Language Models Still Struggle to Zero-shot Reason about Time Series

Mike A. Merrill
University of Washington
mikeam@cs.washington.edu

Mingtian Tan
University of Virginia
wtd3gz@virginia.edu

Vinayak Gupta
University of Washignton
vinayak@cs.washington.edu

Thomas Hartvigsen
University of Virginia
hartvigsen@virginia.edu

Tim Althoff
University of Washington
althoff@cs.washington.edu

Abstract

Time series are critical for decision making in fields like finance and healthcare. Their importance has driven a recent influx of works passing time series into language models, leading to non-trivial forecasting on some datasets. But it remains unknown whether non-trivial forecasting implies that language models can reason about time series. To address this gap, we introduce a first-of-its-kind evaluation framework for time series reasoning, including formal tasks and a corresponding dataset of multi-scale time series paired with text captions across ten domains. Using these data, we probe whether language models achieve three forms of reasoning: (1) *Etiological Reasoning*—given an input time series, can the language model identify the scenario that most likely created it? (2) *Question Answering*—can a language model answer factual questions about time series? (3) *Context-Aided Forecasting*—does relevant textual context improve a language model's time series forecasts? We find that otherwise highly-capable language models demonstrate surprisingly limited time series reasoning: they score marginally above random on etiological and question answering tasks (up to 30 percentage points worse than humans) and show modest success in using context to improve forecasting. These weakness showcase that time series reasoning is an impactful, yet deeply underdeveloped direction for language model research.[1]

1 Introduction

Time series measure how systems change over time and contain information that is uncommon in language. They are a critical data modality in healthcare (Morid et al., 2023), finance (Sezer et al., 2020), agriculture (Kamilaris and Prenafeta-Boldú, 2018), economics (Nerlove et al., 2014), political science (Beck and Katz, 2011), astronomy (Benson et al., 2020), signal processing (Jagannath et al., 2021), and beyond. As the scientific community races to bring language models (LMs) to these domains, we must ensure LMs can support decisions about these sources of valuable information. If successful, LMs could perform novel tasks like citing patterns and events in time series as evidence for observations and inferences, drawing interpretable conclusions from complex dynamical systems, or learning to recognize and respond to temporal patterns.

Several recent works have shown that LMs can be used for zero-shot time series tasks, though nearly all focus on forecasting. These works typically forecast by structuring historical observations as raw text (Liu et al., 2023b; Xue and Salim, 2023; Zhang et al., 2024; Gruver et al., 2023) or images (Li et al., 2023). This is promising work, and suggests language models may someday demonstrate the same remarkable zero-shot performance that they do with text and images. But it remains unknown whether non-trivial forecasting implies that LMs can *reason* about time series, as opposed to simply generating matching temporal patterns that appear in their inputs. In fact, recent works indicate that a LM's ability to generate data does *not* imply deeper reasoning (West et al., 2024; Hessel et al., 2023).

In this work, we develop, apply, and release a framework to ultimately find that despite excitement about using LMs for time series analysis, **current language models are remarkably bad at zero-shot time series reasoning**. We propose three components of time series reasoning. First, for a LM to reason about time series it must be able to consider the etiology (the set of possible causes) of a time series through **etiological reasoning** (Figure 1(a)). Second, a successful model should excel at **question answering** and be able to address queries about time series and how they relate to one another (Figure 1(b)). Finally, time

[1]All data and code are available at https://github.com/behavioral-data/TSandLanguage

Findings of the Association for Computational Linguistics: EACL 2024, pages 3512–3533

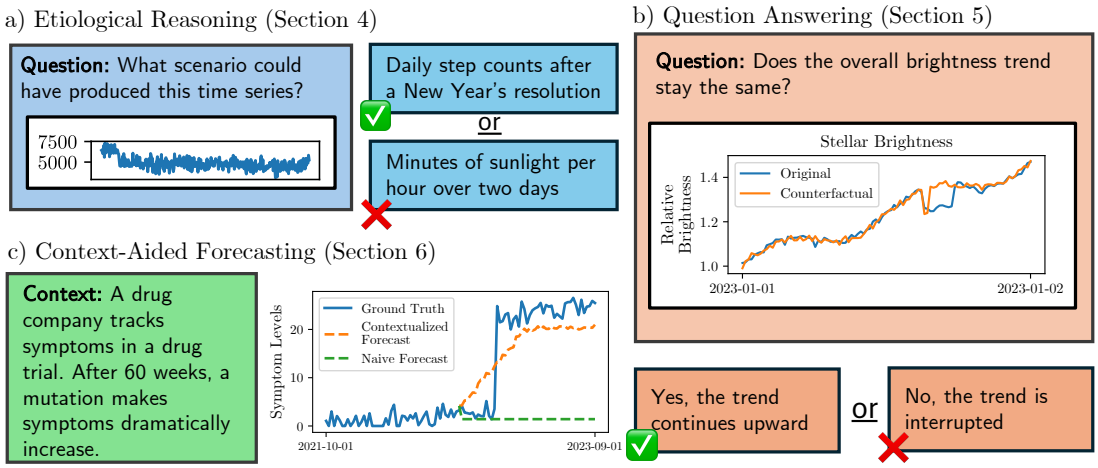


Figure 1: The three forms of time series reasoning (Section 2).

series reasoning implies **context-aided forecasting**, wherein a language model can leverage its world model and natural language context to aid in forecasting (Figure 1(c)).

