment and reasoning, while producing a substantially larger improvement on the broader TSRBench evaluation. Importantly, neither duplicated-view control reproduces the overall TSRBench gain of the complementary pair, despite matching its architecture and visual-token budget.

C.2 TSRBench Category-Level Ablations

Although Table 3 reports all 15 individual TSRBench tasks, Table 7 provides the corresponding category-level aggregation. The dual-view model leaves the already strong perception performance of Plot only essentially unchanged, while improving the aggregate results on reasoning, prediction, and decision-making. The same-view controls again remain close to their corresponding single-view variants in overall accuracy.

C.3 Detailed ChatTS Results

We further examine the representation ablations on the ChatTS reasoning evaluation. Table 8 reports the four reasoning dimensions individually rather than only their average. The results show that no representation uniformly dominates every reasoning dimension. The full model achieves the highest average accuracy, while individual single-view variants remain stronger on some dimensions. This

Table 8: ChatTS reasoning accuracy for the single-view and dual-view models on inductive, deductive, and causal reasoning and two-series multiple-choice questions (MCQ2). Inductive reasoning is scored with the RAGAS pipeline; Average reports the mean over the four tasks.

Model	Induct.	Deduct.	Causal	MCQ2	Average
Delay only	0.403	0.605	0.857	0.520	0.596
Plot only	0.441	0.548	0.851	0.610	0.613
Ours	0.479	0.535	0.826	0.620	0.615

variation is consistent with the central premise that the representations provide different inductive biases rather than one view uniformly subsuming the other.

The alignment results in Table 6 provide a complementary picture. Delay only performs considerably worse on numerical alignment than on categorical alignment, whereas Plot only retains strong performance on both. The dual representation preserves the strong categorical performance of Plot only while improving numerical accuracy from 0.781 to 0.805 on ChatTS-A and from 0.864 to 0.868 on ChatTS-B.

C.4 Complete Intervention Results

Table 9 expands the intervention analysis from Section 5.3 by reporting both absolute accuracy and the percentage-point change relative to clean performance. Unlike the main analysis, which is restricted to the 2,020 scale- and sampling-invariant items, this table covers all 15 TSRBench tasks. We additionally report two temporal-symmetry transformations, temporal reversal and sign inversion. Unlike the other interventions, these can legitimately change the gold answer (e.g., reversing an increasing trend), so they illustrate that invariance is not always desirable: a model that is insensitive to these transformations may be ignoring information the task requires.

The full results make the distinct sensitivity profiles particularly clear. Delay only changes by at most 0.8 percentage points under the five affine interventions, including z-score normalization, which is consistent with the invariance induced by its per-series normalization. In contrast, Plot only and our model remain sensitive to transformations of the measurement frame. Delay only and Plot only also exhibit the smallest average degradation under noise and smoothing. Our model is not uniformly the most invariant representation, but retains substantially higher absolute accuracy across all reported interventions. Under changes in sampling horizon, its category-average accuracy is virtually unchanged from clean performance.

C.5 Alternative Secondary Representations

The main model uses a normalized delay representation as the complementary view. To test whether the observed gains follow simply from introducing any second representation, we additionally replace the delay view with a short-time Fourier transform (STFT) representation while retaining the line-plot stream. We evaluate this alternative both in a controlled setting and after full training.

Controlled representation comparison. We first compare delay and STFT representations under the same controlled TSExam setting. Table 10 reports overall accuracy for the secondary representation alone and when combined with the line-plot representation. Both raw and normalized STFT variants are included. The results show that the three secondary representations are similar when combined with the line plot on this evaluation, indicating that TSExam alone does not strongly distinguish among them.

Robustness to evaluation-time transformations. We further apply deterministic transformations to the TSExam inputs of the single-representation ("Only") models from Table 10. As shown in Table 11, the representations exhibit different robustness profiles. The delay embedding and the z-normalized STFT are nearly invariant to scaling and offsets, whereas the raw STFT degrades under mean shifts. The raw STFT is the most robust to additive noise, while the delay embedding is the most robust to smoothing and to resampling at a longer horizon, but the most sensitive to temporal reversal and sign inversion.

Table 9: Evaluation-time affine and dynamical interventions across the full TSRBench benchmark (all 15 tasks, including forecasting). For each model, we report clean accuracy and the percentage-point change (Δ) relative to clean performance. Daggers (†) mark non-invariance interventions for which gold targets legitimately change. Bold values indicate the smallest absolute shift ($|\Delta|$) in each row.

