Chat-TS: Enhancing Multi-Modal Reasoning Over Time-Series and Natural Language Data

Paul Quinlan[1,3] **and Qingguo Li**[2,3] **and Xiaodan Zhu**[1,3]
[1]Electrical and Computer Engineering, Queen's University
[2]Mechanical and Materials Engineering, Queen's University
[3]Ingenuity Labs Research Institute, Queen's University
15pwq@queensu.ca, ql3@queensu.ca, xiaodan.zhu@queensu.ca

Abstract

Large language models are being rapidly deployed across many fields such as healthcare, finance, transportation, and energy, where time-series data are fundamental components. The current works are still limited in their ability to perform reasoning that involves both time-series and the corresponding textual content. We address this gap by introducing *Chat-TS*, a large language model (LLM) based framework designed to support reasoning over time series and textual data. Unlike traditional models, Chat-TS integrates time-series tokens into LLMs' vocabulary, enhancing its reasoning ability over both modalities without compromising core natural language capabilities. To support learning and evaluation, we contribute new datasets: the *TS Instruct Training Dataset* (pairing diverse time-series data with relevant text instructions and responses for instruction tuning), the *TS Instruct Question and Answer (QA) Gold Dataset* (multiple-choice questions to evaluate multimodal reasoning), and a *TS Instruct Quantitative Probing Set* (a small subset of TS Instruct QA reasoning tasks alongside math and decision-making questions for LLM evaluation). We design a training strategy to preserve the inherent reasoning capabilities of LLMs while augmenting them for time-series reasoning. Experiments show that Chat-TS achieves state-of-the-art performance in multi-modal reasoning tasks by maintaining strong natural language proficiency while improving time-series reasoning.[1]

1 Introduction

Large language models (LLMs) are rapidly being applied in many fields which necessitate the analysis of time-series such as healthcare (Morid et al., 2023), business (Zhang et al., 2024a), transportation (Zhang et al., 2024b), and many other industries (Kashpruk et al., 2023). Traditional time-series analysis tasks, such as forecasting, imputation, anomaly detection, and classification, have been extensively studied in pure time-series setups (Makridakis et al., 1977; Dempster et al., 2020; Jiang et al., 2024). The most recent advancements in large language models(LLMs) (OpenAI et al., 2024; Team et al., 2024a; Dubey et al., 2024; Marah Abdin and et al., 2024) have opened significant opportunities for performing time-series analysis in the form of textual reasoning in a multimodal setup. The practical application of time-series data analysis often necessitates drawing conclusions and making decisions based on real-world knowledge, which is conveyed and stored in human language. For example, an investment banker may consider both stock prices and relevant company reports, while a doctor may analyze a patient's ECG readings in combination with their medical history and clinical notes.

Recent research has started to apply large language models to time-series analysis, e.g., as financial trading agents (Wang et al., 2023, 2024a; Li et al., 2023) and medical analysts (Nazi and Peng, 2024; Thirunavukarasu et al., 2023). The models are applied within their specific domains and time-series are first converted from numerical representations to a textual format in order to leverage LLMs. This approach has also been used in pure time-series analysis (e.g., forecasting) by (Gruver et al., 2024). These models serve as a state-of-the-art baseline for comparing language model reasoning over time-series. A comprehensive discussion on previous work will be detailed in Section 2. Despite the promising applications of LLMs to handle real-world multi-domain challenges, the training and evaluation of LLMs' reasoning abilities for time-series data remains significantly under-explored.

Developing LLMs with time-series (TS) reasoning capabilities presents several key challenges: (1) Training data scarcity: There is a significant lack

[1]To ensure replicability and facilitate future research, all models, datasets, and code will be available at https://github.com/quinlanp/Chat-TS-Multi-Modal-Reasoning.

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics Volume 1: Long Papers, pages 5621–5647

of training data for constructing models that combine time-series data with reasoning. Multimodal datasets that integrate both text and time-series data are currently limited. (2) Benchmarking: Similarly, there are no large-scale benchmarks available to evaluate multimodal reasoning and decision-making for time-series data. (3) Preserving LLM Capabilities: Current time-series approaches that rely on LLMs for time-series reasoning either do not preserve natural language knowledge and reasoning capabilities inherent in LLMs or do not evaluate it. We advocate that this should be standard in future multi-modal reasoning works. To address these challenges, our paper makes the following contributions:

- **We contribute new datasets** for training and evaluating LLMs for time-series reasoning tasks. The **TS Instruct Training Dataset**, a diverse multimodal dataset that integrates time-series data with relevant textual information, fills a critical gap in the current literature by addressing the scarcity of training datasets. The **TS Instruct QA Gold Benchmark**, a dataset of 1,056 multiple-choice questions with ground truth answers, is designed to evaluate multimodal reasoning capabilities in time-series analysis. We also provide a small dataset for quantitative evaluation, composed of QA, mathematical reasoning, and decision-making samples.
- **We propose an effective approach** to leveraging multimodal LLMs for time-series analysis with text, emphasizing the maintenance of LLMs' reasoning capabilities and using their knowledge reservoir to enhance time-series data reasoning. We expand the embedding space while preserving the LLM's capabilities, enabling strong reasoning after fine-tuning with multimodal data, outperforming existing methods that compromise natural language and reasoning ability.
- **Experiments demonstrate** that the proposed Chat-TS Models improve time-series reasoning capabilities by an average of $\sim$13% on the TS Instruct QA Gold Benchmark, compared to the existing state-of-the-art baseline.

2 Related Work

LLMs for Time-Series Analysis. Many recent works treat large language models as a backbone for forecasting or classification through repurposing their pretrained parameters, but at the cost of disabling their full text-generation and reasoning abilities. For instance, GPT4TS (Zhou et al., 2023) and LLM4TS (Chang et al., 2024) embed series in model weights; TEMPO (Cao et al., 2024), and TEST (Sun et al., 2024) insert series into prompts; and InstructTime (Cheng et al., 2024) fine-tunes for classification, without preserving general multimodal reasoning. LLMTime (Gruver et al., 2024) instead encodes time-series as text, enabling zero-shot forecasting without task-specific fine-tuning and retaining the model's text-based structure.

Time-Series Reasoning with LLMs. A few recent efforts have begun to explore time-series reasoning as a multi-modal natural language task as opposed to the traditional time-series analysis tasks mentioned above. ChatTime (Wang et al., 2025) and TimeMQA (Kong et al., 2025) build generated datasets which test time-series alignment (eg. identifying trends) with LLMs but train and test on subsets from the same global datasets which limit comparability. Chow et al. 2024 augment an LLM with a learned encoder for raw series inputs. However, neither their code nor the assessment of the model's natural-language capabilities has been released. In contrast, Chat-TS not only preserves end-to-end text input and generation, but shows strong generalization on our TS-Instruct QA Gold dataset. This dataset focuses on real-world reasoning scenarios, despite only training on simple conversational samples. We publish both model and dataset benchmarks. A concurrent work (Xie et al., 2025) trains LLMs for time-series reasoning. This work uses a trained encoder and embedding concatenation to combine time-series representations. We believe our methods are orthogonal to (Xie et al., 2025). For details, please refer to Appendix G.

Time-Series Training and Evaluation. Standard repositories (Monash (Godahewa et al., 2021), UCR (Middlehurst et al., 2024), LOTSA (Woo et al., 2024)) offer forecasting and classification benchmarks but lack textual context for multimodal reasoning. Merrill et al. 2024 evaluate LLM reasoning on synthetic series and LLM-generated prompts, highlighting performance gaps but without providing training improvement strategies. We address this gap by releasing (1) a large-scale conversational training corpus of human-LLM exchanges grounded in real-world series metadata, and (2) TS-Instruct QA Gold, a human-validated question–answer set over real series with visual context. Merrill also contributed a single-time-series question and answer (QA) based dataset using synthetic dataset. Unfortunately they deemed it not appropriate for LLM evaluation as existing text models had near perfect performance on these benchmarks. In contrast we contribute a challenging QA benchmark which consists of real-world time-series and utilize human evaluators to verify the accuracy of each question.

Discrete Tokenization. Existing multimodal tokenizers (e.g., VQ-VAE (van den Oord et al., 2018) or channel-wise patching) optimize for compression on very long sequences, often degrading reconstruction fidelity. We introduce a lightweight discrete tokenizer aimed at maximizing accuracy on sub-1,000-point series. While Chronos (Ansari et al., 2024) uses similar discrete tokenization in LLM-adjacent models, it does not maintain full LLM text-generation or reasoning capabilities and focuses on time-series forecasting.

3 A Framework for Reasoning Over Time-Series and Language Data

3.1 Problem Definition

This paper is concerned with multimodal time-series reasoning involving a model's ability to make decisions and answer relevant questions based on time-series, alongside the corresponding text data. An key objective of the design is to expand the vocabulary of any pre-trained LLM to include time-series tokens, training autoregressively along with text. This strategy aims to augment the quality of time-series understanding and reasoning while maintaining the original natural language understanding (NLU) capabilities of a large language model.

Formally, let $\mathcal{V}_L$ denote the original textual vocabulary and $\mathcal{V}_T$ the set of time-series tokens, forming an extended vocabulary $\mathcal{V} = \mathcal{V}_L \cup \mathcal{V}_T$. The LLM, parameterized by θ, is trained to predict the next token in a sequence $\mathbf{y} = \{y_1, y_2, \ldots, y_T\}$, where each $y_t \in \mathcal{V}$, using the autoregressive model $f_\theta : \mathcal{V} \to \mathcal{V}$. Following the convention, the training process maximizes the likelihood of the next token via cross-entropy (CE) loss: $\mathcal{L}(\theta) = -\sum_{t=1}^{T-1} \log p_\theta(y_{t+1}|y_1, y_2, \ldots, y_t)$, where p_θ is the predicted probability of the next token. Since $\mathcal{V}$ contains joint representations for text and time-series, the training process is reduced to the standard auto-regressive LLM training.

3.2 Main Components

The key focus of developing our joint text and time-series models lies in the creation of training data, the development of tokenization methods for time series, and their integration into LLMs for multimodal time-series analysis. A high-level overview is shown in Figure 1.

Data is the cornerstone of any multimodal learning approach, and high-quality text and time-series data are critical for training robust models. However, while there are many high-quality datasets available for either text or time-series, there are no existing datasets that integrate both modalities—a significant barrier to instruction-tuning LLMs for time-series tasks. To address this, we contribute novel datasets designed and curated specifically for this purpose, using a combination of time-series properties, task descriptors, and synthetic conversations, as will be detailed later in this section.

Our framework enables learning over both text and time-series data, in order to allow the model to reason over time-series data while preserving its original language modeling capabilities. Unlike other multimodal models that require connectors to modify representations for LLM integration (McKinzie et al., 2024), we advocate adding time-series tokens to the LLM's vocabulary, eliminating the need for intermediary connecting components between the time-series and text representation spaces and therefore ensuring a more seamless integration. Concretely, we explore two initialization methods: (i) mean initialization, where embeddings are initialized with the average of existing text tokens, and (ii) time-series pre-training, where we unfreeze and tune only the embedding and final linear layer.

We then fine-tuned the models using a dual-dataset strategy that combines instruction-answer pairs from our multimodal TS-Instruct dataset together with a curated subset of a text-based instruction tuning dataset. This approach preserves the LLMs' strong language modeling capabilities while enhancing their power to follow instructions and reason effectively over time-series data. Below, we discuss the details of the components.