To evaluate LMs we create a first-of-its kind dataset that contains 230k time series multiple choice questions and 8.7k pairs of synthetic time series and text captions that describe the series and the context in which it was observed (Section 3). These data span a diverse set of time series scenarios across including health data, transport and traffic trends, finance, and more.

We use this dataset to evaluate *etiological reasoning* by tasking models to select the most probable time series caption given the observed time series (Section 4) and find that human annotators outperform language models by a margin of up to thirty percentage points, with otherwise strong language models like GPT-4 barely doing better than random chance. Then, we test models on a *question answering* task by augmenting our dataset to include 230k question-answer pairs (Section 5). Again, we find that human annotators significantly outperform language models, indicating that language models have limited capacity to interpret the information in time series. Finally, we evaluate language models on a *context-aided forecasting* task (Section 6). We find that even with text descriptions of what will happen in future, GPT-4 struggles to incorporate this information, resulting in negligible improvements over models without additional context. Taken as a whole these results indicate that despite modest time series forecasting ability, current language models fail to reason about these ubiquitous, critical data despite considerable human performance on the same tasks.

2 Forms of Time Series Reasoning

Here we propose a rigorous (though non-exhaustive) definition of time series reasoning.

Consider a univariate uniformly-sampled time series of n observations, $x = \{x_0 \cdots x_n\}$, $x \in \mathbb{R}^n$. Suppose that an autoregressive language model M is able to represent this time series as input and produce time series observations and text as outputs. That is, M estimates the probability p of an output token sequence Y given some context tokens C and the time series: $p_M(Y|x, C) = M(Y, x, C)$.

Definition 2.1 (Etiological Reasoning)**.** Etiological reasoning is the property by which language models are able to hypothesize about the cause of a time series. That is, given a time series x, textual instructions as context C, a correct description D^+ of how x was generated and an incorrect description D^-, a language model should assign higher probability to D^+:

$$p_M(D^+|x, C) > p_M(D^-|x, C) \quad (1)$$

Language models that can reason about time series should also be able to answer questions about the behavior and implications of a time series.

Definition 2.2 (Question Answering)**.** We define question answering as a model's ability to use information in the time series x to interpret queries about the time series or the events surrounding the scenario it represents.

For the sake of evaluation, the questions should be time-series-dependent—correct answers should be unattainable without interpreting x. For example, given an ECG, a dependent question might be, "Does this signal demonstrate atrial fibrillation?" while a trivially non-dependent question would be,

"Who was the first president of the United States?" Formally, given a question Q and an answer A^+, the model should predict

$$p(A^+|x, Q) \gg p(A^+|Q) \tag{2}$$

A language model should be able to exploit this information. In a multiple-choice setting, given a correct answer A^+ and an incorrect answer A^-:

$$p_M(A^+|x, Q) > p_M(A^-|x, Q) \tag{3}$$

Finally, for an LM to reason about time series it should be able to integrate relevant information from text into forecasts about how the time series will behave in the future.

Definition 2.3 (Context-Aided Forecasting)**.** Context-aided forecasting is the property by which a language model can use additional outside information about a time series to guide its forecasts. Given the first t observations of a time series and a relevant text description D, the model should predict:

$$p_M(x_{t+1} \cdots x_n|x_0 \cdots x_t, D) > p_M(x_{t+1} \cdots x_n|x_0 \cdots x_t) \tag{4}$$

Note that D must provide some meaningful information about the behavior of x.

3 Dataset

Evaluating these forms of time series reasoning requires pairs of time series and highly-relevant text descriptions. Without a strong relationship between the two, it is impossible to determine if a model's failure to reason about time series is due to poor fundamental capabilities or a poorly-designed evaluation. However, there is no general corpus of time series and natural language descriptions that captures such relationships (Section 7.1). To address this challenge, here we contribute a first-of-its-kind dataset of synthetic multi-domain time series and highly relevant text captions.

3.1 Dataset Generation

We prompt GPT-4 to generate descriptions of environments that change over time alongside executable Python functions that generate corresponding time series. A naive solution is to generate a time series as text, however autoregressive language models struggle to generate text with long range interactions (Bubeck et al., 2023) and demonstrate poor numerical reasoning (Akhtar et al., 2023; Dziri et al., 2023). Accordingly, time series that are generated as text exhibit poor coherence and are of overall low quality (Figure 3). Instead, we leverage recent language models' capacity to generate code (Zhong and Wang, 2023; Chen et al., 2021; Wang et al., 2023b). We prompt GPT-4 to produce *data generating functions* in the form of Python scripts. We ask the model to "imagine a scenario" that would produce a time series. We then yield the following data for each scenario:

- A **caption** of the scenario that generated the time series.
- Five **characteristics** of a time series which matches this description.
- A **data generating function** which, when executed, returns the time series as an array.
- **Metadata** about the time series, including its start and end timestamp, its sample rate, units, a short caption of less than five words which summarizes the scenario.