		Plot only		**Ours**		**Delay only**	
Intervention Type	Transformation	Acc.	Δ vs. Clean	Acc.	Δ vs. Clean	Acc.	Δ vs. Clean
Baseline	Clean	0.452	—	0.532	—	0.405	—
Affine Invariance	Scale $\times 0.1$	0.467	$+1.5$	0.496	-3.6	0.407	$\mathbf{+0.2}$
	Scale $\times 10$	0.439	-1.4	0.522	-1.0	0.411	$\mathbf{+0.6}$
	Offset $+5\sigma$	0.431	-2.1	0.511	-2.2	0.397	$\mathbf{-0.8}$
	Offset -5σ	0.430	-2.2	0.520	-1.3	0.400	$\mathbf{-0.6}$
	z-score	0.418	-3.4	0.499	-3.3	0.397	$\mathbf{-0.8}$
	Category Average	0.437	-1.5	0.509	-2.3	0.403	$\mathbf{-0.3}$
Noise & Smoothing	Additive Noise (20 dB)	0.440	$\mathbf{-1.2}$	0.520	-1.2	0.390	-1.6
	Additive Noise (10 dB)	0.416	-3.7	0.496	-3.6	0.371	$\mathbf{-3.4}$
	Additive Noise (0 dB)	0.407	$\mathbf{-4.6}$	0.481	-5.1	0.359	-4.7
	Smoothing ($w = 5$)	0.434	-1.8	0.494	-3.9	0.389	$\mathbf{-1.6}$
	Category Average	0.424	$\mathbf{-2.8}$	0.498	-3.4	0.377	-2.8
Temporal Symmetry	Temporal Reversal†	0.426	-2.6	0.500	-3.2	0.398	$\mathbf{-0.8}$
	Sign Inversion†	0.427	-2.5	0.510	-2.2	0.404	$\mathbf{-0.1}$
	Category Average	0.427	-2.6	0.505	-2.7	0.401	$\mathbf{-0.4}$
Sampling Horizon	Resample (256 pts)	0.442	-1.1	0.524	$\mathbf{-0.8}$	0.423	$+1.7$
	Resample (2048 pts)	0.447	-0.6	0.534	$\mathbf{+0.1}$	0.401	-0.4
	Category Average	0.444	-0.8	0.529	$\mathbf{-0.3}$	0.412	$+0.7$

Table 10: Controlled TSExam comparison of alternative secondary representations. "Only" evaluates the secondary representation without the line-plot stream, whereas "+ Plot" combines it with the line plot.

Secondary Representation	Only	+ Plot
STFT (z-normalized)	0.739	0.889
STFT (raw)	0.745	0.894
Delay	0.737	0.893

Full-training comparison. We next train the STFT-based alternative under the full training setup and compare it with the proposed delay-based model. As shown in Table 12, neither representation uniformly dominates across all evaluations. STFT is slightly stronger on ChatTS numerical alignment and the ChatTS reasoning average, whereas the delay-based model is slightly stronger on TSExam and substantially stronger on the broader TSRBench evaluation. On ChatTS categorical alignment, the two models are effectively tied. These results suggest that the benefit of complementary views is not tied to uniform superiority of the delay representation on every task; rather, the choice of secondary representation changes the inductive bias available to the language model.

C.6 Visual-Token Budget and Same-View Controls

For each input series, the line-plot representation is encoded by the native Qwen3-VL vision encoder into 230 visual tokens, while the delay representation contributes 64 visual tokens, for a total of 294 visual tokens per series in the full model. The *Plot+Plot* and *Delay+Delay* controls preserve this budget and the complete dual-tower architecture, changing only the input to one tower. In *Plot+Plot*, the DINOv3 tower receives the line plot, rasterized at 256×256 px, in place of the delay map, yielding 64 tokens. In *Delay+Delay*, the Qwen3-VL tower receives the delay map rendered at approximately 480×480 px, yielding approximately 230 tokens, with the embedding

Table 11: Evaluation-time affine and dynamical interventions on TSExam ($N = 746$) for the single-representation models. For each model, we report clean accuracy and the percentage-point change (Δ) relative to clean performance. Daggers (†) mark non-invariance interventions for which gold targets legitimately change. Bold values indicate the smallest absolute shift ($|\Delta|$) in each row.

		STFT (z-normalized)		**STFT (raw)**		**Delay**	
Intervention Type	Transformation	Acc.	Δ vs. Clean	Acc.	Δ vs. Clean	Acc.	Δ vs. Clean
Baseline	Clean	0.739	—	0.745	—	0.737	—
Affine Invariance	Scale $\times 0.1$	0.725	-1.3	0.727	-1.9	0.733	$\mathbf{-0.4}$
	Scale $\times 10$	0.728	-1.1	0.727	-1.9	0.731	$\mathbf{-0.7}$
	Offset $+5\sigma$	0.728	-1.1	0.634	-11.1	0.732	$\mathbf{-0.5}$
	Offset -5σ	0.729	-0.9	0.649	-9.7	0.731	$\mathbf{-0.7}$
	z-score	0.724	-1.5	0.698	-4.7	0.732	$\mathbf{-0.5}$
	Category Average	0.727	-1.2	0.687	-5.9	0.732	$\mathbf{-0.6}$
Noise & Smoothing	Additive Noise (20 dB)	0.665	-7.4	0.736	$\mathbf{-0.9}$	0.708	-2.9
	Additive Noise (10 dB)	0.538	-20.1	0.639	$\mathbf{-10.6}$	0.572	-16.5
	Additive Noise (0 dB)	0.421	-31.8	0.539	$\mathbf{-20.6}$	0.468	-26.9
	Smoothing ($w = 5$)	0.682	-5.6	0.653	-9.2	0.728	$\mathbf{-0.9}$
	Category Average	0.577	-16.2	0.642	$\mathbf{-10.3}$	0.619	-11.8
Temporal Symmetry	Temporal Reversal†	0.655	-8.3	0.676	$\mathbf{-7.0}$	0.607	-13.0
	Sign Inversion†	0.693	-4.6	0.744	$\mathbf{-0.1}$	0.631	-10.6
	Category Average	0.674	-6.5	0.710	$\mathbf{-3.6}$	0.619	-11.8
Sampling Horizon	Resample (256 pts)	0.768	$+2.9$	0.732	$\mathbf{-1.3}$	0.776	$+3.9$
	Resample (2048 pts)	0.658	-8.0	0.653	-9.2	0.718	$\mathbf{-1.9}$
	Category Average	0.713	-2.6	0.693	-5.3	0.747	$\mathbf{+1.0}$

