TimeSeriesExam: A Time Series Understanding Exam

Yifu Cai, Arjun Choudhry,[*] Mononito Goswami,[*] Artur Dubrawski
Auton Lab, Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
{yifuc, arjuncho, mgoswami, awd}@cs.cmu.edu

Abstract

Large Language Models (LLMs) have recently demonstrated a remarkable ability to model time series data. These capabilities can be partly explained if LLMs understand basic time series concepts. However, our knowledge of what these models understand about time series data remains relatively limited. To address this gap, we introduce TimeSeriesExam, a configurable and scalable multiple-choice question exam designed to assess LLMs across five core time series understanding categories: *pattern recognition*, *noise understanding*, *similarity analysis*, *anomaly detection*, and *causality analysis*. TimeSeriesExam comprises of over 700 questions, procedurally generated using 104 carefully curated templates and iteratively refined to balance difficulty and their ability to discriminate good from bad models. We test 7 state-of-the-art LLMs on the TimeSeriesExam and provide the first comprehensive evaluation of their time series understanding abilities. Our results suggest that closed-source models such as GPT-4 and Gemini understand simple time series concepts significantly better than their open-source counterparts, while all models struggle with complex concepts such as causality analysis. We believe that the ability to programatically generate questions is fundamental to assessing and improving LLM's ability to understand and reason about time series data. TimeSeriesExam is available on https://huggingface.co/datasets/AutonLab/TimeSeriesExam1.

1 Introduction

There has been a lot of interest in exploring connections between Large Language Models (LLMs) and time series modeling. Most recent time series foundation models, for instance, leverage LLM architectures and are pre-trained on copious amounts of time series data [1, 2, 3, 4, 5, 6]. Meanwhile, other studies have shown that when effectively "reprogrammed", LLMs can directly excel at various time series tasks including forecasting [7, 8, 9], anomaly detection, imputation, and classification [10]. But perhaps the most unexpected result was that LLMs can be zero-shot forecasters [11]. These surprising results raise a fundamental question: *do LLMs understand basic time series concepts?*

Existing attempts to answer this question have relied on domain-specific benchmarks [12, 13] or synthetic datasets generated using LLMs themselves [14]. However, performance on these benchmarks is not a perfect measure of an LLM's understanding of time series data. This is because they are confounded by the additional domain knowledge required to answer domain-specific questions (e.g., understanding of what the *arrhythmia* is in the context of ECG data). Furthermore, synthetic time series generated using LLMs may not be entirely accurate, as their correctness depends on the very ability we aim to evaluate. Finally, these benchmarks are static, offering little to no control over the qualitative properties (e.g., difficulty) of their inquiries.

[*] Equal contribution.

NeurIPS 2024 Workshop on Time Series in the Age of Large Models.

Category	Subcategory	Example question
Pattern Recognition	Trend	What is the most likely linear trend coefficient of the given time series?
	Cyclic	The given time series has sine wave pattern. How does its amplitude change from the beginning to the end?
	Stationarity	Is the given time series likely to be stationary after removing the cycle component?
	Regime Switching	Based on the given time series, how many different regimes are there?
	Statistical properties	Is the mean stable over time in the given time series?
	Random processes	Does the following time series exhibit a mean reversion property?
Noise Understanding	White Noise	Is the given time series a white noise process?
	Random Walk	Is the given time series likely to be a random walk process?
	Signal / Noise Ratio	You are given two time series with the same underlying pattern but different noise level. Which time series has higher magnitude of noise?
Anomaly Detection		The following time series has two types of anomalies appearing at different time points. What are the likely types of these anomalies?
Comparative Analysis	Shape	Despite the noise, do the given two time series have similar patterns?
	Distributional	You are given two time series which are generated using a random walk. Are they likely to have the same variance?
Causality Analysis	Granger Causality	Is there Granger causality between the two time series?

Table 1: Example template questions for different reasoning tasks. Each subcategory covers a specific aspect of time series understanding, guiding the model to reason about comparative, anomalies, and causal relationships.

To bridge this gap, we introduce TimeSeriesExam, a scalable and configurable multiple-choice question exam to assess LLMs across five core time series understanding categories. TimeSeriesExam comprises of over 700 questions, procedurally generated using carefully designed templates, and refined using Item Response Theory (IRT) [15, 16] to ensure each question has an appropriate level of difficulty and effectively differentiates candidate LLMs with varying abilities.

We evaluate 7 state-of-the-art open and closed-source LLMs on the TimeSeriesExam and provide the first comprehensive evaluation of their ability to understand basic time series concepts. Our findings reveal a decisive capability gap between closed and open-source models in understanding simple time series concepts. For brevity, we defer a detailed discussion of related work to App. A.