To encourage diversity during generation, we append the latest twenty short descriptions to each new prompt and ask the model to a generate a scenario that is as distinct as possible from these previous generations. Empirically, this step is important for maintaining variety in the generated results. The full prompt is available in Section C. Finally, we filter the scenarios by removing multivariate time series and those with complex, missing, or infinite values, resulting in 8.7k scenarios. Next, we feed 100 captions into GPT-4 and ask the model to categorize these time series into ten domains (Figure A.1). We then automatically apply these categories to the remaining 8.7k scenarios (Figure A.2). We manually reviewed 50 scenarios and found no substantial inaccuracies between the captions and the time series.

3.2 Evaluating Data Quality

The quality of our synthetic data relies on the realism of the generated time series as well as the relevance of the associated text captions. To systematically quantify the quality of this relationship we recruited ten experienced PhD-level data scientists with relevant time series experience for human evaluation. The ten participants were each shown 50 unique time series line plots with four possible captions each (for a total of 500 examples). One caption was the model-generated ground truth, while the other three were randomly sampled from other, irrelevant time series. A screenshot of the annotation tool is available in Figure A.7. Annota-

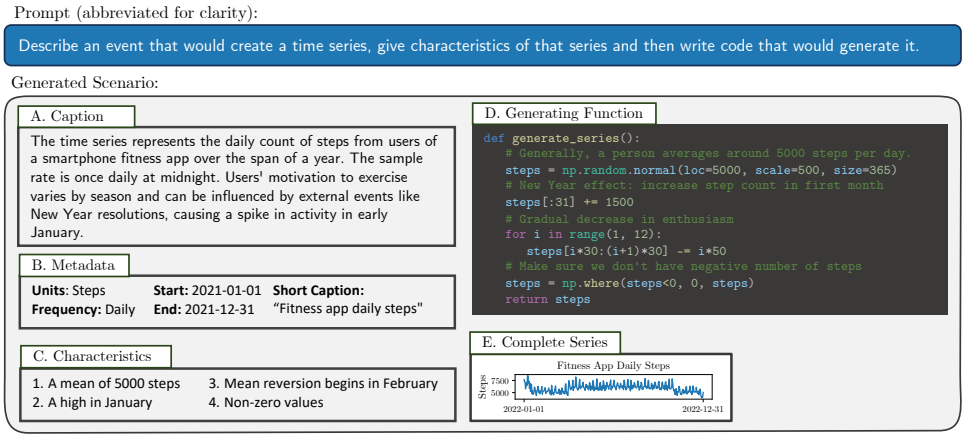


Figure 2: We generate realistic time series and text pairs by querying GPT-4 for code that can be executed to generate the signal (Section 3).

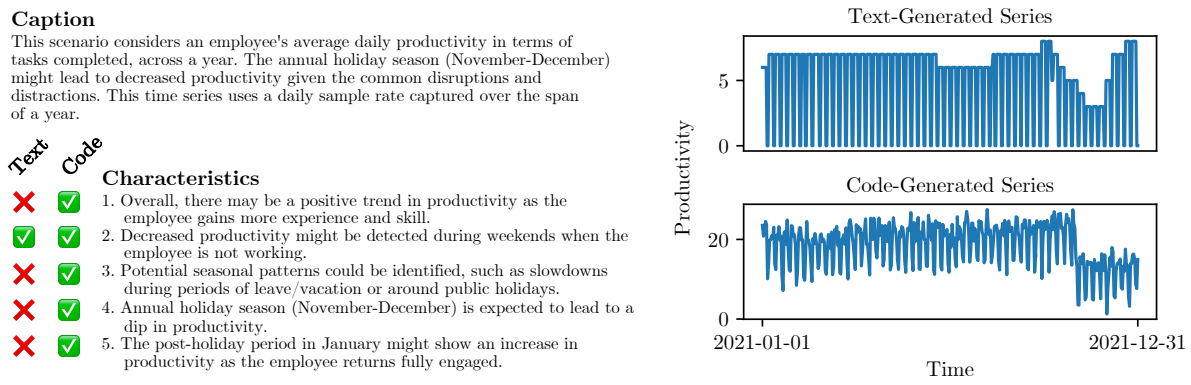


Figure 3: By simulating time series with GPT-4-generated code (rather than generating the series directly from the model itself) we're able to produce substantially more complex data which better represent realistic scenarios. Checks indicate desired characteristics are captured by the time series.

tors selected the correct caption 66.1% of the time, far above random chance of 25%. We note that skilled humans often struggle to interpret even simple time series plots (Albers et al., 2014), and so human performance on this task may not represent the upper bound of possible performance. Later, in Section 4, we will use this performance as a baseline for etiological reasoning.