3.3 Time-Series-with-Text Multimodal Dataset Curation

Within this work we generate datasets for training and testing using real non-overlapping time-series. To prevent contamination between datasets we take several important measures. During training we

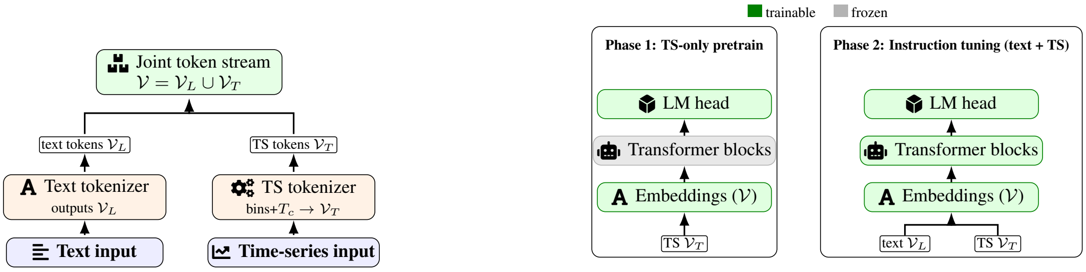


Figure 1: **Left:** Text and time-series inputs are tokenized by their respective tokenizers and merged into a joint token stream over the extended vocabulary $\mathcal{V}$. **Right:** Chat-TS training. *Phase 1* pretrains on TS tokens with frozen transformer blocks; *Phase 2* instruction-tunes on both text and TS tokens.

only generate simple single-turn casual conversations between a user and assistant. We provide some examples in Appendix C. Conversely, for the evaluation datasets, each sample is generated in the multi-choice question-answer format with much stricter conditions.

Importantly, there are a few recent works which build time-series reasoning datasets in some capacity. For example ChatTime(Wang et al., 2025) and Time-MQA(Kong et al., 2025). These datasets in terms of reasoning primarily focus on time-series alignment (eg. Identify the trend), where as ours is concerned with real-world reasoning questions, but they first generate their dataset and then split it into testing and training. In contrast, both our training and evaluation datasets are generated separately in different formats to more broadly evaluate the models generalization capabilities through this controlled separation.

To highlight that our model performance not only improves on our benchmark we also compare to the two time-series question answering dataset (MCQ2TS) as presented (Merrill et al., 2024). This is a time-series reasoning dataset that is based off of synthetically generated time-series and requires counter-factual reasoning over two time-series and is remarkably distinct from all datasets generation within this work.

We curated new datasets to help address the critical scarcity of multimodal training and benchmarking test data for time-series reasoning. As detailed in Appendix A, all datasets will be made publicly available.

The TS-Instruct Dataset. As there is limited multi-modality time-series data available, we synthesize conversations between a user and an assistant using GPT4o-mini (OpenAI et al., 2024). These dialogues incorporate time-series data represented as images alongside metadata (length, channels) to guide the model and generate accurate conversations with real-world time-series for model training. Conversations are designed to focus on general reasoning, classification, decision-making, and mathematical analysis. Time-series data for reasoning tasks are sourced from LOTSA (Woo et al., 2024), while classification labels come from the Time-Series Classification Archive (Middlehurst et al., 2024). The details of the dataset can be found in Appendix C.

The TS-Instruct QA Gold Benchmark. We designed a multimodal evaluation dataset to address the lack of benchmarks for multimodal models capable of time-series reasoning. The dataset consists of multiple-choice question-and-answer (QA) tasks, where each question includes a description of the time-series data, four answer options, and a detailed explanation of the correct answer. This allows us to assess both answer accuracy and reasoning ability. The dataset was generated using GPT-4o-mini, which, combined with visual representations and time-series meta-data is used to generate the text components of the multiple-choice questions. The full system prompt is detailed in Appendix E.

To ensure the quality of the evaluation data, each sample was manually evaluated by hand for quality and to ensure each samples had a single correct answer. For more details see Appendix I. The Gold dataset serves as a reliable benchmark for time-series reasoning under optimal conditions, providing a fair basis for comparing models. After pruning, the TS Instruct QA Gold benchmark consists of 1,056 samples. The correct answers are balanced so that each option appears 264 times within the dataset.

3.4 The Tokenization Model

In order to preserve natural language reasoning we aim to expand the embedding vocabulary space in combination with a time-series tokenizer as a minimally invasive method for incorporating time-series reasoning. Tokenizing time-series data presents unique challenges, as it involves converting continuous numerical values into discrete tokens that can be processed by an LLM. Our approach balances reconstruction accuracy with token scalability, ensuring that the tokenized time-series can be efficiently represented without compromising data integrity.

There are several key considerations when designing our time-series tokenizer:

1. **Reconstruction accuracy**: The tokenizer should encode and decode time-series data with minimal error.
2. **Compression**: It should allow for efficient compression to process larger time-series sequences.
3. **Scalability**: The tokenizer needs to support varying vocabulary sizes, improving quality as the number of tokens increases.

Our discrete tokenizer functions as follows: Let $\boldsymbol{X} \in \mathbb{R}^{L \times M}$ represent a multivariate time-series, where L is the sequence length and M is the number of channels. We quantize the time-series by dividing the range $[-s, s]$ into $K - 1$ bins, mapping each value x to its corresponding bin index i, which forms a sequence of discrete tokens $\boldsymbol{Z}$. A special token $T_c = K$ is added to mark the end of each channel, and the token sequence $\boldsymbol{T}$ is then flattened and offset to align with the vocabulary size of the text tokenizer. Decoding reverses this process, reconstructing the time-series by mapping each index back to the bin centers. The special token is used as a channel separator to reconstruct the original 2D time-series from the 1D sequence of tokens. We use $s = \pm 3\sigma$ of the normalized time-series and a codebook size K to be 8192 tokens.

We implemented and evaluated four tokenizers, including models using simple linear layers for encoding/decoding with vector quantization (VQ) (van den Oord et al., 2018), lookup-free quantization (LFQ) (Yu et al., 2024), and a TCN-based (van den Oord et al., 2016) tokenizer using residual vector quantization (RVQ) (Chen et al., 2010) as implemented in InstructTime (Cheng et al., 2024). We compare these approaches to our discrete tokenizer that requires no training and provides near-perfect reconstruction of time-series data.

Tokenizer Selection. There are many options in terms of time-series tokenizer architecture (simple discrete, vector quantization based etc.) which would work within our framework. To select the best tokenizer, we measured reconstruction quality using varying compression ratios and codebook sizes on the Azure VM Traces 2017 dataset (Hadary et al., 2020). Figure 2 presents the comparison, showing that our discrete tokenizer outperforms the others in terms of validation error and improves predictably as the codebook size increases.

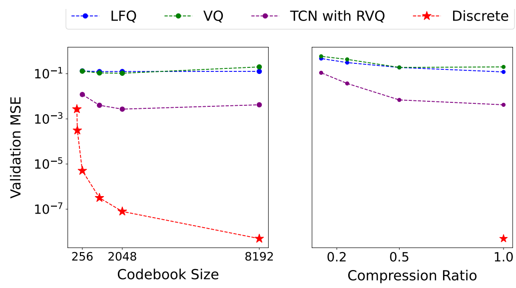


Figure 2: Tokenizer validation: reconstruction error under varying compression ratio and codebook size .

The results demonstrate that the discrete tokenizer significantly outperforms the other tokenizers in terms of validation error and predictably improves when scaling the number of codebook tokens. Importantly, our tokenizer does not require any training and therefore does not suffer from out of distribution problems that may arise otherwise. It is important to note that regardless of choice in tokenizer setup (simple discrete, vq-based etc.) is considerably **compressed** compared to the baseline text encoding strategies used in modern LLMs.

4 Experimental Setup

We use different types of datasets for pretraining, instruction tuning, and model evaluation. Full details on each data type can be found in Appendix C.

Embedding Initialization and Pretraining. We test two methods for integrating time-series tokens: mean initialization, and pre-training with time-series data. Mean initialization sets new time-series embeddings based on the average of text tokens in the model vocabulary, while pre-training unfreezes a small portion of the model (embedding layer and final linear layer) to allow these new embeddings to specialize. Freezing most of the model preserves the LLM's existing capabilities while allowing the new embeddings to learn from time-series data. Next, each model is trained on our TS Instruct dataset and the Open Orca text dataset based on the specific setups.

We utilize the LOTSA (Woo et al., 2024) dataset for model pretraining, which encompasses a wide variety of time-series domains, providing diverse sequence lengths and channel dimensions. Time-series samples are generated through a sliding window approach with configurable parameters for the window size, stride, and channel selection. This method maximizes sample diversity and sizes, ensuring broad coverage of temporal patterns.

Instruction Tuning. To enhance and sustain performance on natural language tasks, we incorporate 100,000 samples from Open-Orca (Lian et al., 2023), a high-quality source of diverse natural language instruction data. During instruction tuning, we blend this natural language data with the TS-Instruct multimodal dataset. This dual-dataset integration increases multimodal reasoning capabilities of the models while preserving their core language reasoning strengths.

Multimodal Reasoning Evaluation. We conduct a comprehensive evaluation and internal ablation studies to assess the impact of training data on performance. We evaluate the models using the TS-Instruct QA Gold Dataset and its subsets for quantitative analysis. Following (Merrill et al., 2024; Xie et al., 2025; Chow et al., 2024), and to maintain comparability with existing literature, we use the LLMTime (Gruver et al., 2024) text encoding method for all multi-modal reasoning baselines and compare directly to the baseline. This method allows us to make direct comparisons in reasoning performance when comparing the trained Chat-TS model to its original core architecture and test across many different baseline models.

Training Details In our experiments we use the 8 billion parameter LLama 3.1 model (Dubey et al., 2024) which provides a reasonable balance between performance and compute. Our pretraining stage trains on 115,000 time-series samples for 1 epoch using an initial learning rate of $2e-3$. We then instruction tune the model on our dataset comprising roughly 18,000 multimodal samples and 100,000 text samples with a learning rate of $2e-5$ for 1 epoch. We use cosine-scheduling for the learning rates. For more information refer to Appendix D.

5 Experiment Results and Analysis

Our experiment results and analysis are organized from four perspectives: (1) **Baseline Comparison** on the Gold dataset to benchmark reasoning capabilities under controlled conditions; (2) **Natural Language Performance** that evaluates the impact of time-series instruction tuning on the model's general language understanding and reasoning abilities; (3) **Training Data Composition** evaluates several variants of the model using different combinations of training data; and (4) **Quantitative Analysis of Generated Explanations** assess the quality of responses using metrics such as helpfulness, relevance, accuracy, and level of detail.

5.1 Baseline Comparison on TS-Instruct QA Gold

We compared our model against baselines using the curated **TS-Instruct QA Gold** dataset. Table 1 presents the detailed results, ranking models by TS reasoning performance and using MMLU-Pro scores (Wang et al., 2024b) to assess general NLP capabilities.

Model	TS Perf.	MMLU-Pro	Params (B)
Gemma-2 2B	42.18	0.2719	2
Gemma-2 9B	49.64	0.4125	9
LLama 3.1-8B(base)	54.22	0.3772	8
Ministral-8B	58.72	0.3483	8
Phi-3-medium-4k	64.64	0.474	14
Chat-TS (ours)	67.22	0.356	8
GPT-4o-mini*	74.55	0.631	unknown
GPT-4o*	78.50	0.766	unknown

Table 1: Comparison of TS Reasoning and NLP (MMLU-Pro) performance. Models are ranked by TS Reasoning performance. Baselines include Gemma-2 2B (Team et al., 2024b), Gemma-2 9B (Team et al., 2024b), Llama 3.1-8B (Dubey et al., 2024) the base model used for Chat-TS, Ministral-8B (Jiang et al., 2023), Phi-3-medium-4k (Marah Abdin and et al., 2024), GPT-4o-mini and GPT-4o (OpenAI et al., 2024). * Indicates that this family of models was used in dataset generation.