Table 12: Full-training comparison between the proposed delay-based dual-view model and an alternative model using STFT as the secondary representation. ChatTS-A and ChatTS-B report categorical (Cat.) and numerical (Num.) alignment accuracy; Reason. reports the ChatTS reasoning average.

		ChatTS-A		ChatTS-B			
Model	TSExam	Cat.	Num.	Cat.	Num.	Reason.	TSRBench
Plot + STFT	0.890	0.859	0.809	0.894	0.871	0.628	0.467
Plot + Delay (Ours)	0.899	0.864	0.805	0.893	0.868	0.615	0.532

window scaled so that its geometry matches the 256-px DINOv3 input. Each control is trained from its own Stage I alignment followed by Stage II instruction tuning, using the same data and hyperparameters as our model. The controls are therefore matched to the full model in architecture, trainable parameters, visual-token budget, and training procedure, and differ only in whether the two towers receive distinct or identical views. They are not matched to the single-view variants, which use one tower and fewer visual tokens. Notably, Delay+Delay allocates substantially more tokens to the delay representation than our model does, and Plot+Plot allocates more tokens to the line plot, yet neither reproduces the gain of the complementary pair, indicating that the improvement cannot be attributed to additional capacity devoted to either view alone.

C.7 Complete Benchmark Results

This section reports the complete per-task results underlying the summaries in the main text. Table 13 gives TSRBench accuracy for every sub-task, grouped by input modality (text, vision, both, and dedicated TSLMs). Table 14 reports the ChatTS alignment benchmark on univariate and multivariate series, and Table 15 reports the ChatTS reasoning evaluation. Table 16 breaks down TSExam accuracy by question category. Unless otherwise noted, baseline results are taken from the original

Table 13: TSRBench accuracy (%) by input modality. Bold columns (Perc., Reas., Pred., Dec.) are item-weighted category averages over their sub-tasks. Gray rows denote reference models (proprietary or open-weight models larger than 72B parameters).