Is this performance good enough? We conducted an error analysis to contextualize this result by manually annotating 50 incorrectly answered questions. We found that participants' errors fell into three categories:

- Hard (27) - There is exactly one correct answer, but annotators were not able to identify it. Common reasons for wrong answers are (a) signal noise (e.g, trends are hard but not impossible to pick out), (b) ignoring the scale of the data (e.g. the data is in the range 20-25 and the caption describes ambient temperature in degrees Celsius), and (c) misinterpreting sample rate (e.g the sample rate is visibly very high and all but one caption describes a lower frequency reading). Additional annotator training could potentially eliminate these errors.
- Flawed (13) - The correct caption has some flaw, although may not be entirely incorrect.
- Ambiguous (10) - at least one other caption is plausibly correct, although the ground truth answer remains a strong option.

While these results indicate some ambiguity, over half of errors are attributable to human skill and not noise in the underlying data ("Hard"). As discussed in Section 4, our human subjects dramatically outperformed language models on this task. The gap in performance (32.6% in Table 1) cannot be explained by the frequency of flawed or ambigu-

ous questions ($(100\% - 66.1\%) * (13 + 10)/50 = 15.6\% < 32.6\%$). Importantly, this implies that even if we assume adversarially imperfect scenarios there is still substantial room for model improvement.

4 Etiological Reasoning: Near Random Performance

By defining time series reasoning (Section 2) and creating our first-of-its kind dataset of time series and associated captions (Section 3) we can evaluate the capacity of LMs to reason about these ubiquitous data. Reasoning implies an ability to provide explanations for observed phenomena. In our context if a model can reason about a time series then it should be able to hypothesize about how that series was generated. For example, given a time series with a strong daily seasonality "sunlight intensity" is a more likely description than "Nvidia stock price since 1999."

We evaluate etiological reasoning by tasking an LLM to select the correct time series caption from a set of four, with three incorrect captions (Figure 1(a)). We sampled incorrect descriptions by randomly selecting three captions from the remainder of the dataset. To encourage the models to focus on the time series itself and not on metadata like the series' units or start and end timestamps we only provided the values of the time series. For pure language models, time series were encoded into text using the method from (Gruver et al., 2023) which uses separate schemes for LLAMA and GPT-4. Details on this method are available in Section A.1. We also experiment with representing time series as images of plots and passing them to GPT-4-Vision, as in as (Li et al., 2023).

A natural question is whether text is the correct way to represent a time series. To answer this, we experimented with five other representations (including time series as audio, spectrograms, precomputed embeddings, and images) by training existing multimodal models on our data and found no difference in performance (Section B.1).

Our results show that all models perform remarkably poorly relative to the human baseline (66.1% accuracy, Section 3.2), with some models performing at or near random chance (e.g LLAMA with 27.3% accuracy) (Table 1). GPT-4-Vision performs best (34.7%) while still falling short of human performance by over 30 percentage points.

These results indicate that current language models are poor zero-shot judges of time series etiology.

5 Question Answering: Trailing Behind Human-Level Proficiency

A LM that can reason about time series should be able to answer questions about a time series and the implications of the scenario it describes. To properly evaluate this property it should not be possible to answer the questions *without* the time series. This avoids misleading performance estimates observed in Visual Question Answering with models performing well even without the associated image (Wang et al., 2023c). A good candidate for these questions are counterfactual "what-if"-style queries that ask the LM to interpret how the time series might be different if its related scenario were changed. For example, given a time series of coffee shop sales over the course of a day with a peak at 2pm, a good "what-if" question might be, "If half as many customers visited the shop at noon, would the peak sales change?" We evaluate this ability by presenting LMs with Multiple Choice Questions (MCQs) with four options – one correct and three incorrect.

We evaluate question answering using the same techniques as etiological reasoning (Section 4). Additionally, in Section B.4 we experiment with representing the time series as plain, unformatted text and show no appreciable difference. Human performance was again assessed using a team of ten data scientists who annotated 500 time series plots using the exact same data as the LMs (metadata, time series [as a plot], and the short description).

5.1 Time Series Questions

All models showed near-random performance (except the one generating the MCQs). To create time series MCQs that cannot be answered by LMs without attending to the time series itself, we first create 'what-if' scenarios for a time-series *alongside a second time series that materializes this counterfactual scenario.* We create these MCQs using a three-step procedure.