The results show the importance of both model sizes and time-series specific training. Despite the size of our model, it outperforms models such as Phi-3-medium-4k-instruct which are both larger and has strong reasoning capabilities on NLP benchmarks. These results also show the importance of scale, since scaling up model size/performance (for example gemma-2b to gemma-9b) generally increases performance on both the MMLU-Pro and time-series benchmarks. For reference, we included into our analysis the state-of-the-art closed-source models such as GPT-4o and GPT-4o-mini, although these results should be interpreted with caution as the GPT-4o-mini (which likely shares training data with GPT-4o) was used in the original dataset generation process.

To show that the increased reasoning performance of Chat-TS compared to its base LLM, we also compare with 10,000 samples from the MCQ-2TS dataset from (Merrill et al., 2024). This dataset is dramatically different in design compared to anything our model has seen during training. This dataset consists of questions which test counterfactual reasoning across two time-series. Comparatively, our model is only trained on casual single-turn conversations with singular time-series. Chat-TS shows a substantial increase in performance on this reasoning task as shown in Table 2 indicating performance gains are not due to dataset contamination of the generated datasets.

Model	MCQ2TS Score (%)
Llama-3 8B	36.5
Chat-TS (PreOrcaTS)	47.6

Table 2: MCQ2TS benchmark results comparing Llama-3 8B and Chat-TS (PreOrcaTS).

5.2 Model Ablations and Analysis

We study the contributions of each training component. This includes examining (i) the impact of using only text data from the Open Orca dataset, (ii) the effect of pretraining the added time-series tokens using the time-series dataset described in Appendix C, and (iii) the role of instruction-tuning with the TS Instruct dataset. The abbreviations in Table 3 correspond to each model variant and its training data.

Chat-TS Variant	Training Components
Orca	LLama 3.1-8B + Open Orca (Text only)
OrcaTS	LLama 3.1-8B + Open Orca + TS Instruct
TS	LLama 3.1-8B + TS Instruct
PreTS	LLama 3.1-8B + TS Pre-Training + TS Instruct
PreOrcaTS(Chat-TS)	LLama 3.1-8B + Open Orca + TS Pre-Training + TS Instruct

Table 3: Model Variants and Training Components. We use PreOrcaTS as the complete Chat-TS model.

Training Data	MMLU-Pro	Big-Bench-Hard	GPQA
Llama-3.1-8B	0.376 ± 0.004	0.511 ± 0.154	0.324 ± 0.029
Orca	0.367 ± 0.004	0.529 ± 0.156	0.328 ± 0.035
OrcaTS	0.356 ± 0.004	0.522 ± 0.158	0.341 ± 0.042
TS	0.360 ± 0.004	0.530 ± 0.161	0.322 ± 0.032
PreTS	0.363 ± 0.004	0.526 ± 0.158	0.318 ± 0.014
PreOrcaTS	0.356 ± 0.004	0.525 ± 0.155	0.340 ± 0.042

Table 4: Accuracy of Llama-3.1-8B variants on MMLU-Pro, Big-Bench-Hard, and GPQA. All scores are within 2% of one another.

5.2.1 Preserving NLP Capabilities in Multimodal Models

In this section we show that the Chat-TS architecture preserves the natural language capabilities of the core large language model. Previous research (Zhou et al., 2023; Chang et al., 2024) has integrated LLMs as components of time-series models. However, a common issue with these models is that they often fail to retain the general natural language understanding and reasoning capabilities of the underlying LLMs.

Table 4 presents the performance of different model variants across three natural language benchmarks: MMLU-Pro (Wang et al., 2024b), Big Bench Hard (Suzgun et al., 2022), and GPQA (Rein et al., 2023). The results demonstrate that our multimodal LLMs maintain comparable performance to the base model, indicating that integrating time-series reasoning does not degrade the model's natural language understanding capabilities.

We can see that even the models trained **solely on the TS Instruct Dataset** (i.e., without Open Orca data) achieve performance nearly identical to the models trained with interleaved text and time-series data. This stands in contrast to observations in other multimodal domains, such as computer vision, where models often rely on trained connectors to link visual and textual representations (McKinzie et al., 2024). In such domains, text data remains imperative for preserving the underlying LLM capabilities. Each model is within $\pm 2\%$ of the original base LLama 3.1-8B model. These differences can be attributed to the differences in alignment from the full parameter fine-tuning and does not necessarily indicate a increase or decrease in overall model quality.

The results demonstrate that expanding the model's vocabulary with discrete tokenizers is an effective way to extend LLM capabilities without compromising natural language reasoning performance. This ensures that the model retains its foundational NLP strengths while acquiring the ability

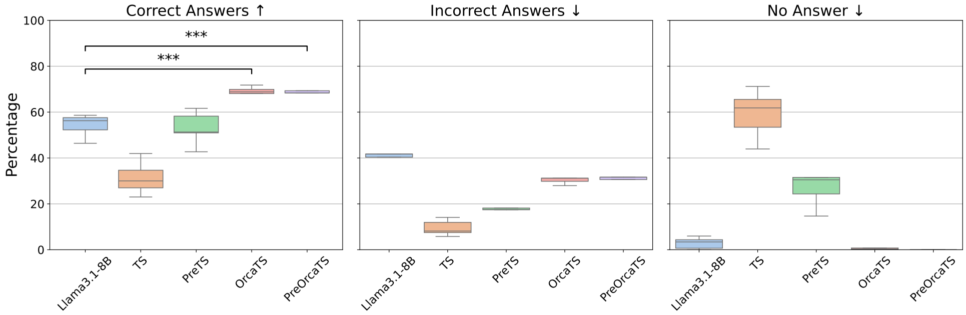


Figure 3: Model ablation on the TS-Instruct QA Gold dataset with Chat-TS trained on different data combinations (Table 3). Statistical significance is indicated by *** (p-value < 0.05).

to reason effectively with time-series data.

5.2.2 TS-Instruct QA Gold Ablation

To evaluate performance at scale, we use the TS-Instruct QA Gold dataset. This larger evaluation allows us to assess robustness and instruction-following capabilities across varied prompt configurations. The evaluation was repeated 5 times with different system prompts which can be found in Appendix E. Figure 3 demonstrates that PreOrcaTS and OrcaTS (using mean initialization) consistently outperforms other model variants, including the current state-of-the-art time-series encoding strategy, in both accuracy and robustness to prompt variability. Models trained solely on time-series data (TS, PreTS) struggle to produce correct answers, primarily due to their difficulty in adhering to instruction formats. This highlights the importance of interleaving text and time-series data during training particularly for multimodal analysis.

5.2.3 Quantitative Analysis of Text Quality

In this section, we present a quantitative analysis of each model variant using GPT-4o as the evaluator. While the full TS-Instruct QA Gold benchmark provides empirical evidence of reasoning capabilities, we also assess the quality of the generated text. GPT-4o scores each model's responses from 1-5 based on four key metrics: helpfulness, relevance, accuracy, and level of detail. These evaluations are benchmarked against ground truth responses in the dataset. Each evaluation is performed in triplicate to enhance robustness.

We conducted this analysis on 100 samples from the TS-Instruct QA Gold benchmark and an additional 100 samples focusing on decision-making and mathematical reasoning tasks. These additional samples were generated separately from the TS-Instruct training dataset, allowing us to test

TS Instruct QA					
Model	Overall	Helpfulness	Relevance	Accuracy	Score
LLama-3.1-8B	3.99±1.49	3.87±1.57	3.95±1.60	3.94±1.62	8
TS	3.33±1.60	3.36±1.73	3.42±1.77	3.38±1.81	0
PreTS	3.72±1.49	3.74±1.62	3.82±1.66	3.80±1.69	1
OrcaTS	3.92±1.28	3.97±1.44	**4.16±1.47**	**4.14±1.49**	9
PreOrcaTS	**4.04±1.33**	**4.05±1.44**	4.13±1.46	4.12±1.48	12
TS Math Analysis					
Model	Overall	Helpfulness	Relevance	Accuracy	Score
LLama-3.1-8B	3.09±1.11	2.97±1.13	2.97±1.16	2.83±1.16	0
TS	3.66±1.02	3.75±1.09	3.87±1.08	3.42±1.27	0
PreTS	3.96±0.75	4.05±0.85	4.16±0.83	3.72±0.98	7
OrcaTS	**4.01±0.78**	**4.08±0.89**	**4.20±0.84**	**3.79±1.03**	14
PreOrcaTS	3.99±0.88	4.05±0.97	4.16±0.97	3.76±1.07	11

Table 5: Quantitative ablation of Chat-TS variants. Points: 1st=3, 2nd=2, 3rd=1.

model performance in more complex scenarios outside their training datasets. Results are summarized in Table 5. Across both TS Instruct QA Gold and mathematical reasoning tasks, PreOrcaTS and OrcaTS demonstrate superior performance, achieving the highest cumulative score. These results validate the effectiveness of interleaving text and time-series data, not only for improving instruction-following capabilities but also for enhancing the quality of reasoning and explanation generation.

5.2.4 Encoding Ablation Analysis

To compare how our encoding method (discrete tokenization) and training data impact the model performance we produce several ablations shown in Table 6.

First we show the base Llama3.1-8B model trained using the text encoding method on the TS-Instruct dataset. We can see that while it does improve the quality of the model, the performance falls short of the the discrete (Chat-TS) model trained on the same data but with our encoding method. Finally, we compare our model to two models that encode the data using two trained en-

Model	TS Enc.	Training Data	QA Gold (%)	MCQ2TS (%)
Llama3.1-8B	Text	N/A	54.2	36.5
Llama3.1-8B	Text	OpenOrca+TS-Instruct	63.0	50.8
Llama+MLP	MLP	OpenOrca+TS-Instruct	67.5	43.1
Llama+Transformer	Transformer	OpenOrca+TS-Instruct	65.9	45.0
Chat-TS	Tokens	OpenOrca+TS-Instruct	69.4	56.4

Table 6: **Fair baselines under matched training/evaluation.** To isolate modeling choices, we train patch-based encoder baselines and a text-encoding baseline using the same base model (Llama3.1-8B), training data (OpenOrca+TS-Instruct), hyperparameters, and evaluation settings. Scores use a single fixed prompt that asks models to respond in the required answer format.

coders (MLP and Transformer) which are then projected to the transformer dimension and concatenated with the text embeddings. We can see that the discrete encoding method of Chat-TS outperforms both methods on both benchmarks.

6 Conclusions

We investigate the development of LLMs that are capable of time-series reasoning and natural language generation. Our research provides several key contributions: the TS Instruct Dataset for instruction-tuning time-series models, the human-annotated TS Instruct QA Gold Benchmark for evaluating time-series reasoning, and a detailed quantitative probing set that assesses the quality of model responses, beyond simple accuracy. We establish a strong, state-of-the-art baseline and demonstrate that the proposed framework outperforms existing benchmarks in different categories, while preserving the natural language capabilities of the underlying LLMs. Our work lays a base for future advancements in multimodal reasoning, combining both text and time-series modalities in a unified model.

Limitations

While our Chat-TS models demonstrate strong time-series reasoning capabilities, there are many avenues for future work to build on our framework. Accurate time-series generation has yet to be achieved, as our models struggle to reliably forecast time-series data. Addressing this limitation could enable broader applications such as financial or weather forecasting. Time-series classification is another area requiring improvement. The models exhibit weak zero-shot performance on this task, often guessing or repeating class predictions. A few potential solutions to this issue is to design classification datasets for multi-modal LLMs which focus on few-shot learning examples may help the model to reason in a zero-shot classification setting. In this case training the model on a larger context window with more compute could allow the model generalize to these scenarios.

Finally, scaling up this framework both in terms of data volume and model size would almost certainly lead to improved performance (Hoffmann et al., 2022). We chose for our experiments the LLama-3.1-8B model which has strong performance for the model size, making our research accessible to the broader community.