	Perception					Reasoning								Prediction			Decision			
Model	**Perc.**	PR	NU	AD	CA	**Reas.**	ER	CD	AR	TR	NR	DR	IR	**Pred.**	TSF	EP	**Dec.**	QualDM	QuantDM	**Overall**
Textual Time Series as Input																				
DeepSeek-V3.2	63.9	67.7	56.3	57.4	64.6	35.6	32.0	35.7	70.7	19.4	27.3	47.6	24.0	33.4	26.5	47.2	30.8	33.1	28.3	39.1
GPT-OSS-120B	64.6	66.8	56.3	59.8	69.0	38.5	39.1	26.3	58.7	31.3	48.5	28.0	40.0	40.6	31.1	59.7	32.4	33.7	31.0	42.0
Qwen3-235B-A22B	63.8	66.0	56.3	59.7	67.3	40.9	22.6	34.7	86.7	28.1	44.8	49.2	39.0	35.6	29.0	48.9	32.5	34.8	30.0	42.2
GPT-5-mini	68.7	72.2	59.8	63.6	69.9	47.5	45.1	27.7	62.7	39.4	65.2	41.3	50.0	38.3	23.5	67.8	33.0	35.5	30.3	46.6
o4-mini	69.7	73.1	65.5	61.2	71.7	47.7	42.6	39.0	58.0	34.4	64.7	42.0	43.0	46.5	33.1	73.3	33.0	30.4	36.0	47.7
GPT-5	70.3	75.7	62.1	61.2	69.0	60.1	42.3	55.3	85.3	68.8	72.2	50.0	62.0	51.8	37.8	79.7	31.9	31.9	32.0	55.5
Qwen2.5-3B	45.4	46.4	51.7	33.3	51.3	27.3	20.0	25.0	38.0	21.2	34.8	29.2	19.0	40.4	31.4	58.3	23.3	22.7	24.0	33.2
Qwen2.5-7B	49.6	50.7	50.6	41.1	54.9	28.3	15.1	32.3	46.0	28.1	33.2	24.0	28.0	34.8	33.5	37.5	28.5	31.0	25.7	33.7
Qwen3-1.7B	47.0	45.8	59.8	38.0	51.3	27.7	18.3	30.1	48.7	30.0	27.0	28.4	20.0	48.7	39.2	67.8	28.2	31.3	24.7	36.8
InternLM3-8B	47.7	51.2	42.5	35.7	54.0	28.6	23.4	24.7	48.0	23.8	30.0	28.7	31.0	46.5	36.6	66.4	30.4	33.7	26.7	37.0
Gemma3-12B-it	54.3	51.5	59.8	48.9	65.5	30.6	21.1	34.7	42.0	23.7	33.2	28.0	41.0	37.7	39.4	34.2	32.9	34.3	31.3	37.2
GPT-OSS-20B	61.7	64.2	58.6	50.4	69.0	33.6	31.4	32.0	48.7	17.5	43.3	20.7	43.0	30.7	27.4	37.2	33.9	34.9	32.7	38.1
Qwen3-8B	52.6	51.8	56.3	48.8	56.6	29.1	14.0	29.3	57.3	23.1	36.5	23.6	33.0	47.4	46.7	48.9	32.0	34.9	28.7	38.3
Qwen3-32B	55.0	54.7	52.9	50.4	62.8	35.2	22.6	35.9	66.0	27.5	37.5	34.4	36.0	36.1	31.1	46.1	33.4	34.9	31.7	38.6
Gemma3-27B-it	57.0	56.1	59.8	48.1	68.1	33.7	22.3	37.0	66.7	21.9	36.0	29.3	35.0	38.3	36.1	42.8	31.8	33.1	30.3	38.6
Qwen2.5-72B	55.6	55.3	55.2	47.3	66.4	39.7	20.9	58.0	62.0	33.8	38.2	36.8	40.0	43.9	30.7	70.3	32.3	34.0	30.3	42.4
Visual Time Series as Input																				
Qwen3-VL-235B-A22B	65.9	65.8	65.5	61.2	71.7	44.4	39.7	47.7	84.7	25.0	43.0	42.8	31.0	25.7	25.3	26.4	30.7	32.5	28.7	41.0
Llama-4-Scout-17B-16E	42.3	41.5	44.8	38.8	46.9	39.1	25.1	53.0	75.3	28.1	41.5	30.0	23.0	52.5	39.7	78.0	33.5	33.7	33.3	42.3
GPT-5-mini	74.1	78.7	69.0	69.0	69.0	45.2	36.0	41.0	58.7	41.9	54.5	39.6	51.0	40.4	27.5	66.1	27.1	26.0	28.3	46.0
o4-mini	74.2	77.4	72.4	66.7	73.5	40.9	29.7	30.7	55.3	39.4	56.2	32.8	51.0	44.1	29.4	73.6	35.8	29.9	42.3	46.6
GPT-5	78.0	83.6	72.4	69.8	73.5	52.4	34.6	37.0	81.3	68.1	64.0	48.0	57.0	49.8	38.8	71.9	28.4	26.9	30.0	52.4
Qwen2.5-VL-3B	46.0	44.2	50.6	41.9	53.1	27.6	23.4	24.0	42.0	25.6	30.2	27.6	25.0	30.9	24.2	44.2	25.8	31.0	20.0	31.3