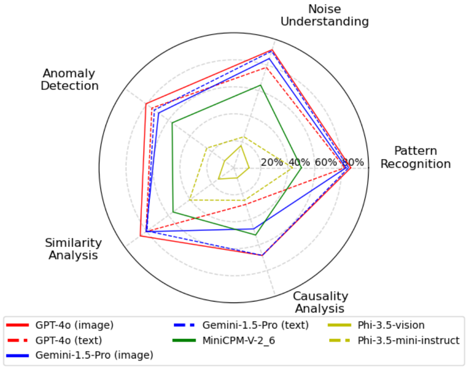
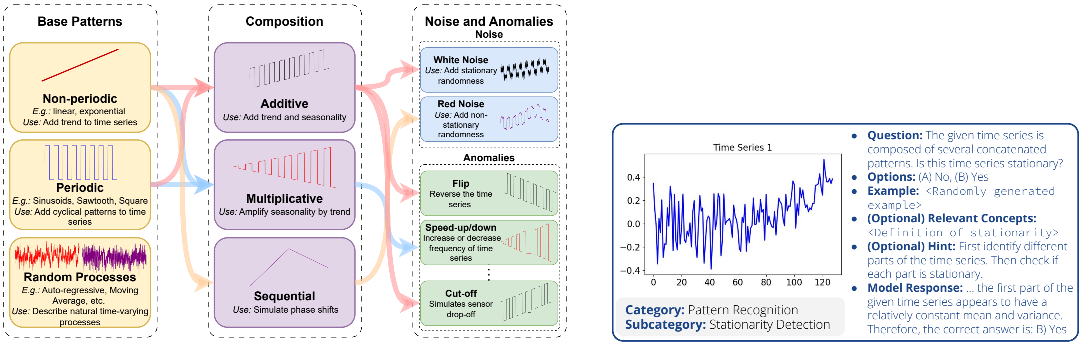


Figure 1: Accuracy of latest LLMs on the TimeSeriesExam. Closed-source LLMs outperform open-source ones in simple understanding tasks, but most models struggle with complex reasoning tasks.

2 TimeSeriesExam: A Scalable and Configurable Time Series Exam

Composition. The TimeSeriesExam systematically assesses whether LLMs **understand** basic time series patterns such as trends and seasonality (*pattern recognition*), the concept of noise and other time series concepts in the presence of noise (*noise understanding*). It also evaluates LLMs on three different **reasoning** tasks: identifying abrupt deviation from “normal" behavior [17] (*anomaly detection*), comparing and contrasting statistical properties of 2 time series (*comparative reasoning*), reasoning about causality, specifically Granger Causality [18] (*causality*). We expect these five categories will present increasing levels of difficulty for LLMs, particularly the reasoning tasks, which typically involve multiple time series and require a grasp of basic time series concepts. As shown in Tab. 1, each category is further divided into sub-categories that represent more specific concepts within the broader category.

Question Templates. The TimeSeriesExam comprises over 100 unique templates, carefully curated in collaboration with time series experts and cross-verified for accuracy, that can be used to generate any number of random questions. Each template (Fig. 2)(*Right*) evaluates a specific (sub-)category (e.g., *pattern recognition*), and comprises of a question (e.g., ‘‘Is this time series


Base Patterns
Non-periodic
E.g.: linear, exponential
Use: Add trend to time series
Periodic
E.g.: Sinusoids, Sawtooth, Square
Use: Add cyclical patterns to time series
Random Processes
E.g.: Auto-regressive, Moving Average, etc.
Use: Describe natural time-varying processes
Composition
Additive
Use: Add trend and seasonality
Multiplicative
Use: Amplify seasonality by trend
Sequential
Use: Simulate phase shifts
Noise and Anomalies
Noise
White Noise
Use: Add stationary randomness
Red Noise
Use: Add non-stationary randomness
Anomalies
Flip
Reverse the time series
Speed-up/down
Increase or decrease frequency of time series
Cut-off
Simulates sensor drop-off
Time Series 1
0.4
0.2
0.0
−0.2
−0.4
0
20
40
60
80
100
120

Category: Pattern Recognition
Subcategory: Stationarity Detection

- **Question:** The given time series is composed of several concatenated patterns. Is this time series stationary?
- **Options:** (A) No, (B) Yes
- **Example:** `<Randomly generated example>`
- **(Optional) Relevant Concepts:** `<Definition of stationarity>`
- **(Optional) Hint:** First identify different parts of the time series. Then check if each part is stationary.
- **Model Response:** ... the first part of the given time series appears to have a relatively constant mean and variance. Therefore, the correct answer is: B) Yes

Figure 2: (*Left*) **Time Series Curation Pipeline:** The composition model generates controlled synthetic time series step-by-step. The pipeline enables diversity by combining different components to create numerous synthetic time series with varying properties. (*Right*) Each template evaluates a specific category, and includes a question, list of options, example question and answer pair for in-context learning, and optionally a hint and descriptions of complicated technical terms. Here, `GPT-4o` showcases its ability to transfer visual understanding and time series concepts into effective reasoning.