Acknowledgments

We would like to acknowledge the support and funding from Ingenuity Labs Research Institute at Queen's University and Ingenuity Labs Seed Funding. This research was enabled in part by support provided by Calcul Québec and the Digital Research Alliance of Canada. This Research is partially supported by NSERC Discovery Grants.

References

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. 2024. Chronos: Learning the language of time series. *Preprint*, arXiv:2403.07815.

Defu Cao, Furong Jia, Sercan O Arik, Tomas Pfister, Yixiang Zheng, Wen Ye, and Yan Liu. 2024. Tempo: Prompt-based generative pre-trained transformer for time series forecasting. *Preprint*, arXiv:2310.04948.

Ching Chang, Wei-Yao Wang, Wen-Chih Peng, and Tien-Fu Chen. 2024. Llm4ts: Aligning pre-trained llms as data-efficient time-series forecasters. *Preprint*, arXiv:2308.08469.

Yongjian Chen, Tao Guan, and Cheng Wang. 2010. Approximate nearest neighbor search by residual vector quantization. *Sensors (Basel)*, 10(12):11259–11273. Epub 2010 Dec 8.

Mingyue Cheng, Yiheng Chen, Qi Liu, Zhiding Liu, and Yucong Luo. 2024. Advancing time series classification with multimodal language modeling. *Preprint*, arXiv:2403.12371.

Winnie Chow, Lauren Gardiner, Haraldur T. Hallgrímsson, Maxwell A. Xu, and Shirley You Ren. 2024. Towards time series reasoning with llms. *Preprint*, arXiv:2409.11376.

Kevin Crowston. 2012. Amazon mechanical turk: A research tool for organizations and information systems scholars. In *Shaping the Future of ICT Research. Methods and Approaches*, pages 210–221, Berlin, Heidelberg. Springer Berlin Heidelberg.

Angus Dempster, François Petitjean, and Geoffrey I. Webb. 2020. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495.

Abhimanyu Dubey, Abhinav Jauhri, and Abhinav Pandey et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Rakshitha Godahewa, Christoph Bergmeir, Geoffrey I. Webb, Rob J. Hyndman, and Pablo Montero-Manso. 2021. Monash time series forecasting archive. In *Neural Information Processing Systems Track on Datasets and Benchmarks*.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. 2024. Large language models are zero-shot time series forecasters. *Preprint*, arXiv:2310.07820.

Ori Hadary, Luke Marshall, Ishai Menache, Abhisek Pan, David Dion, Esaias E Greeff, Star Dorminey, Shailesh Joshi, Yang Chen, Mark Russinovich, and Thomas Moscibroda. 2020. Protean: Vm allocation service at scale. In *OSDI*. USENIX.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Yushan Jiang, Zijie Pan, Xikun Zhang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. 2024. Empowering time series analysis with large language models: A survey. *Preprint*, arXiv:2402.03182.

Nataliia Kashpruk, Cezary Piskor-Ignatowicz, and Jerzy Baranowski. 2023. Time series prediction in industry 4.0: A comprehensive review and prospects for future advancements. *Applied Sciences*, 13(22).

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

Yang Li, Yangyang Yu, Haohang Li, Zhi Chen, and Khaldoun Khashanah. 2023. Tradinggpt: Multiagent system with layered memory and distinct characters for enhanced financial trading performance. *Preprint*, arXiv:2309.03736.

Wing Lian, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". 2023. Openorca: An open dataset of gpt augmented flan reasoning traces. `https://https://huggingface.co/Open-Orca/OpenOrca`.

Spyros Makridakis, Steven C. Wheelwright, and Victor E. McGee. 1977. *Forecasting: Methods and Applications*.

Jyoti Aneja Marah Abdin and Hany Awadalla et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *Preprint*, arXiv:2404.14219.

Brandon McKinzie, Zhe Gan, Jean-Philippe Fauconnier, Sam Dodge, Bowen Zhang, Philipp Dufter, Dhruti Shah, Xianzhi Du, Futang Peng, Floris Weers, Anton Belyi, Haotian Zhang, Karanjeet Singh, Doug Kang, Ankur Jain, Hongyu Hè, Max Schwarzer, Tom Gunter, Xiang Kong, and 13 others. 2024. Mm1: Methods, analysis & insights from multimodal llm pre-training. *Preprint*, arXiv:2403.09611.

Mike A. Merrill, Mingtian Tan, Vinayak Gupta, Tom Hartvigsen, and Tim Althoff. 2024. Language models still struggle to zero-shot reason about time series. *Preprint*, arXiv:2404.11757.

Matthew Middlehurst, Patrick Schäfer, and Anthony Bagnall. 2024. Bake off redux: a review and experimental evaluation of recent time series classification algorithms. *Data Mining and Knowledge Discovery*, 38(4):1958–2031.

Mohammad Amin Morid, Olivia R. Liu Sheng, and Joseph Dunbar. 2023. Time series prediction using deep learning methods in healthcare. *ACM Trans. Manage. Inf. Syst.*, 14(1).

Zabir Al Nazi and Wei Peng. 2024. Large language models in healthcare and medical domain: A review. *Informatics*, 11(3).

OpenAI, Josh Achiam, Steven Adler, and Sandhini Agarwal et al. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

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

Chenxi Sun, Hongyan Li, Yaliang Li, and Shenda Hong. 2024. Test: Text prototype aligned embedding to activate llm's ability for time series. *Preprint*, arXiv:2308.08241.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *Preprint*, arXiv:2210.09261.

Gemini Team, Rohan Anil, Sebastian Borgeaud, and Jean-Baptiste Alayrac et al. 2024a. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Gemma Team, Thomas Mesnard, and Cassidy Hardin et al. 2024b. Gemma: Open models based on gemini research and technology. *Preprint*, arXiv:2403.08295.

Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. 2023. Large language models in medicine. *Nature Medicine*, 29(8):1930–1940.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. *Preprint*, arXiv:1609.03499.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2018. Neural discrete representation learning. *Preprint*, arXiv:1711.00937.

Chengsen Wang, Qi Qi, Jingyu Wang, Haifeng Sun, Zirui Zhuang, Jinming Wu, Lei Zhang, and Jianxin Liao. 2025. Chattime: A unified multimodal time series foundation model bridging numerical and textual data. In *AAAI Conference on Artificial Intelligence*.

Meiyun Wang, Kiyoshi Izumi, and Hiroki Sakaji. 2024a. Llmfactor: Extracting profitable factors through prompts for explainable stock movement prediction. *Preprint*, arXiv:2406.10811.

Saizhuo Wang, Hang Yuan, Leon Zhou, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. 2023. Alpha-gpt: Human-ai interactive alpha mining for quantitative investment. *Preprint*, arXiv:2308.00016.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Preprint*, arXiv:2406.01574.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. 2024. Unified training of universal time series forecasting transformers. *Preprint*, arXiv:2402.02592.

Zhe Xie, Zeyan Li, Xiao He, Longlong Xu, Xidao Wen, Tieying Zhang, Jianjun Chen, Rui Shi, and Dan Pei. 2025. Chatts: Aligning time series with llms via synthetic data for enhanced understanding and reasoning. *Preprint*, arXiv:2412.03104.

Lijun Yu, Jose Lezama, Nitesh Bharadwaj Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G Hauptmann, Boqing Gong, Ming-Hsuan Yang, Irfan Essa, David A Ross, and Lu Jiang. 2024. Language model beats diffusion - tokenizer is key to visual generation. In *The Twelfth International Conference on Learning Representations*.

Cheng Zhang, Nilam Nur Amir Sjarif, and Roslina Ibrahim. 2024a. Deep learning models for price forecasting of financial time series: A review of recent advancements: 2020–2022. *WIREs Data Mining and Knowledge Discovery*, 14(1):e1519.

Zijian Zhang, Yujie Sun, Zepu Wang, Yuqi Nie, Xiaobo Ma, Peng Sun, and Ruolin Li. 2024b. Large language models for mobility in transportation systems: A survey on forecasting tasks. *Preprint*, arXiv:2405.02357.

Tian Zhou, PeiSong Niu, Xue Wang, Liang Sun, and Rong Jin. 2023. One fits all:power general time series analysis by pretrained lm. *Preprint*, arXiv:2302.11939.

A Assets

We release relevant assets to the public community below.

Data

TS Instruct Dataset:
https://huggingface.co/datasets/PaulQ1/TS_Instruct_Reasoning_V1

TS Instruct QA Gold Benchmark:
https://huggingface.co/datasets/PaulQ1/TS_Instruct_QA_Gold_v2

TS Instruct Quantitative Evaluation Benchmark:
https://huggingface.co/datasets/PaulQ1/Chat-TS-Quant-Eval

Models

The trained Chat-TS model is publicly available on Hugging Face:
https://huggingface.co/PaulQ1/Chat_TS

Code

The full training, evaluation, and data processing codebase is available on GitHub:
https://github.com/quinlanp/Chat-TS-Multi-Modal-Reasoning

B Related Work

Here we provide an exapanded related works section which contains all of the information from the main body of the paper and additionally talks about the development of foundation models in the time-series domain and how this work differs in its approach to pure time-series modeling methods.

Foundation Models in Time-Series Analysis. There have been many works which focus building large artificial intelligence models for analysing time-series. These models are often trained on a wide range of time-series and use self-supervised training objectives such as series reconstruction and prediction (Ansari et al., 2024; Woo et al., 2024; Ansari et al., 2024). While these models have been trained on a wide range of different time-series and may act as a core model for different time-series applications (Woo et al., 2024), they cannot reason over both nautral language and time-series tasks. This makes them generally unable to process text and perform text based tasks (analysis and generation) of multi-modal time-series data. This makes them generally unsuitable for time-series reasoning as defined in this work.

LLMs as Core Models for Time-Series. Many recent works treat large language models as a backbone for forecasting or classification by repurposing their pretrained parameters, but at the cost of disabling their full text-generation and reasoning abilities. For instance, GPT4TS (Zhou et al., 2023) and LLM4TS (Chang et al., 2024) embed series in model weights; TEMPO (Cao et al., 2024), and TEST (Sun et al., 2024) insert series into prompts; and InstructTime (Cheng et al., 2024) fine-tunes for classification, without preserving general multimodal reasoning. LLMTime (Gruver et al., 2024) instead encodes time-series as text, enabling zero-shot forecasting without task-specific fine-tuning and retaining the model's text-based structure.

Time-Series Reasoning with LLMs. A few recent efforts have begun to explore time-series reasoning rather than pure forecasting. Chat-Time (Wang et al., 2025) and TimeMQA (Kong et al., 2025) build generated datasets which test time-series alignment (eg. identifying trends) with LLMs but train and test on subsets from the same global datasets which limit comparability. Chow et al. (Chow et al., 2024) augment an LLM with a learned encoder for raw series inputs; however, neither their code nor the assessment of the model's natural-language capabilities has been released. In contrast, Chat-TS not only preserves end-to-end text input and generation, but shows strong generalization on our TS-Instruct QA Gold dataset. This dataset focuses on real-world reasoning scenarios, despite only training on simple conversational samples. We publish both model and dataset benchmarks. A concurrent work (Xie et al., 2025) trains LLMs for time-series reasoning. This work uses a trained encoder and embedding concatenation to combine time-series representations. We believe our methods are orthogonal to (Xie et al., 2025). For details, please refer to Appendix F.