Phi4-Multimodal-8B	49.0	52.3	46.0	41.9	48.7	26.0	24.6	22.3	28.7	23.1	30.5	25.2	26.0	30.0	32.1	25.8	32.1	34.9	29.0	31.9
InternVL3.5-1B	45.9	47.2	49.4	38.8	46.9	27.8	25.7	24.3	46.7	24.4	29.2	24.0	27.0	38.6	34.7	46.4	28.2	34.3	21.3	33.8
Qwen2.5-VL-7B	54.7	55.0	52.9	52.7	57.5	28.0	15.1	28.0	54.0	33.1	28.0	26.8	28.0	44.1	38.6	55.0	21.9	36.4	5.7	35.8
MiniCPM-V-4.5-8B	61.3	63.6	56.3	56.6	62.8	26.5	22.3	24.3	54.7	20.6	26.5	22.8	24.0	39.0	44.6	27.8	27.9	26.3	29.7	35.9
MiMo-VL-7B-RL	59.1	58.2	64.4	52.7	65.5	33.1	23.7	34.7	65.3	30.6	33.0	25.6	36.0	33.2	35.0	29.7	30.9	29.0	33.0	37.2
Qwen2.5-VL-72B	63.9	63.1	55.2	59.7	77.9	37.1	27.4	52.3	58.0	33.1	39.5	24.0	23.0	35.3	16.7	72.5	29.0	29.3	28.7	39.9
InternVL3.5-38B	60.3	58.8	60.9	55.8	69.9	38.1	30.9	43.0	52.7	31.2	43.8	30.4	34.0	32.3	32.2	32.5	32.3	35.2	29.0	39.4
InternVL3.5-8B	59.3	60.9	55.2	52.7	64.6	35.6	25.7	38.3	60.0	27.5	40.5	31.6	29.0	39.4	38.2	41.9	28.2	33.4	22.3	39.5
Qwen3-VL-8B	58.3	54.7	57.5	62.8	65.5	29.5	24.3	31.7	52.0	23.1	33.2	19.2	29.0	44.6	51.7	30.6	32.8	36.1	29.0	38.9
Qwen3-VL-32B	67.3	67.4	63.2	63.6	74.3	40.4	27.7	62.3	60.0	29.4	44.8	26.4	24.0	46.2	42.6	53.3	30.9	32.8	28.7	45.0
Both Visual & Textual Time Series as Input																				
Claude-4.5-Haiku	53.4	57.1	41.4	48.8	55.8	39.6	24.0	19.7	78.0	32.5	50.0	45.2	52.0	30.9	22.2	48.3	22.8	18.1	28.0	37.1
Gemini-2.5-Flash	74.6	75.2	73.6	69.0	79.6	40.9	36.0	31.0	86.7	35.6	39.8	36.8	43.0	40.0	26.7	66.7	41.6	34.3	49.7	46.5
GPT-5-mini	73.1	78.4	65.5	63.6	72.6	47.1	38.9	36.3	76.0	33.1	66.2	30.0	54.0	38.1	24.4	65.6	32.1	34.3	29.7	46.9
o4-mini	74.9	79.0	70.1	65.1	76.1	45.4	37.4	38.0	64.0	35.0	66.8	24.8	51.0	44.3	29.3	74.4	32.8	29.9	36.0	48.2
o4-mini-high	76.7	82.5	71.3	64.3	76.1	56.7	44.0	66.3	76.7	55.0	70.8	30.8	53.0	46.4	33.6	71.9	24.9	25.4	24.3	52.5
GPT-5-mini-high	75.9	80.3	75.9	63.6	75.2	60.5	47.4	69.0	82.0	56.2	71.8	42.0	57.0	44.6	32.5	68.9	28.8	32.8	24.3	54.1
GPT-5	76.7	84.9	72.4	61.2	70.8	58.1	38.9	50.3	93.3	68.8	72.0	44.0	58.0	51.2	37.8	78.1	33.0	31.9	34.3	55.6
Time Series Language Models (TSLM)																				
OpenTSLM-tsqa-flamingo-3B	39.5	41.2	41.4	34.1	38.9	26.3	22.9	30.3	36.0	21.2	22.5	28.8	29.0	33.9	28.1	45.6	27.4	25.4	29.7	30.7
OpenTSLM-tsqa-sp-3B	39.7	39.9	40.2	35.7	43.4	26.8	20.3	32.3	33.3	28.1	24.5	26.0	32.0	33.5	32.5	35.6	28.5	28.4	28.7	31.0
ChatTS-14B	50.7	50.7	50.6	46.5	55.8	29.3	21.7	34.3	51.3	24.4	30.5	23.6	25.0	30.2	19.2	52.2	28.5	29.3	27.7	33.5
TS-Reasoner-7B	52.1	53.1	56.3	48.1	50.4	31.3	28.9	24.3	57.3	26.9	37.2	23.6	24.0	39.5	31.7	55.0	27.6	32.8	21.7	36.4
TimeOmni-1-7B	54.7	55.0	59.8	41.9	64.6	31.0	28.0	35.3	46.7	24.4	31.5	22.8	34.0	36.5	30.0	49.4	32.3	34.0	30.3	36.7
Ours	86.4	85.4	94.3	78.3	92.9	40.1	43.1	51.7	56.0	28.7	35.5	29.2	34.0	61.7	50.4	84.2	37.6	32.8	43.0	53.2