- For each time series x (Section 3) we query GPT-4 to produce a 'what-if' scenario and a corresponding generative function for $\overline{x}$, which reflects that scenario.
- We use the 'what-if' scenario, short captions, time series x and $\overline{x}$, and their generating functions to generate MCQs about similarities and

Model/Task	Etiological Reasoning	Question Answering	
		Original	Perturbed
Random baseline	25%	25%	25%
Human	66.1%	67.0%	61.7%
LLAMA-7B- No TS	N/A†	24.7%	25.6%
LLAMA-7B	27.3%	25.2%	24.3%
LLAMA-13B- No TS	N/A†	26.3%	25.6%
LLAMA-13B	27.8%	25.8%	25.6%
GPT-3.5- No TS	N/A†	29.8%	26.3%
GPT-3.5	33.5%	27.4%	27.7%
GPT-4- No TS	N/A†	51.3%*	28.4%
GPT-4	33.5%*	52.7%*	28.4%
GPT-4-Vision	33.5%*	**53.6%***	**30.5%**
Gap - Human vs Best LM	32.6%	13.4%	33.3%

* GPT-4 generated all data and its performance should be considered an upper bound of true capability (Section 3).
† These results are not included for etiological reasoning because in this task models *only* have the time series (and no metadata) as input.

Table 1: Accuracy of LMs on Etiological Reasoning and Question Answering. Human performance was evaluated on a subset of data (N=500). **No TS** indicates that the model was evaluated without the time series as input (i.e. with only metadata in the prompt). *Etiological Reasoning:* Performance is near-random for LLAMA models and slightly better for GPT models. Human performance is significantly higher. *Question Answering:* LM performance is near-random for LLAMA models, and is slightly better for GPT models, though again trailing human performance (Section 5).

differences between x and $\overline{x}$.

- To ensure that all MCQs are answerable only in the presence of *both* time series, we filtered out questions that GPT-3.5 could answer in the absence of *any* time series, which led to almost half of the MCQs being discarded. In total, this process generated over 130k MCQs, with one correct and three incorrect answers each. An example question is in Figure 1.

We also experimented with generating questions about a single time series, but found that language models could successfully answer these questions *even without the time series*, making them poor tools for evaluating time series reasoning. More details on these experiments are available in Section B.3.

We make the following observations: (1) All LMs, other than GPT-4, had close to random performance (Table 1). (2) Only GPT-4 achieves non-trivial performance on this MCQ task. However, performance does not meaningfully increase when the time series is added to the LM input. Again, the fact that GPT-4, with and without time series, achieves non-trivial performance may be because GPT-4 was used to generate these scenarios. We describe additional experimental evidence that is consistent with this interpretation in Section 5.2. (3) Human performance, when given the exact same information as the LMs is significantly higher than all LMs at 67% which perform at near-random performance (other than the aforementioned GPT-4 and GPT-3.5 exceptions). This gap to human performance demonstrates that higher LM performance should be possible given the information available.

One potential reason for LMs performing just as badly even with a time series representation is that these time series may not contain any relevant information. However, since human performance is substantial at 67% we can rule out this possibility. The only model achieving meaningful levels of performance in the MCQ task with multiple time series is GPT-4, and we have to caution again that GPT-4 was used to generate these MCQs and this evaluation is likely to overestimate generalization performance of GPT-4.

5.2 Manually-Perturbed MCQs

Minor manual perturbations in MCQs eradicate above-random zero-shot performance for any LM, including GPT-4 which generated all data. Upon first inspection it is notable that GPT-4 achieved non-trivial levels of performance in question answering. However, we show that this performance is possibly explained by GPT-4 being the model used to synthetically generate these data

and MCQ tasks, casting significant doubt on any actual time series reasoning ability of GPT-4, and therefore *all* of the LMs evaluated in this study.

We demonstrate this by taking 144 samples from the previously described MCQ dataset and make manual perturbations to the answers. For each question we select the correct answer for the MCQ and create a similar incorrect answer as a *distractor* by editing the numerical values so that they are similar while still incorrect. We provide an example in Section B.6.

Prior to the manual perturbations, GPT-4 and GPT-4- No TS answered over half the MCQs correctly. However, after only minor changes to MCQ options performance decreases to near-random performance as well (Table 1). This strongly suggests that GPT-4's above-random performance in all prior time series MCQ tasks is due to the fact that it created the data and MCQs itself, and that does not generalize to slightly varied settings. We hypothesize (i.e., do not claim or prove) that the prior non-trivial performance is explained by the model recognizing likely correct answers due to artifacts of the distribution that this LM models.

We note that we experimented with using GPT-4 to automatically perturb questions as in Hong et al. (2024), but were unable to generate questions that were sufficiently hard.

In summary we show that LMs exhibit near-random performance on meaningful QA tasks while human evaluations demonstrate that significantly better performance is possible. **In none of these zero-shot evaluations did LMs perform better with than without the time series, suggesting that current LMs cannot integrate information from time series to answer questions.**

6 Context-Aided Forecasting

We next evaluate whether LMs can leverage relevant textual context when forecasting future time series values. We build on recent works that find LMs can non-trivially zero-shot forecast time series (Gruver et al., 2023; Xue and Salim, 2023). Using the same zero-shot forecasting method as LLM-TIME (Gruver et al., 2023), we experiment with prepending different corresponding textual context alongside the time series. We randomly select 2000 time series with their captions, descriptions, and metadata, feed the first 80% as context and forecast the remaining 20% of the timesteps. This textual context contains highly-relevant information, including *future information* about the series' behavior. To understand how well these methods compare to a simple baseline we include the "Predict Median" baseline, which simply computes the median of the first 80% of a time series' values then repeats it for the forecasting window.