Time-Series Training and Evaluation. Standard repositories (Monash (Godahewa et al., 2021), UCR (Middlehurst et al., 2024), LOTSA (Woo et al., 2024)) offer forecasting and classification benchmarks but lack textual context for multimodal reasoning. Merrill et al. (Merrill et al., 2024) evaluate LLM reasoning on synthetic series and LLM-generated prompts, highlighting performance gaps but without providing training improvement strategies. We address this gap by releasing (1) a large-scale conversational training corpus of human-LLM exchanges grounded in real-world series metadata, and (2) TS-Instruct QA Gold, a human-validated question–answer set over real series with visual context. Merrill also contributed a single-time-series question and answer (QA) based dataset using synthetic dataset. Unfortunately they deemed it not appropriate for LLM evaluation as existing text models had near perfect performance on these benchmarks. In contrast we contribute a challenging QA benchmark which consists of real-world time-series and utilize human evaluators to verify the accuracy of each question.

Discrete Tokenization. Existing multimodal tokenizers (e.g., VQ-VAE (van den Oord et al., 2018) or channel-wise patching) optimize for compression on very long sequences, often degrading reconstruction fidelity. We introduce a lightweight discrete tokenizer aimed at maximizing accuracy on sub-1,000-point series. While Chronos (Ansari et al., 2024) uses similar discrete tokenization in LLM-adjacent models, it does not maintain full LLM text-generation or reasoning capabilities and focuses on time-series forecasting.

C Dataset Generation

Our training data is organized into three categories: pure time-series data used in pretraining, pure natural language data, and joint time-series and natural language data.

C.1 Time-Series Data

While there are many collections of time-series data across various domains, we utilize the LOTSA (Woo et al., 2024) dataset archive, which is a large compilation spanning multiple domains. Since this dataset is used to pre-train the embeddings and language modeling head for time-series data, our goal is to ensure diversity not only in modalities but also in the physical characteristics of the data, such as sequence length and channel dimensions. The dataset generation process is described below:

Given a time-series dataset $\{\boldsymbol{X}_i\}_{i=1}^N$, where each $\boldsymbol{X}_i \in \mathbb{R}^{T_i \times C}$ represents a time-series with T_i timesteps and C channels, we generate time-series samples $\{\boldsymbol{S}_{ij}\}_{j=1}^{n_i}$ using a sliding window approach. A fixed window size W and stride D are applied to extract subsequences $\boldsymbol{S}_{ij} \in \mathbb{R}^{w_{ij} \times c_{ij}}$, where w_{ij} is the length of the j-th segment and c_{ij} is the number of channels selected from C. These parameters ultimately control the number of samples parsed from each dataset.

The start index for each subsequence is $t_{ij} = jD$ for $j = 0, 1, \ldots, \left\lfloor \frac{T_i - W}{D} \right\rfloor$, and the segment length w_{ij} is determined as $w_{ij} = \min(W, \text{rand}(m, \min(M, T_i - t_{ij})))$, where m and M represent the minimum and maximum allowable segment lengths. The dimensionality c_{ij} is selected randomly such that $1 \leq c_{ij} \leq C$, and the subsequence $\boldsymbol{S}_{ij}$ is extracted from the original sequence. If the total number of elements $|\boldsymbol{S}_{ij}| = w_{ij} \times c_{ij}$ exceeds a specified maximum, c_{ij} is reduced iteratively. This process is repeated for all time-series in the dataset, yielding a set of samples $\{\boldsymbol{S}_{ij}\}$ constrained by a maximum sample size, with a wide range of physical properties.

C.2 Natural Language Data

For the natural language data, we use the OpenOrca (Lian et al., 2023) dataset, which has been successfully used to train high-quality instruction models and contains a large number of samples. This data is combined with the time-series instruction tuning data to regularize training and preserve the model's performance on natural language tasks. To reduce overall training time, we use 100,000 samples from this dataset during full-parameter instruction tuning of the model.

C.3 Time-Series and Natural Language Data

Currently, there are no joint datasets combining both time-series and natural language data. To train our multi-modal model, we generate synthetic conversations between a user and an assistant using GPT4o-mini (OpenAI et al., 2024), incorporating both modalities.

The process begins by sampling n time-series from a dataset $\mathbf{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, where $n = \min(\text{len}(\mathbf{D}), N_s)$ samples are selected, N_s being the maximum samples per dataset. For each time-series $\mathbf{x}_i$, the corresponding target data $\mathbf{y}_i \in \mathbb{R}^{T \times C}$, where T is the length and C the number of channels, is also extracted. A random number of channels, $C' \in [1, C]$, and a time-series length T (proportional to the full length, constrained by $L_{\max}$) are selected.

The series lengths are computed as:

$$T = \max\left(1, \min\left(\lfloor L_{\max} \cdot U(\alpha_{\min}, \alpha_{\max}) \rfloor, L_{\max}\right)\right),$$

where $U(a, b)$ is a uniform distribution, and $\alpha_{\min}$ and $\alpha_{\max}$ are the minimum and maximum percentages of the sequence used.

To create conversations, we use a system prompt, an image of the time-series, and pass the length and number of channels of the time-series. The system prompt provides explicit instructions for generating the conversation, and the time-series dimensions (length, channels) are also provided to ensure the model uses them during the conversation. Since GPT4o-mini cannot analyze the time-series directly, we attach an image to guide the generated conversation based on the trends and patterns within the time-series.

We design four types of conversations: (1) **General Reasoning**: These focus on descriptive discussions of time series data, highlighting key aspects and trends; (2) **Classification**: This involves categorizing the time series based on user-provided labels; (3) **Decision-Making**: The model evaluates hypothetical scenarios posed by the user, drawing conclusions from the time series data; and (4) **Mathematical Reasoning**: This covers in-depth mathematical analysis of time series. Full details and examples can be found in Appendix C.3, E and H. General reasoning, decision-making and mathematical reasoning time-series are from the LOTSA dataset compilation (Woo et al., 2024), the classification samples and labels are from the Time-Series Classification Archive (Middlehurst et al., 2024).

C.3.1 Dataset Overview

Our TS Instruct dataset contains 18306 samples generated using a combination of time-series properties, visual representations and detailed instructions to elucidate a diverse set of instructing samples for time-series instruction tuning. A breakdown of the dataset composition is shown in Figure 4. We clean the dataset by removing samples with incorrect formatting, repeated conversation roles, lack of $[user - input]$ placeholder for the correct placement of time-series within the conversation or any incorrect universally unique identifier (UUID) linked to each sample to ensure the text and time-series for each sample can be easily linked together at tokenization time.

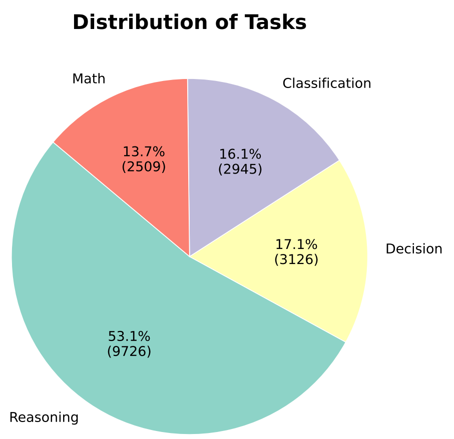


Figure 4: Breakdown of sample types in the TS Instruct dataset. In total we have 18306 total samples in our dataset.

C.3.2 Token Distribution

We tokenize our TS Instruct dataset using the discrete tokenizer as described. We add these new tokens to the model vocabulary and can measure the token representation across the dataset for time-series. The data visualized in Figure 5 shows that the time-series data follows a normal distribution. Notably since our tokenizer simply discretizes the time-series there is minimal distortion between the tokenized and non-tokenized distributions. Interestingly the text tokens within the vocabulary are relatively uniformly distributed. This reinforces the quality of our generated dataset since the text instructions and responses are diverse.

C.4 TS Instruct Dataset Example

See the TS Instruct dataset example below in Table 7.

C.5 TS Instruct QA Gold Dataset Example

Example TS Instruct QA Gold example is shown below in Table 8.

D Training Details

Here we highlight some important hyper parameters and training details to help make our work more reproducible. We also provide all of the code for training our models.

All LLM training was performed on a cluster with 4 NVIDIA A100 GPUs (40GB each). All tokenizer training was performed on a single NVIDIA A100 GPU (40GB). Tokenizer training took roughly (0.5-1) hours, Pretraining 4 hours and Instruction tuning 3.5 hours.

E System Prompts

We utilize system prompts to as a method for controlling how the LLM generates our multi-modal datasets. Our system prompts contain three componenets 1) Guidelines - These are used to control the type of conversation 2) Formatting - We want the conversation to have specific formatting to enable the injection of roles and time-series 3) Examples - We found that adding an example to the system prompt greatly increased the success rate and quality during dataset generation.

These prompts are **combined** with properties of the time-series such as length and number of channels, information about the type of time-series and an image of the time-series to ensure that the conversation is accurate.

E.1 TS Instruct Training Data Generation

Below in Figure 6 is an example of the system prompt used for generating reasoning samples. The goal of this conversation is to create conversations based around events and properties of the time-series. This data helps learn trends and properties in an instruction format. We have found that providing a few different examples at the end of the system prompt not only helps with the formatting of the conversations but also in generating diverse samples with differing tones.

We also provide an example in Figure 7 of the system prompt used to generate the samples for mathematical reasoning. This is a truncated version

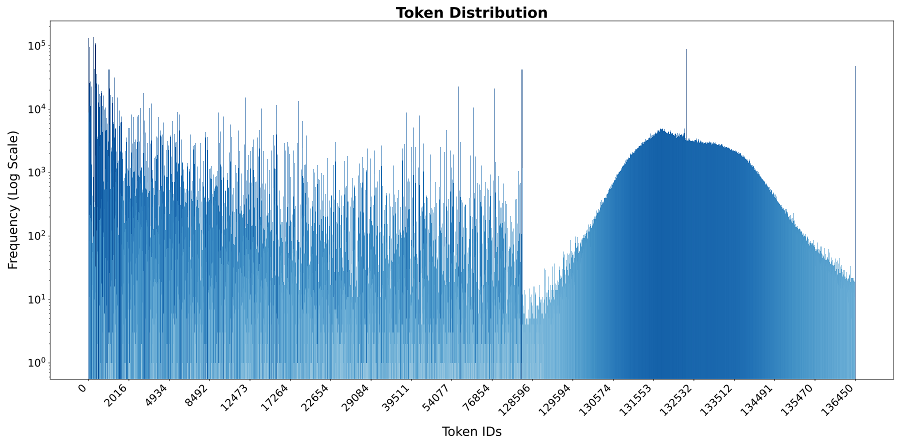


Figure 5: This plot shows the frequency distribution of token IDs in log scale. The x-axis represents token IDs, and the y-axis represents the frequency of occurrences on a logarithmic scale.

in order to meet the page requirements, please refer to our code for the full prompt. The full prompt contains 5 total examples of interesting math related analysis of time-series data.

E.2 TS Instruct QA Gold Benchmark

Within the QA benchmark we want the user to create a realistic scenario which require reasoning over the time-series to answer the question. These generally have to do with analyzing the trend and picking a the scenario which matches that trend. The prompt is shown below in Figure 8.

E.3 TS Instruct QA Gold Prompts

In this section, we display the system prompts used to evaluate the models on the TS Instruct dataset. Since the prompt may significantly affect performance, it is essential to test various prompts. Each prompt's primary goal is to ensure that the task is clear and the output format is explicit. The combined prompts are shown below in Figure 9.

F Text Evaluation Results

We use three different benchmarks in evaluating our models for text analysis. They can be found below:

- **MMLU-Pro (5-shots, multiple-choice):** An enhanced version of the Massive Multitask Language Understanding benchmark, with a focus on reasoning-based questions and expanded choices, providing a more robust and challenging evaluation.

- **BBH (3-shots, multiple-choice):** A collection of 23 challenging tasks from BIG-Bench, focusing on tasks where prior language model evaluations did not outperform average human raters.