`stationary?"`), a list of options (e.g., `‘‘(A) Yes, (B) No"`), and an example question and answer pair for in-context learning. Each template comes with a *hint* which breaks down complex questions into simpler steps and textual descriptions of complicated technical terms. By incorporating these relevant concepts, we can isolate an LLM's ability to understand time series concepts (e.g., whether the mean and variance remain constant) from its understanding of complex technical jargon (e.g., stationarity). Each option (e.g. `‘‘(A) Yes"`) is linked to a synthetic time series generator (Fig. **??**) that produces a random time series as if the current option were true (e.g., a random stationary time series). This allows us to generate random but accurate time series at scale.

Generating Questions. We generate different questions from the same template by systematically varying the correct option and producing synthetic time series conditioned on the template and the correct option pair. Our simple and scalable approach, illustrated in Fig. 2(*Left*), involves sampling a small number of base patterns from a predefined pool and combining them using a composition function. Base patterns can be periodic (e.g., sine function), non-periodic (e.g., linear increasing function), or random time-varying processes (e.g., AR process). Depending on the template's nature, the final step adds additive noise or anomalies using the anomaly injection process described in [17].

Improving Questions Iteratively. We use Item Response Theory (IRT) [19] to achieve finer grained control over the quality of randomly generated questions included in the `TimeSeriesExam`. IRT is a statistical framework that models the relationship between an individual's (or LLM's) latent trait (e.g., knowledge, ability) and their responses to a set of items (e.g., questions on a test). It is a valuable tool in exam development as it helps to identify weak exam items, ensures consistent scoring across different versions of the exam, and also allows tailoring the testing experience to the LLM's abilities.

Our primary objective is to design a `TimeSeriesExam` where each question can maximally distinguish the abilities of candidate LLMs. We use the two-parameter logistic (2PL) model for this. Formally, for LLM j with ability θ_j, and question i with difficulty b_i, discrimination ability a_i, the 2PL model defines the probability of a correct response as: $\mathbb{P}(r_{ij} = 1 | a_i, b_i, \theta_j) = 1/(1 + e^{-a_i(\theta_j - b_i)})$

Each `TimeSeriesExam` typically undergoes 1–3 rounds of iterative refinement. In each round, all candidate models take the exam. Based on their responses, we fit the parameters of Equation 2 using maximum likelihood estimation (MLE). Then, we drop $X\%$ of samples with the lowest sum of difficulty and discrimination ability. Finally, we randomly re-generate questions from the dropped templates. This iterative process is detailed in Algorithm 1 and the hyper-parameters of the fitting process are provided in App. C.2.

3 Experiments and Results

Experimental Setup. We evaluate LLMs on TimeSeriesExam using two different setups for feeding time series into language models: (1) Image, where time series are plotted and input as images; and (2) Text, where time series values are truncated to one decimal place and separated by commas. We evaluate two proprietary models, GPT-4o [20] and Gemini [21]. For open source models, we chose Phi3.5 [22], and MiniCPM [23]. We selected these models to study: (1) the impact of model size on time series understanding, and (2) the effect of time series tokenization on model performance. Previous studies, such as LLMTime [11], have explored tokenizing time series as text. We chose not to scale the time series, as their magnitudes are intentionally small to conserve tokens, and scaling could distort the shape, which is critical for many of the questions. The evaluations are done with one shot setting. The experiment details are provided in App. C.

Impact of Model Size and Time Series Tokenization on Reasoning Fig. 1 reveals two key findings: first, closed-source models like GPT-4o and Gemini outperform open-source models in basic tasks such as pattern recognition but all models struggle with more complex tasks like causality analysis. This performance gap can likely be attributed to differences in pretraining data and model size, as seen in the comparison between MiniCPM (7B parameters) and Phi-3.5 (3.8B parameters). Second, tokenizing time series data as images generally produces better results than textual tokenization. We offered an example response from GPT-4o using both tokenization method in Fig. C.7. Our qualitative analysis suggests that this outcome is likely due to the text-based approach causing models to focus excessively on details. This finding suggests that tokenization strategy is a critical factor in advancing reasoning capabilities, and they point to the potential for multimodal models that integrate time series and text data for more robust interactive reasoning.