We measure forecasting success using the common metrics Mean Absolute Error (MAE) and Mean Squared Error (MSE). Since the values of the time series in our dataset span several orders of magnitude we min/max and z-score normalize values before computing these metrics so that error on high-magnitude series does not dominate perceived model performance.

Highly-relevant captions barely change LM forecasts. As shown in Figure 5, adding all textual context only marginally improves MAE. Of 2,000 zero-shot samples, only 1,040 show improvement in MAE when the full context is shown and in the remaining time series MAE *increases*. An example is illustrated in Figure 4, showing that the LM ignores potentially useful information in the context. We also experimented with other combinations of metadata, characteristics, and descriptions and found that adding more information gradually improves performance, but overall performance remains below or comparable to the weak "Predict Median" baseline (Section E). We include an example where the model appears to integrate context in Figure A.6.

This lack of improvement is surprising and demonstrates a clear gap in these LM-powered methods' capacities to leverage relevant text when forecasting time series. Further, neither forecasting method clearly outperforms the simple "Median Prediction" baseline. We note that because our series were intentionally designed to contain interruptions from external events (Section 3) median prediction is a particularly weak baseline on our dataset. **This experiment shows that current LMs largely fail to use context to inform forecasting.**

7 Related Work

7.1 Datasets for Time Series and Language

Dozens of time series classification and forecasting datasets aggregate data from diverse domains (Tan et al., 2020; Dau et al., 2018; Bauer et al., 2021; Grauman et al., 2023). Unlike these works, we evaluate the relationship between time series and text and motivate time series reasoning as an area

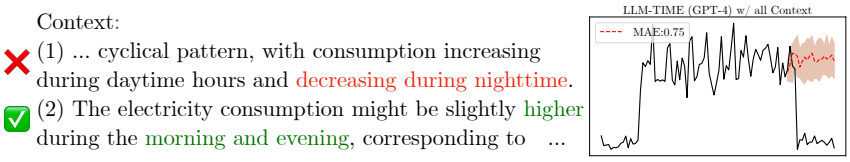
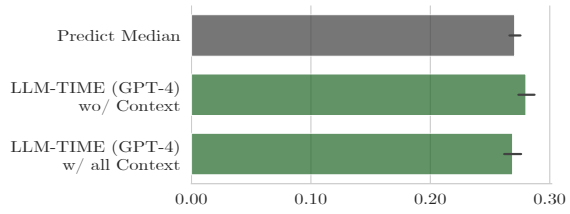
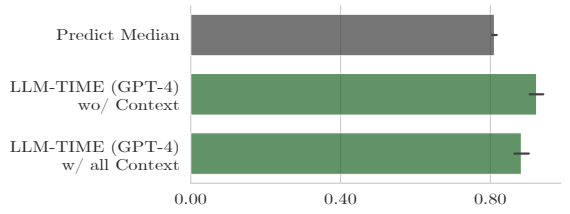


Figure 4: An example of forecasting with context. This data is sampled every 15 minutes from 0:00 to 23:45, with electricity usage dropping sharply near midnight. Forecasting starts at 19:15. The left side displays the captions in our dataset and the right side presents the performance of LLM-TIME (GPT-4) which fails to incorporate this highly-relevant information.



(a) MAE with max-min normalization



(b) MAE with z-score normalization

Figure 5: After adding contextual information corresponding to the time series, forecasting performance improved *marginally* and is still the same or worse than a simple baseline that only predicts the median of the historical signal (Section 6).

of research beyond forecasting and classification. Some datasets focus on single-domain question answering with time series. Oh et al. (2023) and Xing et al. (2021) provide question answering datasets based on templated questions about ECGs and activity recognition, whereas Xie et al. (2023) present templated questions that concern tweets and stock price data. Jhamtani and Berg-Kirkpatrick (2021) provide simple captioned time series, but these are abstract shapes with no semantic grounding and simple captions like "consistent in the first two thirds" or "slightly climbs up at the end".

7.2 Language Models and Time-Series

Recent work demonstrates that LMs can perform time series forecasting (Gruver et al., 2023) and classification (Zhou et al., 2023). These methods be categorized into two paradigms. The first involves fine-tuning LMs, such LLAMA-7B, for specific tasks and datasets (Zhou et al., 2023; Jin et al., 2024; Cao et al., 2024). The second approach entails inputting specially tokenized time series into an LM for forecasting, imputation, and classification tasks (Gruver et al., 2023; Xue and Salim, 2023).

Most methods require fine-tuning the model with domain-specific data. In cross-domain tasks, the strategy often involves fitting one dataset and then transferring to another (Jin et al., 2024; Cao et al., 2024; Zhou et al., 2023; Wang et al., 2023a). This approach is not suitable for our dataset, where each time series originates from a different setting. Therefore, to evaluate our entirely cross-domain dataset, we utilize the latest state-of-the-art zero-shot method, LLM-TIME Gruver et al. (2023), as our baseline.