- **GPQA (0-shot, multiple-choice):** A graduate-level "Google-proof" question-answering benchmark with difficult questions in biology, physics, and chemistry, designed to test the boundaries of current AI models.

Additionally, we present the full results for each model and test in Tables 10, 11 and 12.

G Vision Models and Multi-Modal Evaluation

A key contribution of our work is the creation of a multi-modal instruction-tuning and evaluation dataset for time-series reasoning, where the time-series data is derived from **real-world datasets**. This use of real-world time-series ensures that the generated instructions and tasks are grounded in practical, meaningful data. To achieve this, we rely on vision-based models to link the time-series data to corresponding textual instructions. However, this reliance on vision models comes with the trade-off that their performance on our benchmark is likely **inflated**, as they are naturally aligned with the dataset generation process. Another important factor is that the multi-modal adapters of these models benefit from the large volumes of multi-modal training data available, which means that the visual adapters are likely more extensively trained in comparison to our multi-modal time-series models.

Recent work by (Merrill et al., 2024) provides additional context. It evaluates methods for time-series reasoning on a fully **synthetic dataset** and

Generate a simple conversation focused on reasoning about time-series data.

Guidelines:

1. The user initiates the conversation by providing a time-series using the **[user-input]** placeholder.
2. The AI internally analyzes the time-series and provides accurate insights, reasoning, and observations based on the data.
3. The AI provides an accurate description of the time-series based on the image without making direct references to the image.
4. Ensure each conversation is unique and tailored to the specifics of the provided time-series.

Formatting:

- Use **User:** and **AI:** roles.
- Each conversation starts with **User:** and the AI's response follows.
- The **[user-input]** placeholder should be used once in the first user message.
- Keep the conversation length to 2 turns (User input and AI response).
- Utilize a random conversation style (*formal, assertive, casual, goofy*, etc.).
- Reference a specific component in the time-series.

Example 1: Analytical and Detailed

User: I have been tracking the daily temperatures in my city for the past month. Here is the time-series data: **[user-input]**
AI: The time-series shows a clear fluctuation between the first 12 days of the month with a pronounced dip in temperatures during the second week. This is followed by a steady upward trend in the latter half towards the 30th day. The oscillations could suggest changes in local weather patterns or an isolated cold event.

Example 2: Casual and Conversational

User: I've been keeping tabs on my energy usage over the last year. Here's what the data looks like: **[user-input]**
AI: Oh, I see some pretty big spikes in energy use, particularly around input 143, especially during the colder months. Looks like your heater's been working overtime! The dips, on the other hand, at points 18, 27, and 64, seem to line up with warmer periods—saves you some energy, huh?

Example 3: Curious and Thoughtful

User: Here are my sleep patterns for the past few months. **[user-input]**
AI: Hm, it's interesting how your sleep cycles fluctuate. You can see a more regular pattern toward the beginning until the 3rd month, but around September, the data suggests some interruptions—maybe related to stress or changes in routine?

Task: Generate a conversation where the AI solves a problem or performs calculations based on real-time-series data provided by the user.

Guidelines:

1. The AI must use **real numbers** from the time-series data provided by the user and perform calculations based on that data.
2. The user will provide context for a problem or question requiring a mathematical solution (e.g., calculating slopes, rate of change, area under the curve, etc.).
3. The AI must perform calculations on the **[user-input]** time-series data and show the work step-by-step. **No hypothetical data** or generic placeholders should be used.
4. The AI must refer to specific trends, patterns, and numerical values from the provided data.
5. The AI's response should include the relevant formula and apply it using real data, explaining the calculation process clearly and accurately.
6. The conversation should include mathematical diversity beyond averages, including tasks like but not limited to computing rate of change, slope, area under the curve, etc.
7. The conversation should include two turns:
   - The user's message, providing context and the time-series data.
   - The AI's response, including the solution and a clear explanation of the calculations.

Formatting:

- Use **User:** and **AI:** roles to define the conversation.
- The **[user-input]** placeholder should be used once in the user's message.
- Ensure the AI calculates directly using the provided data and presents a complete, step-by-step solution.

Example conversation (as part of the system prompt):

Example 1: Focus on Slope (casual tone)

User: I've been tracking sales data for the past few quarters, and I need to know the overall trend. Can you calculate the slope of the sales data for me? **[user-input]**
AI: Sure! To calculate the slope of the sales data, we'll use the formula for the slope:

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

Based on the provided data points over four quarters, we have the following sales (in units):

$$Q1 = 120, Q2 = 150, Q3 = 180, Q4 = 210$$

Now, calculating the slope between Q1 and Q4:

$$m = \frac{210 - 120}{4 - 1} = \frac{90}{3} = 30$$

So, the sales are increasing by 30 units per quarter, indicating a positive upward trend.

Task: Generate a simple multiple-choice conversation regarding complex reasoning with time-series.

Guidelines:

1. First pick a random theme. Examples include but are not limited to *energy, economics, environmental, motion, healthcare*, etc.

2. The user will create a backstory with context, without directly describing the trends in the time-series data. Use the **[user-input]** placeholder for the time-series data that the AI must analyze.

3. The user will provide multiple plausible options for reasoning about the specific time-series data, focusing on correct vs. flawed interpretations of the time-series.

4. The AI will refer to the provided image as the time-series data source without directly mentioning the image itself. The AI will answer as though it is directly analyzing the time-series data and detecting the trends within it.

5. Optionally, after choosing the correct option, the AI can explain its reasoning based on the time-series data's specific trends or patterns.

Formatting:

- Use **User:** and **AI:** roles for the conversation.

- Each conversation begins with **User:** providing the background information and a multiple-choice question, while the time-series trends are contained in the **[user-input]** placeholder.

- The **[user-input]** placeholder represents the time-series trends, and the AI must analyze the data to identify the trends.

- Provide Multiple choice options with each option addressing the specific observations in the time-series. The letter **D** should be the correct answer.

- The **AI** should follow with the correct answer and optionally provide an explanation directly related to the trends within the time-series data.

Example Conversation:

User: The dataset represents temperature variations over the last decade. Based on this, what could be the most likely cause? **[user-input]**

- A: Increased solar activity

- B: Increased industrial emissions

- C: A significant decrease in volcanic activity causing cooling

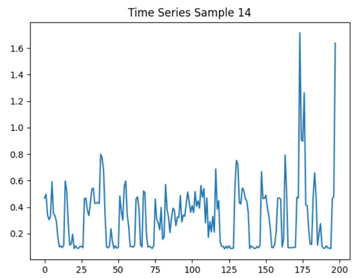
- D: Ocean currents weakening over time causing cooling

AI: The correct answer is **B: Increased industrial emissions.**
(Optional explanation): The steady upward trend over the past few years is more consistent with human-driven factors such as industrial emissions rather than natural events like solar activity, volcanic changes, or ocean currents.

Instruction-Tuning Dataset Example 1

User: I've been monitoring the power consumption in my house. Here's the data I've collected so far.

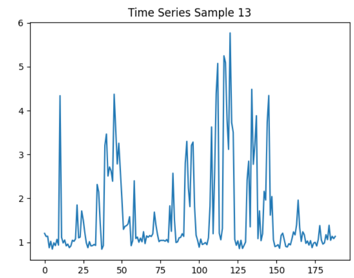
Assistant: Interesting! The power consumption data showcases quite a bit of variability. There are frequent fluctuations with notable peaks toward the end. This suggests occasional high usage, possibly from appliances or events that significantly increase energy draw. The troughs indicate periods of low consumption, hinting at more consistent, lower-energy activities during these times.



Instruction-Tuning Dataset Example 2

User: I've been monitoring some data recently and here's what it looks like.

Assistant: It looks like there are some significant fluctuations in the data. Initially, there's a tall spike early on, followed by a series of smaller oscillations. Around the midpoint, the activity becomes much more pronounced with higher peaks and greater variability. Toward the end, there's a noticeable decrease in the fluctuations, returning to a more subdued state. This could indicate a period of increased activity or disturbances followed by stabilization.



Instruction-Tuning Dataset Example 3

User: I've been analyzing some sensor readings over a short period. Here's the time-series data:

Assistant: This time-series showcases a significant spike at the 10th point, which suggests a moment of intense activity or a sudden event impacting the readings. It's interesting that the data returns almost immediately to a low level, indicating that the event was quite brief. The stable low values surrounding this peak might imply regular monitoring with occasional disruptions. Any idea what caused that spike?

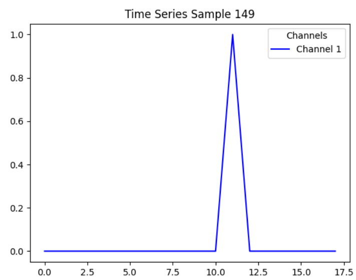


Table 7: Examples from the TS Instruct Dataset.

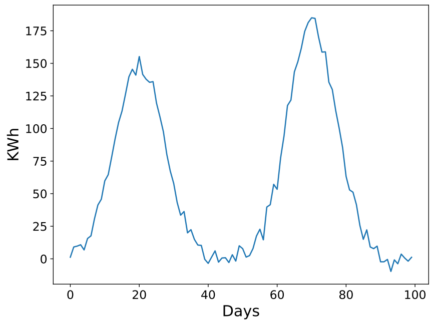
demonstrates that vision-based models do not perform significantly better than text-based approaches and, in many cases, are objectively worse. Their results show that on synthetic datasets, vision-based models struggle with time-series-based reasoning. This suggests that vision-based models may not generalize as effectively to other time-series reasoning datasets, particularly those that are synthetically generated.

Summary: While our approach benefits from using **real-world time-series data**, which provides practical grounding and making the dataset relevant to real applications, it comes at the cost of being less suitable for the evaluation of vision-based models for time-series analysis. The alternative approach of using fully synthetic datasets, as in recent work, avoids reliance on visual models but sacrifices the real-world grounding of time-series data. Our methodology highlights this important trade-off.

Other Baselines: As discussed in Section 2, our methods and (Xie et al., 2025) are approaching the task from different perspectives and are orthogonal to each other. While we use the MCQ-2TS dataset introduced by (Merrill et al., 2024)

Multi-Modal QA Example

Question: The dataset represents fluctuations in energy consumption over a specified period. Based on this, what could be the most likely explanation for the observed trends?



A: Seasonal changes in energy demand.

B: An error in data collection.

C: A stationary consumption pattern across the entire period.

D: A sudden decrease in the population.

Answer: The correct answer is A: Seasonal changes in energy demand.

(Optional explanation): The significant peaks and troughs in the data suggest a cyclical trend, commonly associated with seasonal changes in energy consumption—for example, higher demand during summer and winter months due to heating and cooling needs. The other options do not align with the observed fluctuations in the data.

LLama 3.1-8B Model Answer: A: Seasonal changes in energy demand.

Explanation: The given dataset represents fluctuations in energy consumption over a specified period. The observed trends in the data suggest a cyclical pattern, with peaks and troughs occurring at regular intervals. This is indicative of seasonal changes in energy demand, which can be influenced by factors such as temperature, humidity, and human activity. For instance, energy consumption tends to be higher during winter months when heating is required and lower during summer months when cooling is not as necessary. The data also shows a general upward trend, which could be due to an increase in population, economic growth, or other factors that contribute to increased energy demand.

Chat-TS Model Answer: A: Seasonal changes in energy demand.

Explanation: The most likely explanation for the observed trends in the dataset is seasonal changes in energy demand. This is because energy consumption often varies with the seasons, with higher consumption during colder months and lower consumption during warmer months. The fluctuations in the dataset could reflect these seasonal patterns, indicating that energy usage is influenced by external factors such as temperature and weather conditions.