Iterative Improved Benchmark We observed an increasing trend for the sample average discrimination parameter(a_i in Equation 2), which reflects the model's capacity to distinguish between individuals with varying ability levels. As the discrimination parameter increases, the model's ability to differentiate improves. The figure for the same can be found in App. C.3.

Effect of Guidance on Model Reasoning Tab. 2 shows the performance of Gemini-1.5-Pro with various dataset elements provided. Question hints, which offer the first step in the reasoning process, improved the model's performance, suggesting that sometimes model could struggle to form a coherent, logical sequence to reach the correct answer. Interestingly, providing relevant concepts hindered performance, possibly due to confusion or discrepancies with the model's pretraining data.

Question & Guidance	Accuracy
Query only	79.16
Query + Hint	**79.29**
Query + Relevant Concepts	75.88
Query + Hint + Relevant Concepts	77.71

Table 2: Gemini-1.5-Pro Performance with Different Dataset Components: The results indicate that adding relevant concepts, alongside hints, can sometimes decrease the model's performance. This may occur because the introduced concepts either contradict the model's internal knowledge or misguide its reasoning, leading to an incorrect chain of thought.

4 Conclusion

In this work, we introduced a controlled, salable time series benchmark. We demonstrated that proprietary models like Gemini and GPT-4 achieved non-trivial performance when time series were provided as both images and text. However, all models still struggle with complex reasoning tasks requiring multiple time series and multi-step inference. This highlights some key future directions:

Developing a Benchmark for Practical and Complex Reasoning Tasks While this work emphasizes reasoning based on time series understanding of patterns, future benchmarks should address more advanced tasks including complex causality analysis and context-driven forecasting.

Designing a More Rigorous Exam We refer to our benchmark as an examination due to its structured components that scientifically assess a range of abilities. Future benchmarks should adopt more rigorous designs that query specific knowledge. Drawing from concepts such as knowledge tracing, we can introduce purposefully designed detractors to evaluate model performance better.

5 Acknowledgment

This work was partially supported by the NSF award 2406231, NIH (R01NS124642 and R01DK131586), and the US Army (W911NF-20-D0002).

References

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

[2] Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*, 2024.

[3] Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Arian Khorasani, George Adamopoulos, Rishika Bhagwatkar, Marin Biloš, Hena Ghonia, Nadhir Vincent Hassen, Anderson Schneider, et al. Lag-llama: Towards foundation models for time series forecasting. *arXiv preprint arXiv:2310.08278*, 2023.

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

[5] Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Generative pre-trained transformers are large time series models. In *Forty-first International Conference on Machine Learning*, 2024.

[6] Azul Garza and Max Mergenthaler-Canseco. Timegpt-1. *arXiv preprint arXiv:2310.03589*, 2023.

[7] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.

[8] Defu Cao, Furong Jia, Sercan O Arik, Tomas Pfister, Yixiang Zheng, Wen Ye, and Yan Liu. Tempo: Prompt-based generative pre-trained transformer for time series forecasting. *arXiv preprint arXiv:2310.04948*, 2023.

[9] Ching Chang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Two-stage fine-tuning for time-series forecasting with pre-trained llms. *arXiv preprint arXiv:2308.08469*, 2023.

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

[11] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems*, 36, 2024.

[12] Jungwoo Oh, Gyubok Lee, Seongsu Bae, Joon-myoung Kwon, and Edward Choi. Ecg-qa: A comprehensive question answering dataset combined with electrocardiogram. *Advances in Neural Information Processing Systems*, 36, 2024.

[13] Tianwei Xing, Luis Garcia, Federico Cerutti, Lance Kaplan, Alun Preece, and Mani Srivastava. Deepsqa: Understanding sensor data via question answering. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, pages 106–118, 2021.

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

[15] Susan E Embretson and Steven P Reise. *Item response theory*. Psychology Press, 2013.

[16] Gauthier Guinet, Behrooz Omidvar-Tehrani, Anoop Deoras, and Laurent Callot. Automated evaluation of retrieval-augmented language models with task-specific exam generation. *arXiv preprint arXiv:2405.13622*, 2024.

[17] Mononito Goswami, Cristian Challu, Laurent Callot, Lenon Minorics, and Andrey Kan. Unsupervised model selection for time-series anomaly detection. *arXiv preprint arXiv:2210.01078*, 2022.

[18] Clive WJ Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: journal of the Econometric Society*, pages 424–438, 1969.