benchmark papers. We evaluated GPT-5 (`gpt-5-2025-08-07`) on TSExam and ChatTS ourselves with text (T), vision (V), and combined (T+V) inputs, using the default sampling parameters and the default (medium) reasoning effort. For TSExam, we use the same minimal prompt as in all of our evaluations: the question and its options, with an instruction to reply with the option letter alone, and without the optional hints, concept definitions, example series, worked example, or step-by-step reasoning instruction. Responses are scored by the selected option letter. On TSRBench, we re-evaluated the Qwen vision-language models (Qwen2.5-VL-7B, Qwen2.5-VL-72B, Qwen3-VL-8B, and Qwen3-VL-32B) ourselves, rendering each time series with the same line-plot pipeline used to evaluate our model, so that the comparison with our Qwen3-VL-based model is not confounded by differences in plot rendering. All other TSRBench baselines, including GPT-5, are taken from the benchmark.

Table 14: ChatTS alignment benchmark on univariate (Dataset A) and multivariate (Dataset B) series. Gray rows denote reference models. Corr. and Clus. are the correlation and clustering subtasks (categorical only). Bold values mark the best result per column within each dataset, excluding reference models.

Dataset	Model	Trend		Season		Noise		Local		Corr.	Clus.	Overall	
		Cate.	Num.	Cate.	Num.	Cate.	Num.	Cate.	Num.	Cate.	Cate.	Cate.	Num.
A	GPT-4o (T)	0.585	0.882	0.811	0.768	0.905	0.153	0.379	0.256	0.476	0.333	0.542	0.371
	GPT-4o (V)	0.659	0.613	0.811	0.559	0.810	0.248	0.537	0.414	0.476	0.480	0.609	0.436
	GPT-5 (T)	0.512	0.887	0.973	0.000	0.905	0.350	0.905	0.741	0.881	0.485	0.797	0.724
	GPT-5 (V)	0.732	0.646	0.973	0.000	0.762	0.394	0.874	0.725	0.857	0.600	0.810	0.670
	GPT-5 (T+V)	0.585	0.905	0.973	0.000	0.905	0.423	0.863	0.735	0.857	0.558	0.801	0.731
	Qwen3-VL-8B (V)	0.098	0.011	0.108	0.000	0.167	0.051	0.095	0.052	0.619	0.454	0.231	0.043
	Qwen3-VL-32B (V)	0.854	0.603	0.946	0.000	0.833	0.298	0.663	0.507	0.643	0.573	0.733	0.500
	Qwen2.5-14B (T)	0.707	0.709	0.622	0.205	0.833	0.231	0.137	0.099	0.571	0.349	0.464	0.241
	TS-Reasoner-7B	0.902	0.706	0.919	0.428	0.929	0.383	0.700	0.586	0.663	0.371	0.743	0.631
	TimeOmni-1-7B	0.658	0.535	0.297	0.137	0.738	0.374	0.117	0.069	0.244	0.148	0.316	0.191
	ChatTS-14B	**0.927**	**0.874**	**0.973**	**0.849**	0.857	0.511	0.895	0.805	**0.905**	**0.782**	**0.889**	0.788
	Ours	0.829	0.864	**0.973**	0.515	**0.976**	**0.622**	**0.900**	**0.813**	0.737	0.778	0.864	**0.805**
B	GPT-4o (T)	0.690	0.825	0.732	0.474	0.573	0.331	0.191	0.136	0.324	0.281	0.366	0.284
	GPT-4o (V)	0.702	0.361	0.938	0.589	0.610	0.398	0.375	0.265	0.367	0.389	0.472	0.311
	GPT-5 (T)	0.655	0.914	0.948	0.790	0.768	0.476	0.648	0.519	0.527	0.445	0.602	0.606
	GPT-5 (V)	0.738	0.620	0.959	0.701	0.793	0.576	0.648	0.507	0.633	0.530	0.660	0.544
	GPT-5 (T+V)	0.750	0.922	0.990	0.841	0.829	0.554	0.648	0.537	0.629	0.511	0.661	0.627
	Qwen3-VL-8B (V)	0.167	0.022	0.155	0.093	0.146	0.015	0.074	0.041	0.400	0.373	0.255	0.042
	Qwen3-VL-32B (V)	0.786	0.382	0.918	0.549	0.744	0.221	0.375	0.281	0.516	0.417	0.532	0.318
	Qwen2.5-14B (T)	0.711	0.669	0.705	0.217	0.256	0.094	0.111	0.082	0.402	0.276	0.339	0.193
	TS-Reasoner-7B	0.905	0.600	0.990	0.778	0.976	0.403	0.481	0.341	0.665	0.586	0.675	0.425
	TimeOmni-1-7B	0.524	0.231	0.515	0.366	0.329	0.166	0.068	0.028	0.247	0.011	0.195	0.094
	ChatTS-14B	**0.976**	0.902	**1.000**	**0.930**	0.927	0.572	0.828	0.752	0.818	0.834	0.862	0.787
	Ours	0.964	**0.906**	**1.000**	0.920	**0.988**	**0.705**	**0.917**	**0.860**	**0.829**	**0.846**	**0.893**	**0.868**

Table 15: ChatTS reasoning evaluation (Dataset A). Inductive reasoning is free-form Q&A scored with the RAGAS pipeline; the other tasks are multiple-choice or true/false accuracy. MCQ2 samples 100 questions from the two-series set. Gray rows denote reference models. Bold values mark the best result per column, excluding reference models.

Model	Induct.	Deduct.	Causal	MCQ2	Average
GPT-4o (T)	0.336	0.628	0.685	0.510	0.540
GPT-4o (V)	0.322	0.605	0.652	0.490	0.517
GPT-5 (T)	0.277	0.581	0.663	0.730	0.563
GPT-5 (V)	0.382	0.674	0.707	0.675	0.610
GPT-5 (T+V)	0.315	0.605	0.717	0.760	0.599
Qwen3-VL-8B (V)	0.197	0.558	0.685	0.480	0.480
Qwen3-VL-32B (V)	0.399	0.651	0.750	0.610	0.603
Qwen2.5-14B (T)	0.184	0.605	0.348	0.320	0.364
ChatTS-14B	**0.518**	**0.744**	0.804	0.600	**0.667**
TS-Reasoner-7B	0.282	0.632	0.717	0.610	0.560
TimeOmni-1-7B	0.040	0.419	0.400	0.580	0.360
Ours	0.479	0.535	**0.826**	**0.620**	0.615

Table 16: TSExam accuracy across the 746 questions, by category. Gray rows denote reference models. Bold values mark the best result per column, excluding reference models.