8 Limitations

One limitation of this work is its reliance on synthetic data. While we go to great care to manually validate the quality of the data (Section 3.2) and provide examples of our scenarios (Figure A.2) we nonetheless recognize that questions may arise about our data's realism. It is important to remember that no "real" dataset of diverse time series and highly relevant text exists. By providing our dataset, tasks, and evaluations we provide progress that would not be possible without synthetic data. We leave it to future work to mine and document a similar "real" dataset.

A related limitation is that because GPT-4 was used to generate data (and some questions in Section 5) it is possible that the performance of this model (and this model only) is an over-estimate of true ability. We provide evidence to this end in Section 5.2, where we show that manually perturbed MCQs are harder for GPT-4 but just as difficult for humans. Nevertheless, even if we assume that this is an overestimate the substantial gap between human and GPT-4 performance on all tasks indicates

significant room for improvement.

9 Conclusion

We identified three forms of time series reasoning and used them to create a first-of-its-kind dataset of time series and highly relevant text. We then used this dataset to assess etiological reasoning, question answering, and context-aided forecasting. Given the substantial gap between language model and human performance on the first two tasks, and mediocre performance on the third, we identified opportunities for the NLP community to develop models that can deeply reason about these critical data.

Acknowledgements

This research was supported in part by NSF CAREER IIS-2142794, Bill & Melinda Gates Foundation (INV-004841), NSF IIS-1901386, NSF CNS-2025022, the Microsoft Accelerating Foundation Models Research Program, and UW eScience Azure Cloud Computing support.

-

References

Mubashara Akhtar, Abhilash Shankarampeta, Vivek Gupta, Arpit Patil, Oana Cocarascu, and Elena Simperl. 2023. Exploring the numerical reasoning capabilities of language models: A comprehensive analysis on tabular data. In *EMNLP*.

Danielle Albers, Michael Correll, and Michael Gleicher. 2014. Task-driven evaluation of aggregation in time series visualization. In *SIGCHI*.

André Bauer, Marwin Züfle, Simon Eismann, Johannes Grohmann, Nikolas Herbst, and Samuel Kounev. 2021. Libra: A benchmark for time series forecasting methods. In *ICPE*.

Nathaniel Beck and Jonathan N Katz. 2011. Modeling dynamics in time-series–cross-section political economy data. *Annual review of political science*, 14:331–352.

B Benson, WD Pan, A Prasad, GA Gary, and Q Hu. 2020. Forecasting solar cycle 25 using deep neural networks. *Solar Physics*, 295(5):65.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Kamar, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

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. In *ICLR*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde, Jared Kaplan, Harrison Edwards, Yura Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, David W. Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William H. Guss, Alex Nichol, Igor Babuschkin, Suchir Balaji, Shantanu Jain, Andrew Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew M. Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Hoang Anh Dau, Eamonn Keogh, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, Yanping, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, Gustavo Batista, and Hexagon-ML. 2018. The ucr time series classification archive. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Sean Welleck, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. 2023. Faith and Fate: Limits of Transformers on Compositionality. In *NeurIPS*.

Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, Eugene Byrne, Zachary Chavis, Joya Chen, Feng Cheng, Fu-Jen Chu, Sean Crane, Avijit Dasgupta, Jing Dong, María Escobar, Cristhian Forigua, Abrham Kahsay Gebreselasie, Sanjay Haresh, Jing Huang, Md Mohaiminul Islam, Suyog Dutt Jain, Rawal Khirodkar, Devansh Kukreja, Kevin J Liang, Jia-Wei Liu, Sagnik Majumder, Yongsen Mao, Miguel Martin, E. Mavroudi, Tushar Nagarajan, Francesco Ragusa, Santhosh K. Ramakrishnan, Luigi Seminara, Arjun Somayazulu, Yale Song, Shan Su, Zihui Xue, Edward Zhang, Jinxu Zhang, Angela Castillo, Changan Chen, Xinzhu Fu, Ryosuke Furuta, Cristina Gonzalez, Prince Gupta, Jiabo Hu, Yifei Huang, Yiming Huang, Weslie Khoo, Anush Kumar, Robert Kuo, Sach Lakhavani, Miao Liu, Mingjing Luo, Zhengyi Luo, Brighid Meredith, Austin Miller, Oluwatumininu Oguntola, Xiaqing Pan, Penny Peng, Shraman Pramanick, Merey Ramazanova, Fiona Ryan, Wei Shan, Kiran Somasundaram, Chenan Song, Audrey Southerland, Masatoshi Tateno, Huiyu Wang, Yuchen Wang, Takuma Yagi, Mingfei Yan, Xitong Yang, Zecheng Yu, Shengxin Cindy Zha, Chen Zhao, Ziwei Zhao, Zhifan Zhu, Jeff Zhuo, Pablo Andrés Arbeláez,

Gedas Bertasius, David J. Crandall, Dima Damen, Jakob Julian Engel, Giovanni Maria Farinella, Antonino Furnari, Bernard Ghanem, Judy Hoffman, C V Jawahar, Richard A. Newcombe, Hyun Soo Park, James M. Rehg, Yoichi Sato, Manolis Savva, Jianbo Shi, Mike Zheng Shou, and Michael Wray. 2023. Ego-exo4d: Understanding skilled human activity from first- and third-person perspectives. *arXiv preprint arXiv:2311.18259*.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. 2023. Large language models are zero-shot time series forecasters. In *NeurIPS*.