[19] Frederic M Lord and Melvin R Novick. *Statistical theories of mental test scores*. IAP, 2008.

[20] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[21] Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[22] Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

[23] Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*, 2024.

[24] Aya Abdelsalam Ismail, Mohamed Gunady, Hector Corrada Bravo, and Soheil Feizi. Benchmarking deep learning interpretability in time series predictions. *Advances in neural information processing systems*, 33:6441–6452, 2020.

[25] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Cost: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. *arXiv preprint arXiv:2202.01575*, 2022.

[26] Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. *arXiv preprint arXiv:2310.10688*, 2023.

[27] Sabera Talukder, Yisong Yue, and Georgia Gkioxari. Totem: Tokenized time series embeddings for general time series analysis, 2024.

[28] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115, 2021.

[29] John Patrick Lalor and Pedro Rodriguez. py-irt: A scalable item response theory library for python. *INFORMS Journal on Computing*, 35(1):5–13, 2023.

A Related Works

Time Series Reasoning Benchmark. There exist a few benchmarks for time series reasoning, including a recent work that categorized time series reasoning into three primary tasks: *context-aided forecasting*, *question answering*, and *etiological reasoning* [14]. However, the work was limited by the fact that all samples were generated using GPT, where scientific design and correct time series correspondence are not guaranteed. Other efforts towards building time series reasoning benchmarks were exclusively for domain-specific tasks, such as those defined in `ECGQA` [12]. Thus, no existing benchmark currently evaluates whether LLMs possess an innate understanding of time series concepts and can transfer that understanding into structured reasoning. Our work bridges this gap by proposing a synthetically generated controlled dataset for this evaluation.

Synthetic Time Series Generation. The generation of synthetic time series with controlled behaviors, such as trends and cyclic patterns, is fundamental for constructing accurate reasoning benchmarks. A common approach involves sampling from diverse random processes [24], such as Autoregressive Processes, which offer variability but lack control over specific patterns like cyclic behavior. To address this, [25] proposed a decomposition-based method, generating desired patterns by incorporating cyclic components into an additive model on top of random processes. We build upon both works by having a more diverse set of random processes and patterns, incorporating not only additive composition methods but also multiplicative and other forms of composition.

Time Series Foundation Models. With the advent of several time series foundation models [2, 1, 3, 8, 4, 26, 5, 6, 27] in recent months, there has been a paradigm-shifting change in the ability of models to perform time series analysis (TSA) tasks like forecasting, classification, imputation, and anomaly detection, including in zero-shot settings. These models apply language model architectures pre-trained on time series data to time series analysis tasks, achieving state-of-the-art performance compared to architectures built exclusively for time series analysis like `Informer` [28]. While their performance benefits for these tasks are noticeable, understanding of their reasoning ability and capability of identifying the nuances of a time series are yet to be evaluated. This is exacerbated by the issue that their outputs are typically time series, which is difficult for users to understand, rather than a response explaining the time series and characteristics that lead to the given output. LLMs' ability to reason is important here, since they generate responses that users can easily understand.

B Dataset Details

Tab. 3 presents the meta information for each category, while Tab. 4 outlines the components of time series synthesis. The dataset includes 11 unique base patterns, 3 composition methods, 10 transformations, and 2 paired time series creation methods. These combinations produce a diverse set of time series with controlled features such as trend and seasonality.

Meta-information Type	Anomaly Detection	Similarity Analysis	Noise Understanding	Pattern Recognition	Causality Analysis
# Questions	129	113	87	371	63
%age questions with 2 time series	31.01	100	14.94	3.77	100

Table 3: TimeSeriesExam meta-information breakdown for each category. Each question is associated with a time series of length 128 time steps, and an example time series of length 64 time steps.

Object Name	Description
Base Time Series Objects	
Linear Trend	Create a linear line following $y = at$
Exponential Trend	Create an exponential line following $y = e^{at}$
Log Trend	Create a log trend following $y = \log t$
Constant	Create a constant line following $y = a$
Gaussian White Noise	Create an independent draw from Gaussian each time step
Red Noise (Random Walk)	Create a random walk where noise is drawn from normal, $y_t = y_{t-1} + \epsilon_t$
SineWave	Generate a smooth, periodic oscillation following the function $y = A\sin(t)$
Sawtooth Wave	Create a periodic waveform with a linear rise and abrupt fall
SquareWave	Generate a periodic waveform that alternates between two levels, with equal durations
Moving Average Process	Create a moving average MA(q) random process, $y_t = \sum_{i=1}^{q} \epsilon_{t-i}\alpha_i + \epsilon_t$
Autoregressive Process	Create an autoregressive AR(p) random process, $y_t = \sum_{i=1}^{p} y_{t-i}\alpha_i + \epsilon_t$
Composition Method	
Additive	Add each time series object at each time stamp
Multiplicative	Multiply each time series object at each time stamp
Concatenate	Concatenate each time series object along time series horizon
Transformation	
Anomaly Creation	Following the nine synthetic transformations defined in [17]
Sign Flip	$y_t = -y_t$
Paired Time Series Creation	
Lagged Pair Creation	Create a lagged time series from a baseline time series
Granger Pair Creation	Create a granger caused time series from a baseline time series