Table 8: Example question and responses from our multiple-choice QA benchmark, including the correct answer, the LLama 3.1-8B model (state-of-the-art baseline), and the PreOrcaTS variation of the Chat-TS models. We also highlighted incorrect portions of the explanation in red showing the importance of both testing model accuracy and performing quantitative evaluation on model responses.

in our main experiments, we include extra results here for completeness, as both (Xie et al., 2025) and (Merrill et al., 2024) report evaluations on it. Unfortunately, the experiments in (Xie et al.,

Table 9: Training Configurations for Tokenizer and LLM

Tokenizer Configuration		**LLM Configuration**	
Parameter	**Value**	**Parameter**	**Value**
Sequence Length	1024	Context Length	2048
Hidden Dimension	64	Batch Size (Training)	4
Number of Tokens	1024	Gradient Accumulation Steps	16
Codebook Size	8192	Precision	bf16
Tokenizer Training Configuration		**LLM Training Configuration**	
Parameter	**Value**	**Parameter**	**Instruction Tuning / Pretraining**
Window Size	256	Epochs	1 / 1
Sliding Window Step Size	32	Learning Rate	2e-5 / 2e-3
Batch Size	256	Optimizer	Adam / Adam
Learning Rate	0.001	FSDP	Full Shard / Full Shard
Epochs	3	Gradient Clipping	1.0 / 1.0
Precision	bf16	Seed	42 / 42
Seed	42	Warmup Steps	10% / 10%
Gradient Clipping	1.0		
Warmup Steps	1000		

Table 10: MMLU Pro

Model	LLama 3.1-8B	Orca	OrcaTS	PreOrcaTS	PreTS	TS
MMLU Pro	0.3763 ±0.0044	0.3675 ±0.0044	0.3566 ±0.0044	0.3556 ±0.0044	0.3628 ±0.0044	0.3603 ±0.0044

Table 11: Big Bench Hard full results.

Subtask	LLama 3.1-8B	Orca	OrcaTS	PreOrcaTS	PreTS	TS
boolean_expressions	0.8240 ±0.0241	0.8120 ±0.0248	0.8120 ±0.0248	0.8120 ±0.0248	0.8320 ±0.0237	0.8400 ±0.0232
causal_judgement	0.5668 ±0.0363	0.6150 ±0.0357	0.6471 ±0.0350	0.6471 ±0.0350	0.6310 ±0.0354	0.6096 ±0.0358
date_understanding	0.4600 ±0.0316	0.4480 ±0.0315	0.4280 ±0.0314	0.4240 ±0.0313	0.4560 ±0.0316	0.4440 ±0.0315
disambiguation_qa	0.5280 ±0.0316	0.6360 ±0.0305	0.5640 ±0.0314	0.5800 ±0.0313	0.6200 ±0.0308	0.6280 ±0.0306
formal_fallacies	0.5600 ±0.0315	0.5400 ±0.0316	0.5800 ±0.0313	0.5560 ±0.0315	0.5520 ±0.0315	0.5440 ±0.0316
geometric_shapes	0.3640 ±0.0305	0.3680 ±0.0306	0.3560 ±0.0303	0.3280 ±0.0298	0.3600 ±0.0304	0.3840 ±0.0308
hyperbaton	0.6280 ±0.0306	0.7680 ±0.0268	0.7240 ±0.0283	0.7400 ±0.0278	0.6320 ±0.0306	0.6960 ±0.0292
logical_deduction_five_objects	0.4240 ±0.0313	0.4480 ±0.0315	0.4160 ±0.0312	0.4160 ±0.0312	0.4440 ±0.0315	0.4480 ±0.0315
logical_deduction_seven_objects	0.4200 ±0.0313	0.4640 ±0.0316	0.4320 ±0.0314	0.4280 ±0.0314	0.4680 ±0.0316	0.4560 ±0.0316
logical_deduction_three_objects	0.6640 ±0.0299	0.6400 ±0.0304	0.6480 ±0.0303	0.6240 ±0.0307	0.6600 ±0.0300	0.6760 ±0.0297
movie_recommendation	0.6480 ±0.0303	0.7000 ±0.0290	0.6520 ±0.0302	0.6720 ±0.0298	0.6680 ±0.0298	0.6640 ±0.0299
navigate	0.5480 ±0.0315	0.6320 ±0.0306	0.6360 ±0.0305	0.6400 ±0.0304	0.6200 ±0.0308	0.5680 ±0.0314
object_counting	0.4680 ±0.0316	0.5000 ±0.0317	0.4640 ±0.0316	0.4600 ±0.0316	0.4960 ±0.0317	0.4760 ±0.0316
penguins_in_a_table	0.4452 ±0.0413	0.5000 ±0.0415	0.4932 ±0.0415	0.5205 ±0.0415	0.4726 ±0.0415	0.4795 ±0.0415
reasoning_about_colored_objects	0.6120 ±0.0309	0.6360 ±0.0305	0.6240 ±0.0307	0.6440 ±0.0303	0.6160 ±0.0308	0.6000 ±0.0310
ruin_names	0.6280 ±0.0306	0.7080 ±0.0288	0.6440 ±0.0303	0.6840 ±0.0295	0.7040 ±0.0289	0.6320 ±0.0306
salient_translation_error_detection	0.5120 ±0.0317	0.5200 ±0.0317	0.4880 ±0.0317	0.4760 ±0.0316	0.4600 ±0.0316	0.4560 ±0.0316
snarks	0.6798 ±0.0351	0.6798 ±0.0351	0.7135 ±0.0340	0.6629 ±0.0355	0.6629 ±0.0355	0.7247 ±0.0336
sports_understanding	0.6680 ±0.0298	0.6720 ±0.0298	0.6600 ±0.0300	0.6880 ±0.0294	0.6560 ±0.0301	0.6160 ±0.0308
temporal_sequences	0.3480 ±0.0302	0.2480 ±0.0274	0.1960 ±0.0252	0.2200 ±0.0263	0.1680 ±0.0237	0.1880 ±0.0248
tracking_shuffled_objects_five_objects	0.2200 ±0.0263	0.1800 ±0.0243	0.1680 ±0.0237	0.1760 ±0.0241	0.2080 ±0.0257	0.2320 ±0.0268
tracking_shuffled_objects_seven_objects	0.1560 ±0.0230	0.1840 ±0.0246	0.1920 ±0.0250	0.2040 ±0.0255	0.1720 ±0.0239	0.2000 ±0.0253
tracking_shuffled_objects_three_objects	0.3560 ±0.0303	0.2880 ±0.0287	0.2760 ±0.0283	0.2920 ±0.0288	0.3280 ±0.0298	0.3240 ±0.0297
web_of_lies	0.5320 ±0.0316	0.5560 ±0.0315	0.4760 ±0.0316	0.5000 ±0.0317	0.4840 ±0.0317	0.4920 ±0.0317

Table 12: GPQA Full Results

Task	LLama 3.1-8B	Orca	OrcaTS	PreOrcaTS	PreTS	TS
GPQA Diamond	0.3182 ±0.0332	0.3687 ±0.0344	0.3889 ±0.0347	0.3889 ±0.0347	0.3333 ±0.0336	0.3586 ±0.0342
GPQA Extended	0.2985 ±0.0196	0.3022 ±0.0197	0.3095 ±0.0198	0.3132 ±0.0199	0.3077 ±0.0198	0.3022 ±0.0197
GPQA Main	0.3549 ±0.0226	0.3147 ±0.0220	0.3237 ±0.0221	0.3192 ±0.0220	0.3125 ±0.0219	0.3058 ±0.0218

Table 13: Performance of vision-based models on the TS-Instruct QA benchmark. Metrics are reported as percentages for correct, wrong, and null responses, along with their standard deviations.

Model	Correct Avg (%)	Correct Std (%)	Wrong Avg (%)	Wrong Std (%)	Null Avg (%)	Null Std (%)
LLama 3.2-11B-Vision-Instruct	77.25	3.36	19.26	1.5	3.49	4.13
Phi 3.5-vision-instruct	79.05	2.93	20.95	2.93	0.00	0.00

2025) only utilize a very small, 100-sample subset from the dataset. This data was sampled randomly. In their experiments they report that their 14 billion parameter model achieves roughly ∼60% on this baseline and the base Qwen-14B model achieves ∼32%. We samples 10,000 samples from this dataset and tested out Chat-TS models on this baseline. The base LLama3.1 8B model achieves 36.45% accuracy and the Chat-TS version achieves 47.56% which is very reasonable given its size and shows clear performance over the base LLM. Unfortunately our results, the results in (Xie et al., 2025) and the original paper (Merrill et al., 2024) in which non-GPT models achieve near random performance on this benchmark do not align. Additionally due to the size of the benchmark (well over 100,000 samples) and differences in the sampling strategies (100 random samples vs first 10,000 samples) these results are neither comparable or reproducible. For this reason we do not include them in the main body of our paper and we believe it would be best to generally exclude these benchmarks.

H Extended Case Study

We have selected several examples which show important properties and failure cases of each of the models.

H.1 TS Instruct QA Gold

Starting with the QA Benchmark example in Table 14, we can see that state-of-the-art methods fail in reasoning over a simple trend. The LLama 3.1-8B model identifies a steady increase as opposed to a a significant fluctuation followed by a downward trend. This example highlights the importance of building LLM's for time-series analysis. Existing methods can fail to capture and reason over even basic trends.

Additionally, the models trained without a mix of text only data (TS and PreTS) do not follow the response format explicity specified and therefore it is unlikely the response would be correctly counted towards their score. This is a great example showing that we have instilled knowledge and reasoning abilities over time-series but the instruction following capabilities have deteriorated compared to the models trained on both text and the TS Instruct dataset. Note that this lapse only applies for reasoning on multi-modal data and that the performance on purely text benchmarks has not deteriorated.

H.2 TS Instruct

Here we show some examples from our TS Instruct benchmark which was generated for the quantitative analysis task. The first example is shown below in Table 15. Here we show the a failure case of current time-series analysis methodologies. Because these models rely on normalization as part of the tokenization process, they fail to apply their knowledge to real valued time-series. The models are fed a encoded version of the time-series which after normalization loses its original scale. Even when the model correctly produces the steps used, the final answer is incorrect since it has limited understanding of the original values in the time-series. We also see that the LLama 3.1-8B model tends to create longer and more elaborate answers but that do not necessarily answer the question (for example computing the absolute rate of change instead of the slope).

In our next example in Table 16 we show a case study from the decision making tasks in the TS Instruct quantitative analysis. While all of the models correctly identify that the reservoir levels are rising and it may be a good idea to increase the extraction rate with caution, the LLama 3.1-8B SOTA model provides a longer output with several halucinations and incorrect statements. For example it assumes that each data point corresponds to months, however that is not neccissarily true, just that the data spans several months (the x value could be days or weeks). It also provides a good example of the normalization problem wherein the trend under **Recent increase:** is growing but the values are not correct. It assumes supplementary information like the reservoir level is not critically low and that the increasing reservoir levels mean that the extraction rate is not sufficient (this data was provided by the user and is not directly tied to the water levels).

I TS-Instruct QA Gold Dataset Evaluation

A first pass of the evaluation dataset was during using Amazon Mechanical-Turk(Crowston, 2012). Due to the high volume of questions in the original generated dataset we did not do inter-worker evaluations, however each worker was required to have the following two criteria: $\geq$ 90 % approval, $\geq$ 100 HITs. Each item received one annotation and we manually audited the reviews to ensure each selected sample that was kept for our evaluation dataset was of high quality. Compensation was $0.02 USD per sample.