Model	Overall	Anomaly	Causality	Noise	Pattern	Similarity
GPT-4o (T)	0.586	0.491	0.458	0.667	0.594	0.667
GPT-4o (V)	0.736	0.583	0.611	0.786	0.787	0.758
GPT-5 (T)	0.724	0.704	0.556	0.679	0.782	0.700
GPT-5 (V)	0.784	0.648	0.431	0.750	0.906	0.775
GPT-5 (T+V)	0.808	0.731	0.417	0.738	0.931	0.792
Qwen3-VL-8B (V)	0.618	0.546	0.514	0.643	0.630	0.692
Qwen3-VL-32B (V)	0.713	0.639	0.431	0.690	0.793	0.725
ChatTS-14B	0.570	0.481	0.500	0.619	0.575	0.642
TS-Reasoner-7B	0.441	0.343	0.458	0.512	0.442	0.467
TimeOmni-1-7B	0.483	0.389	0.375	0.583	0.497	0.517
Ours	**0.899**	**0.759**	**0.764**	**0.988**	**0.948**	**0.900**

D Training Data Details

This section provides the composition and sampling procedure of the data used in the two training stages. Stage I aligns the complementary visual representations with the language-model representation space using descriptive time-series supervision. Stage II performs instruction tuning over a mixture of time-series understanding, numerical, and reasoning tasks. We distinguish throughout between the number of examples available in each source (*raw examples*) and the number sampled during one training epoch (*effective draws*), since several sources are explicitly rebalanced.

D.1 Training Corpus Overview

Table 17 summarizes the datasets used in both training stages. Stage I contains 55,696 raw examples drawn from three captioning sources. Because these sources differ substantially in size, we sample them with equal probability at the source level, such that each contributes one third of the effective Stage-I distribution. We draw 55,696 examples per epoch and train for three epochs.

Stage II combines instruction-following data, multiple-choice and numerical supervision, and a 100,000-example reasoning corpus. The effective training epoch contains 182,815 draws. The reasoning corpus and numerical corpus are retained at their natural sizes, contributing 100,000 and 11,000 examples, respectively. The remaining 71,815 draws are distributed approximately equally among ChatTS IFT, ChatTS SFT, and TSExam perception data. Consequently, the effective training distribution differs from the raw corpus distribution, preventing the largest individual source from dominating instruction tuning.

D.2 Stage I: Modality-Alignment Data

Stage I is designed to align the visual time-series representations with language while preserving the pretrained capabilities of the underlying vision-language model. Its supervision therefore consists of descriptive time-series examples rather than the downstream reasoning tasks used in Stage II.

The corpus combines 15,995 CaTS examples, 32,000 ChatTS-style caption examples, and 7,701 structural caption examples, for 55,696 raw examples in total. The substantial difference in source sizes motivates source-level balancing. At each draw, one of the three sources is selected with equal probability, after which an example is sampled from that source. Thus, each source accounts for approximately one third of the effective Stage-I training distribution, irrespective of its raw size. We use 55,696 draws per epoch for three epochs. This stage exposes the model to diverse descriptions of temporal signals and provides supervision for associating the visual representations with their corresponding temporal semantics before instruction tuning.

Table 17: Training-data composition. Raw examples denote the number of available examples before sampling, whereas effective draws denote the approximate number presented in one training epoch. Stage-I sources are sampled with equal source probability. In Stage II, the reasoning and numerical corpora retain their natural sizes, while the remaining sampling budget is divided approximately equally among the three instruction/multiple-choice sources.

Stage	Source	Raw Examples	Effective Draws / Epoch	Supervision
I	CaTS	15,995	∼18,565	Caption / description
I	ChatTS-style captions	32,000	∼18,565	Caption / description
I	Structural captions	7,701	∼18,565	Caption / description
	Stage I total	*55,696*	*55,696*	
II	ChatTS IFT	6,373	∼23,938	Instruction following
II	ChatTS SFT	44,802	∼23,938	Instruction following
II	TSExam perception	20,640	∼23,939	Multiple choice
II	Numerical corpus	11,000	11,000	Numerical
II	Reasoning corpus	100,000	100,000	Reasoning / multiple choice
	Stage II total	*182,815*	*182,815*	

D.3 Stage II: Instruction-Tuning Data

Stage II broadens the supervision from descriptive alignment to instruction-conditioned time-series understanding and reasoning. The training mixture consists of ChatTS instruction-following data, TSExam perception questions, the numerical corpus, and a separately constructed reasoning corpus.

The raw instruction-tuning sources contain 6,373 ChatTS IFT examples, 44,802 ChatTS SFT examples, 20,640 TSExam perception examples, 11,000 numerical corpus examples, and 100,000 reasoning examples. Rather than sampling this mixture according to raw source frequency, we explicitly control its effective composition. The 100,000 reasoning examples and 11,000 numerical examples are retained in each epoch. The remaining 71,815 draws are divided approximately equally among ChatTS IFT, ChatTS SFT, and TSExam perception data, yielding approximately 23.9k draws from each source and 182,815 total draws per epoch.

The reasoning corpus is designed to cover a broad range of temporal reasoning capabilities and interaction formats, including numerical reasoning, event prediction, entity recognition, causal discovery, sequence continuation, dynamical reasoning, temporal reasoning, abductive reasoning, and inductive reasoning. While these capabilities overlap with those evaluated by TSRBench, the underlying time series, questions, and supervision in the reasoning corpus are constructed independently.

Evaluation-set separation. No evaluation examples from TSRBench, TSExam-HF, or the ChatTS evaluation sets are included in the training mixture. This separation applies to both training stages.