Table 4: Time Series in the `TimeSeriesExam`are created from a combination of diverse baseline Time Series Objects. The baseline objects cover linear/non-linear signals and cyclic patterns. For trend variables, t is a sequence of integers that represents time. ϵ_t is a standard normal random variable. For MA(q) and AR(p) processes, α_i represents its parameter.

B.1 Iterative Refinement Algorithm

Algorithm 1 Iterative Dataset Refinement with IRT and Resampling

```
Require: num_iterations = 3, drop_percentage = 0.2, initial dataset $D_0$
$D \leftarrow D_0$
for iteration $= 1$ to num_iterations do
    Evaluate each candidate $i$ on $D$, and obtain the response set $R = \{r_{ij} \mid r_{ij} = 1 \text{ if candidate } i \text{ correctly answers question } j\}$
    Fit the IRT model to obtain the discrimination parameters $\mathbf{A} = \{a_j \mid j \in \text{Questions}\}$ and difficiulty parameter $\mathbf{B} = \{b_j \mid j \in \text{Questions}\}$
    Normalize set $\mathbf{A}$ and $\mathbf{B}$ between 0 and 1, and calculate score $\mathbf{S} = \{b_j + a_j \mid j \in \text{Questions}\}$
    Find $\mathbf{S}'$ which is the score for samples that are answered correctly by the best model in the round
    Find the index set $I = \{j \mid a_j < \text{Quantile}(\mathbf{S}', \text{drop\_percentage})\}$, where $a_j$ is less than the drop_percentage quantile of $\mathbf{A}$
    for each $j \in I$ do
        Resample a new question $q'$ from the same category as question $j$
        Set $D[j] \leftarrow q'$
    end for
end for
return $D$
```

C Experiment Details

C.1 Generation Configuration

We set the maximum token length to 1024 and a temperature of 0.0 for generation. For models that support seed control, we use a seed value of 42; otherwise, seed control is unavailable in some proprietary models [2]

[2] The evaluation code is available at `https://anonymous.4open.science/r/TimeSeriesExam-8387`

C.2 IRT Model Parameters

The IRT models are fitted using library *py-irt* [29]. The parameters are epochs=2000, lr=0.1, lrdecay=0.9999, dropout=0.5, hidden=100

C.3 Average Sample Discrimination Parameter over Rounds

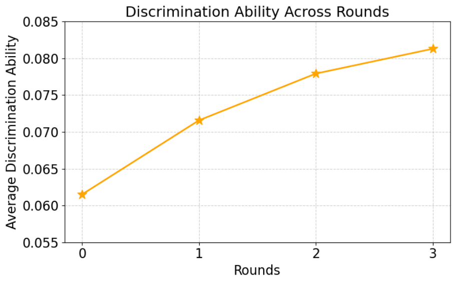


Figure 3: The sample average discrimination parameter across rounds shows an upward trend, indicating an improved ability of the questions to differentiate candidates with varying levels of ability.