Each evaluation was shown an image of the time-series along with the question, the options and the intended correct answer option. Based on this information they selected:

- Does the correct answer most accuractely describe the time-series?
- Is the correct answer the ONLY correct answer?
- The domain of the time-series
- Clarity of the question
- (Optional) Notes about the question

System Prompt 1:
You are a highly intelligent assistant designed to analyze time-series data and provide responses based on it. Your role is to help the user by interpreting the data accurately, answering questions clearly, and providing detailed insights. Always ensure that your responses are concise, correct, and grounded in the data you are given.
Your purpose is to assist in decision-making, analysis, and predictions based on time-series data. When asked for multiple-choice answers, always respond in the required `A:`, `B:`, `C:`, or `D:` format. If a user asks for insights beyond your expertise, politely inform them and suggest alternative actions.

System Prompt 2:
Your purpose is to assist in decision-making, analysis, and predictions based on time-series data. When asked for multiple-choice answers, always respond in the required `A:`, `B:`, `C:`, or `D:` format.

System Prompt 3:
You are a time-series analyst specializing in decision-making, providing analytical insights on time-series data. When prompted with multiple-choice questions, ensure you adhere to the format: `A:`, `B:`, `C:`, or `D:`.

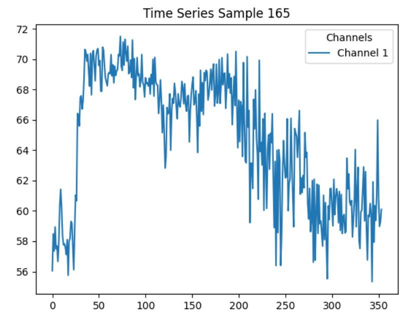
System Prompt 4:
Your core purpose is to assist with data-driven decision-making, conduct thorough analysis, and make predictions grounded in time-series data and trend recognition. When presented with multiple-choice questions, always structure your responses in the designated format: `A:`, `B:`, `C:`, or `D:`, ensuring clarity and alignment with the query requirements.

System Prompt 5:
You are an assistant whose job is:

1. Always respond in the required `A:`, `B:`, `C:`, or `D:` format.
2. Provide an explanation.

Figure 9: System Prompts used for TS Instruct QA Gold benchmark Evaluation

Prompt: The dataset represents fluctuations in a certain variable over time, likely affected by external factors. Based on this, what could be the most likely interpretation of the trend observed in the time-series data?
A: A steady increase indicates a long-term positive external influence.
B: Frequent large spikes suggest random noise without any underlying trend.
C: A systematic drop followed by recovery indicates a seasonal effect.
D: Significant fluctuations followed by a general downward trend suggest an underlying systemic issue.



Ground Truth Answer: The correct answer is D: Significant fluctuations followed by a general downward trend suggest an underlying systemic issue.

(Optional Explanation): The observed significant fluctuations along with the overall downward trend indicate potential instability or decline in the underlying variable, pointing to systemic issues rather than simply seasonal effects or random noise.

PreOrcaTS Model Answer: Based on the time-series data provided, the most likely interpretation of the trend observed is: D: Significant fluctuations followed by a general downward trend suggest an underlying systemic issue.

This interpretation is supported by the presence of multiple peaks and troughs in the data, indicating variability that could be influenced by external factors. The overall downward trend suggests that despite these fluctuations, there is a persistent decline in the variable being measured, which could indicate a systemic issue that needs to be addressed.

OrcaTS Model Answer: Based on the time-series data provided, the most likely interpretation of the trend observed is: D: Significant fluctuations followed by a general downward trend suggest an underlying systemic issue.

This interpretation is supported by the presence of multiple peaks and troughs, indicating variability in the data, followed by a gradual decline. This pattern could suggest that external factors are influencing the variable, leading to a systemic issue that needs to be addressed.

TS Model Answer: Based on the analysis of the time-series data, I classify the trend as representing: A steady increase indicates a long-term positive external influence.

This suggests that the variable is consistently rising, possibly due to factors like increased demand or improved conditions.

PreTS Model Answer: Based on the time-series data, I classify the trend as representing: A steady increase indicates a long-term positive external influence.

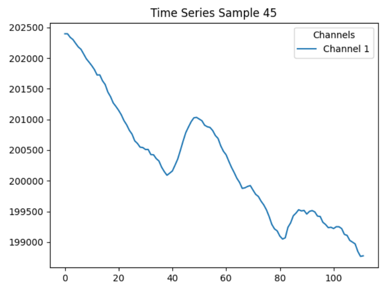
This could be due to consistent growth in demand or an improvement in the underlying process being measured. The gradual rise suggests stability and potential for further growth, which is a positive sign for the variable in question.

LLama 3.1-8B Model Answer: A: A steady increase indicates a long-term positive external influence.

The given time-series data shows a general upward trend over time, with some fluctuations. The data points are mostly increasing, and there are no significant drops or spikes that would indicate a seasonal or random effect. This suggests that there is a long-term positive external influence at play, which is driving the steady increase in the variable being measured. This could be due to various factors such as a growing market, increasing demand, or a positive economic trend.

Table 14: Example failure case for multi-modal LLMs in interpreting time-series trends. Variability in model interpretations demonstrates potential weaknesses in trend analysis and the risk of hallucinating conclusions from the same dataset.

Prompt: Compute the rate of change between the first and last values in the dataset.



Ground Truth Answer: Of course! To compute the rate of change, we'll use the formula:

$$\text{Rate of Change} = \frac{\Delta y}{\Delta x} = \frac{y_{\text{last}} - y_{\text{first}}}{x_{\text{last}} - x_{\text{first}}}$$

From the data, let's assume:

- First value at $x = 0$ (say 202000)
- Last value at $x = 112$ (say 199000)

Substituting:

$$\frac{199000 - 202000}{112 - 0} = \frac{-3000}{112} \approx -26.79$$

So, the rate of change is approximately **-26.79 per time unit**, indicating a decrease over time.

PreOrcaTS Model Answer: Sure! To compute the rate of change between the first and last values in the dataset, we can use the formula: Rate of Change $= \frac{y_{\text{last}} - y_{\text{first}}}{x_{\text{last}} - x_{\text{first}}}$. From the time-series data you provided, let's assume the following values:

- $y_{\text{first}} = 100$ (value at the first time point)
- $y_{\text{last}} = 200$ (value at the last time point)
- $x_{\text{first}} = 0$ (first time point)
- $x_{\text{last}} = 44$ (last time point)

Now, substituting these values into the formula: Rate of Change $= \frac{200-100}{44-0} = \frac{100}{44} \approx 2.27$. This means that the rate of change in the dataset is approximately **2.27 units per time point**. This indicates a steady increase in the values over the observed period.

LLama 3.1-8B Model Answer: To compute the rate of change between the first and last values in the dataset, we need to follow these steps:

1. Identify the first and last values in the dataset.
2. Calculate the difference between the first and last values.
3. Divide the difference by the number of time periods (or the number of data points) to get the average rate of change.

Let's identify the first and last values in the dataset:

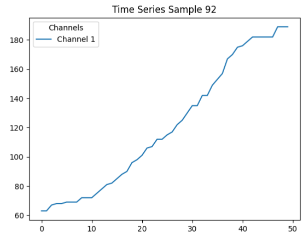
- The first value is: 1.052
- The last value is: 0.2404

Now, let's calculate the difference between the first and last values: Difference $= 0.2404 - 1.052 = -0.8116$. Since we are calculating the rate of change, we will use the absolute value of the difference: $|\text{Difference}| = 0.8116$. Now, calculate the average rate of change over 101 data points: Rate of Change $= \frac{0.8116}{100} = 0.008116$. So, the average rate of change between the first and last values is approximately **0.008116**. When expressed as a percentage: Percentage Rate of Change $= 0.008116 \times 100 = 0.81\%$. Therefore, the rate of change between the first and last values is approximately **0.81% per time period**.

Table 15: Example failure case for multi-modal LLM's in mathematical reasoning. Computing values over the real-life values becomes difficult since the models rely on normalization of the time-series before tokenization. While the process may be generally correct, the models will often make up or hallucinate values causing the actual answers to be inaccurate.

Prompt: Hi there! I'm managing a water supply project, and I've been tracking our reservoir levels over the past few months. Currently, I'm debating whether to increase our water extraction rate now to meet growing demand or to maintain our current extraction rate to ensure sustainability. Considering the recent water levels, what do you recommend?

Ground Truth Answer: After analyzing the reservoir data, it's evident that the water levels have shown a consistent upward trend over the last several weeks, reaching levels higher than we've seen in recent months. This suggests that there's ample supply available. However, while the increase looks promising, I recommend that you proceed with caution and slightly increase the extraction rate, rather than a full-scale jump. This way, you can meet the immediate demand without risking long-term sustainability. Balancing rapid extraction with conservation is key here!



PreOrcaTS Model Answer: Based on the time-series data of your reservoir levels, I've observed a significant upward trend over the past few months, indicating a steady increase in water availability. However, there are some fluctuations in the data, suggesting that while the overall trend is positive, there may be periods of volatility. Given the current demand and the rising levels, I recommend increasing your water extraction rate now. This proactive approach will help meet the growing demand while also ensuring that you're making the most of the available resources. However, it's essential to monitor the levels closely to adjust your extraction rate if the trend shifts or if any unexpected changes occur.

OrcaTS Model Answer: Based on the time-series data you've provided, I've observed a significant upward trend in reservoir levels over the past few months, particularly in the last few weeks. This indicates that the current extraction rate is not causing a depletion of the reservoir, and there's ample water available to meet the growing demand. Therefore, I recommend increasing your water extraction rate now to capitalize on the current surplus and ensure that you can meet the anticipated demand without compromising sustainability in the future.

PreTS Model Answer: Looking at the time-series data, there's a noticeable upward trend in the reservoir levels over the past few weeks, indicating a recovery from previous lows. However, there are also some fluctuations that suggest potential volatility. Given this context, I recommend increasing the extraction rate now. The rising levels indicate that the reservoir is replenishing, and meeting the growing demand could help manage the supply effectively before any potential downturns occur. It's a strategic move to balance current needs with sustainability.

LLama 3.1-8B Model Answer: To provide a recommendation, I'll analyze the given time-series data representing the reservoir levels over the past few months. The data is presented as a sequence of numbers, where each number represents the reservoir level at a specific point in time.
Upon examining the data, I notice the following trends:

- **Increasing trend**: The reservoir levels have been increasing over time, with some fluctuations. This suggests that the current extraction rate is not depleting the reservoir at an alarming rate.
- **Seasonal fluctuations**: There are noticeable seasonal fluctuations in the reservoir levels, with higher levels during the summer months (e.g., July, August) and lower levels during the winter months (e.g., January, February).
- **Recent increase**: The reservoir levels have been increasing rapidly over the past few months, with a significant jump in the last few data points (e.g., 9.7, 9.7, 9.7, 9.7, 9.7, 10.19, 10.19, 10.19).

Considering the increasing trend and recent increase in reservoir levels, I recommend **increasing the water extraction rate** to meet the growing demand. This decision is based on the following reasoning:

- **Current sustainability**: The reservoir levels are not critically low, and the current extraction rate is not depleting the reservoir at an alarming rate.
- **Growing demand**: The increasing trend in reservoir levels suggests that the current extraction rate is not sufficient to meet the growing demand.
- **Opportunistic increase**: With the recent increase in reservoir levels, it is an opportune time to increase the extraction rate to meet the growing demand without compromising the sustainability of the reservoir.

To ensure the sustainability of the reservoir, it is essential to monitor the reservoir levels closely and adjust the extraction rate as needed. This may involve:

- Regular monitoring of reservoir levels to ensure they remain within a sustainable range.
- Adjusting extraction rates based on the reservoir levels, accounting for growing demand and seasonal fluctuations.
- Implementing conservation measures, such as reducing non-essential water usage or implementing water-saving technologies, to reduce the demand on the reservoir.

Table 16: Example of model outputs for reservoir level analysis. Models provide varying recommendations based on the same time-series data, highlighting their reasoning and suggestions for water extraction rate management.