D.4 Reasoning Corpus Specification

The Stage-II reasoning corpus contains 100,000 examples spanning nine task families and 25 generation templates. Table 18 summarizes the size of each family, the underlying data-generating process, the number of input channels and answer choices, and the supervision format. The task families are denoted NR (numerical reasoning), EP (event prediction), ER (entity recognition), CD (causal discovery), TSF (time-series forecasting), DR (dynamical reasoning), TR (temporal reasoning), AR (abductive reasoning), and IR (inductive reasoning).

The corpus deliberately combines tasks with different forms of supervision. NR, EP, and IR require explicit numerical computation or intermediate multi-step inference and together account for 36,328 examples. For these families, we retain rationale supervision followed by the final answer. The remaining six families—CD, ER, TSF, AR, DR, and TR—account for 63,672 examples and are trained using the final option letter directly. Although deterministically generated rationales are available for 97,844 of the 100,000 examples, we discard these rationales for the latter families in the final training configuration rather than applying chain-of-thought supervision uniformly across all tasks.

Table 18: Taxonomy of the reasoning corpus (100,000 examples across 25 templates). Shares are computed over the complete reasoning corpus. Where a family mixes sources, *Channels* and *Options* are listed per source in the order of the description.

Family	Tmpl.	Items	Share	Underlying Generation / Governing Process	Channels	Options	Output
NR	8	16,874	16.9%	Eight physical formulas: traffic flow, cooling load, growing degree days, seismic S–P distance, reaction kinetics, Penman–Monteith evapotranspiration, intersection delay, and wind power.	2–4	4	Rationale + Letter
EP	1	15,225	15.2%	Five-channel synthetic weather traces in which rainfall events are triggered by a latent humidity surge and barometric-pressure drop; two informative and three nuisance channels.	5	2	Rationale + Letter
ER	2	14,802	14.8%	Parameterized light curves spanning five variable-star classes and tri-axial accelerometer segments spanning six physical-activity classes.	1 / 3	5 / 6	Option Letter
CD	1	12,687	12.7%	Random five-node directed acyclic graphs simulating sensor networks across heterogeneous ranges; the task is to identify the generating adjacency structure.	5	4	Option Letter
TSF	1	12,500	12.5%	Histories containing trend, seasonality, and autoregressive structure; the task is to select the appropriate continuation trajectory among generated alternatives.	4	4	Option Letter
DR	3	10,573	10.6%	Propagation of dynamical processes including AR(1), geometric growth, and continuous Lorenz-attractor state trajectories.	4	4	Option Letter
TR	2	6,766	6.8%	Synthetic conflict–cooperation proxy series paired with timestamped event summaries; tasks evaluate event ordering and before/after precedence.	1	4	Option Letter
AR	5	6,344	6.3%	Real NBA and NFL play-by-play games with win-probability trajectories and a masked interval of ten plays; the task is to select the first hidden play among four candidates (2,156 items). Synthetic trend-plus-season series with one injected anomaly (spike, cutoff, scale, speed-up, sign-flip, or wander); the task is to name the anomaly class or the index band containing it (4,188 items).	2 / 1	4 / 5	Option Letter
IR	2	4,229	4.2%	Long-range temporal patterns based on monthly sunspot sequences and multichannel climate indicators used to infer higher-level temporal outcomes.	1 / 8	4	Rationale + Letter

Data provenance. The reasoning corpus combines procedurally generated time series with tasks constructed from real underlying temporal data. Most families use synthetic generative processes that provide direct control over the latent mechanism and the correct answer. The only real temporal data are NBA (2021–22) and NFL (2021–22 to 2023–24) play-by-play records from GAMETime (Tan et al., 2025), used in two AR templates (2,156 items, 2.2% of the corpus); for these, the questions and supervision are constructed independently of the evaluation sets. This design allows the corpus to combine controlled reasoning problems with temporal patterns occurring in real data without using examples from the evaluation sets described above.

Table 19: Taxonomy of the numerical corpus (11,000 examples across 20 templates). Shares are computed over the complete numerical corpus.

Subcategory	Tmpl.	Items	Share	Target Statistic	Channels	Options	Output
Statistical retrieval	10	5,500	50.0%	Maximum, minimum, mean, median, first and third quartile, standard deviation, variance, first value, and last value.	1	—	Scalar
Signal properties	5	2,750	25.0%	Range, interquartile range, mean absolute value, lag-1 autocorrelation, and number of zero crossings.	1	—	Scalar
Step dynamics	5	2,750	25.0%	Mean absolute first difference, largest one-step jump, best-fit slope, first-half minus second-half mean, and energy.	1	—	Scalar

D.5 Numerical Corpus Specification

The Stage-II numerical corpus contains 11,000 examples spanning three subcategories and 20 generation templates. Table 19 summarizes the size of each subcategory and the statistics it targets. Every item is a single series of length 256 drawn from a shared pool of eleven generating recipes (trends, periodic waves, noise, and autoregressive and moving-average processes) with additive observation noise. The target is the requested statistic computed exactly on the observed series and rounded to two decimals; there are no answer options.