C.4 Dropped dataset distribution per round

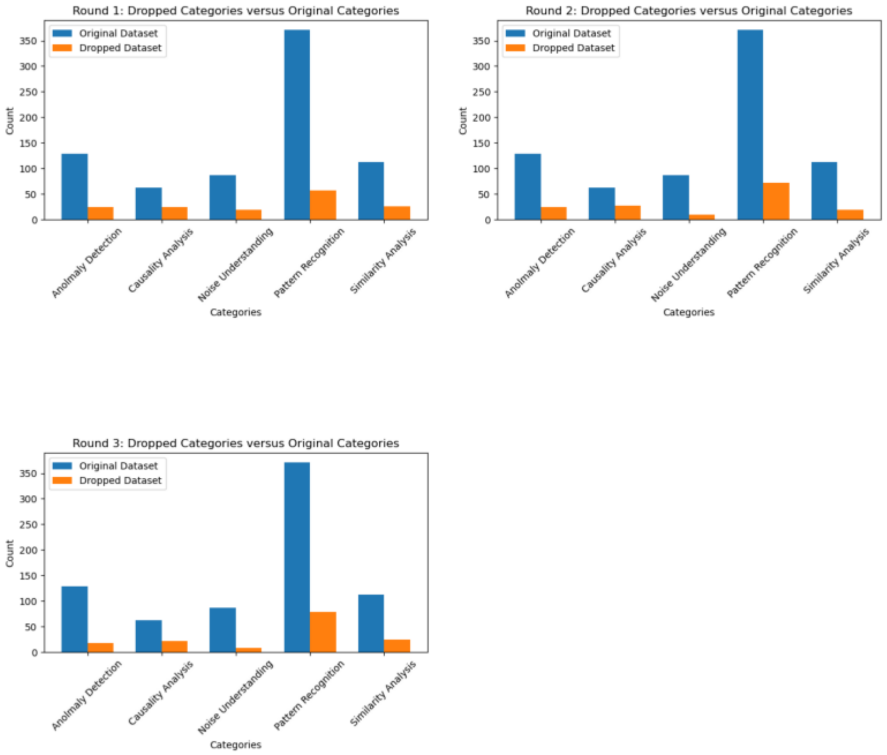


Figure 4: Dropped Dataset Distribution per round. Dropped category distribution per round generally mirrors the overall category distribution.

We can observe in Fig. 4 that the proportion of dropped questions for each category is approximately uniform.

Model	Image Tokenization Cost ($)	Text Tokenization Cost ($)
GPT-4o	0.009703598	0.008769533
Gemini	0.006792518	0.006138673

Table 5: Inference cost per sample (in USD) based on tokenization methods. Costs for both image and text tokenization are similar and cost-effective. Token prices were sourced from official model documentation.

C.5 Inference Cost

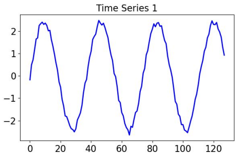
We report inference cost per sample for proprietary models in 5. The average token size per sample for image tokenization is 1940.72, the average token size per sample for text tokenization is 1753.91. Number of tokens are calculated based on GPT4 tokenizer.

C.6 Model Accuracy per Round with Category Break down

Model	Pattern Recognition	Noise Understanding	Anomaly Detection	Similarity Analysis	Causality Analysis
Round 0					
Phi-3.5 (Image)	0.20	0.34	0.11	0.19	0.22
Phi-3.5 (Text)	0.46	0.25	0.27	0.45	0.52
Gemini-1.5-Pro (Image)	0.81	0.82	0.64	0.78	0.59
Gemini-1.5-Pro (Text)	0.75	0.85	0.62	0.65	0.52
GPT-4o (Image)	**0.88**	**0.92**	**0.78**	**0.87**	**0.86**
GPT-4o (Text)	0.74	0.82	0.65	0.71	0.19
MiniCPM (Text)	0.53	0.67	0.56	0.55	0.54
Round 1					
Phi-3.5 (Image)	0.16	0.25	0.13	0.16	0.14
Phi-3.5 (Text)	0.47	0.26	0.28	0.40	0.49
Gemini-1.5-Pro (Text)	0.78	0.89	0.67	0.77	0.60
GPT-4o (Text)	0.77	0.84	0.72	0.77	0.21
GPT-4o (Image)	**0.84**	**0.91**	**0.76**	**0.87**	**0.79**
MiniCPM (Text)	0.53	0.63	0.59	0.54	0.48
Round 2					
Phi-3.5 (Image)	0.14	0.22	0.09	0.15	0.10
Phi-3.5 (Text)	0.47	0.26	0.24	0.42	0.38
Gemini-1.5-Pro (Image)	0.89	0.86	**0.74**	**0.87**	0.67
Gemini-1.5-Pro (Text)	**0.83**	**0.91**	0.72	0.82	0.67
GPT-4o (Image)	0.81	0.90	0.73	0.83	**0.68**
GPT-4o (Text)	0.80	0.83	0.73	0.81	0.25
MiniCPM (Text)	0.51	0.64	0.57	0.53	0.54
Round 3					
Phi-3.5 (Image)	0.11	0.17	0.09	0.14	0.08
Phi-3.5 (Text)	0.44	0.24	0.25	0.41	0.25
Gemini-1.5-Pro (Image)	0.83	0.85	0.69	0.81	0.48
Gemini-1.5-Pro (Text)	0.85	0.91	0.73	0.80	**0.68**
GPT-4o (Image)	**0.87**	**0.92**	**0.81**	**0.86**	**0.68**
GPT-4o (Text)	0.82	0.78	0.75	0.81	0.29
MiniCPM (Text)	0.50	0.64	0.57	0.56	0.52

Table 6: Breakdown of model accuracy across multiple rounds of item response theory.

C.7 Case Study: Model response under image tokenization and text tokenization



Category: Pattern Recognition
Subcategory: Cycle Recognition

- **Question:** What is the primary cyclic pattern observed in the time series?
- **Options:** A) SquareWave, B) SineWave, C) SawtoothWave D) No Wave
- **Example:** `<Randomly generated example>`
- **(Optional) Relevant Concepts:** `<Definition of sine, sawtooth, and Square wave>`
- **(Optional) Hint:** `Check the overall shape of the time series against the definition of provided concepts`

GPT-4o (Image):

```
Based on the given time series and the provided examples, we can observe the following:\n\n1. The time series in the first image
shows a pattern where the values gradually increase to a peak and then gradually decrease to a trough, repeating this cycle
consistently.\n2. Comparing this pattern with the provided examples:\n   - Example 1 (Sine Wave) shows a smooth, continuous
oscillation between peaks and troughs.\n   - Example 2 (Square Wave) shows abrupt transitions between two fixed levels.\n   -
Example 3 (Sawtooth Wave) shows a linear increase followed by a sharp drop.\n\nThe primary cyclic pattern observed in the given
time series closely resembles the Sine Wave pattern (Example 1), as it shows smooth, continuous oscillations.\n\nGiven the
options:\n\nA) SquareWave\nB) SineWave\nC) SawtoothWave\nD) No Pattern at all\n\nThe correct answer is:\n\nB) SineWave
```

GPT-4o (Text):

```
To determine the primary cyclic pattern observed in the given time series, we need to compare the overall shape of the time
series against the definitions of the provided concepts: Sine Wave, Square Wave, and Sawtooth Wave.\n\n### Analysis of the Given
Time Series:\nThe given time series
is:\n```\n-0.2,0.5,0.7,1.2,1.6,1.7,2.3,2.3,2.4,2.3,2.4,2.3,2.0,2.0,1.6,1.3,1.0,0.6,0.3,-0.3,-0.5,-1.0,-1.3,-1.8,-1.8,-2.0,-2.2,-2
.4,-2.4,-2.5,-2.4,-2.0,-1.8,-1.6,-1.3,-0.7,-0.3,-0.2,0.4,0.8,1.1,1.6,1.8,1.9,2.2,2.5,2.4,2.3,2.4,2.2,1.9,1.7,1.3,0.9,0.7,0.1,-0.0
,-0.5,-1.1,-1.2,-1.6,-1.8,-2.1,-2.3,-2.4,-2.6,-2.3,-2.3,-2.0,-1.6,-1.6,-1.2,-0.7,-0.5,0.1,0.3,0.9,1.1,1.5,1.9,2.1,2.3,2.2,2.4,2.4
,2.2,2.2,1.9,1.6,1.4,0.9,0.5,0.3,-0.1,-0.6,-1.1,-1.3,-1.7,-1.8,-2.2,-2.2,-2.4,-2.5,-2.5,-2.3,-2.0,-1.8,-1.5,-1.1,-0.8,-0.4,0.1,0.
4,0.7,1.1,1.3,1.7,1.9,2.3,2.5,2.3,2.3,2.4,2.1,2.0,1.7,1.3,0.9\n```\n\n### Observations:\n1. **Gradual Increase and Decrease**:
The values gradually increase to a peak and then gradually decrease, which is characteristic of a sawtooth wave.\n2. **Sharp
Transitions**: There are sharp transitions after reaching the peak or trough, which is another characteristic of a sawtooth
wave.\n3. **Repetition**: The pattern repeats itself, indicating a cyclic nature.\n\n### Comparison with Provided Concepts:\n-
**Sine Wave**: This pattern oscillates smoothly between peaks and troughs. The given time series does not exhibit smooth
oscillations; instead, it has sharp transitions.\n- **Square Wave**: This pattern alternates between two fixed levels with abrupt
changes. The given time series does not alternate between two fixed levels; it has gradual increases and decreases.\n- **Sawtooth
Wave**: This pattern has a gradual increase or decrease followed by a sharp transition back to the starting level. The given time
series matches this description closely.\n\n### Conclusion:\nGiven the characteristics of the time series and the definitions of
the provided concepts, the primary cyclic pattern observed in the time series is a Sawtooth Wave.\n\n### Answer:\nC) SawtoothWave
```

Figure 5: The image-based model provided the correct answer, while the text-based model failed due to its inability to translate detailed numerical data into a specific trend shape.