Jack Hessel, Ana Marasović, Jena D Hwang, Lillian Lee, Jeff Da, Rowan Zellers, Robert Mankoff, and Yejin Choi. 2023. Do androids laugh at electric sheep? humor "understanding" benchmarks from the new yorker caption contest. In *ACL*.

Pengfei Hong, Deepanway Ghosal, Navonil Majumder, Somak Aditya, Rada Mihalcea, and Soujanya Poria. 2024. Caught in the quicksand of reasoning, far from agi summit: Evaluating llms' mathematical and coding competency through ontology-guided interventions. *Preprint*, arXiv:2401.09395.

Anu Jagannath, Jithin Jagannath, and Tommaso Melodia. 2021. Redefining wireless communication for 6g: Signal processing meets deep learning with deep unfolding. *IEEE Transactions on Artificial Intelligence*, 2(6):528–536.

Harsh Jhamtani and Taylor Berg-Kirkpatrick. 2021. Truth-Conditional Captions for Time Series Data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 719–733. Association for Computational Linguistics.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. 2024. Time-llm: Time series forecasting by reprogramming large language models. In *ICLR*.

Andreas Kamilaris and Francesc X Prenafeta-Boldú. 2018. Deep learning in agriculture: A survey. *Computers and electronics in agriculture*, 147:70–90.

Zekun Li, Shiyang Li, and Xifeng Yan. 2023. Time series as images: Vision transformer for irregularly sampled time series. In *NeurIPS*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023a. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*.

Xin Liu, Daniel McDuff, Geza Kovacs, Isaac Galatzer-Levy, Jacob Sunshine, Jiening Zhan, Ming-Zher Poh, Shun Liao, Paolo Di Achille, and Shwetak Patel. 2023b. Large language models are few-shot health learners. *arXiv preprint arXiv:2305.15525*.

Thomas Mesnard, Gemma Team, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, and et al. 2024. Gemma.

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).

Marc Nerlove, David M Grether, and Jose L Carvalho. 2014. *Analysis of economic time series: a synthesis*. Academic Press.

Jungwoo Oh, Gyubok Lee, Seongsu Bae, Joon-myoung Kwon, and Edward Choi. 2023. Ecg-qa: A comprehensive question answering dataset combined with electrocardiogram. In *NeurIPS*.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision. *Preprint*, arXiv:2212.04356.

Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. 2020. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied soft computing*, 90:106181.

Chang Wei Tan, Christoph Bergmeir, Francois Petitjean, and Geoffrey I Webb. 2020. Monash university, uea, ucr time series extrinsic regression archive. *arXiv preprint arXiv:2006.10996*.

Junxiang Wang, Guangji Bai, Wei Cheng, Zhengzhang Chen, Liang Zhao, and Haifeng Chen. 2023a. Prompt-based domain discrimination for multi-source time series domain adaptation. *arXiv preprint arXiv:2312.12276*.

Shiqi Wang, Zheng Li, Haifeng Qian, Cheng Yang, Zijian Wang, Mingyue Shang, Varun Kumar, Samson Tan, Baishakhi Ray, Parminder Bhatia, Ramesh Nallapati, Murali Krishna Ramanathan, Dan Roth, and Bing Xiang. 2023b. Recode: Robustness evaluation of code generation models. In *ACL*.

Ziyue Wang, Chi Chen, Peng Li, and Yang Liu. 2023c. Filling the image information gap for vqa: Prompting large language models to proactively ask questions. In *EMNLP*.

Peter West, Ximing Lu, Nouha Dziri, Faeze Brahman, Linjie Li, Jena D. Hwang, Liwei Jiang, Jillian R. Fisher, Abhilasha Ravichander, Khyathi Raghavi Chandu, Benjamin Newman, Pang Wei Koh, Allyson Ettinger, and Yejin Choi. 2024. The generative ai paradox:" what it can create, it may not understand". In *ICLR*.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2023. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *ICLR*.

Qianqian Xie, Weiguang Han, Xiao Zhang, Yanzhao Lai, Min Peng, Alejandro Lopez-Lira, and Jimin Huang. 2023. Pixiu: A comprehensive benchmark, instruction dataset and large language model for finance. In *NeurIPS*.

Tianwei Xing, Luis Antonio Garcia, Federico Cerutti, Lance M. Kaplan, Alun David Preece, and Mani B. Srivastava. 2021. Deepsqa: Understanding sensor data via question answering. In *IoTDI*.

Hao Xue and Flora D. Salim. 2023. Promptcast: A new prompt-based learning paradigm for time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–14.

Xiyuan Zhang, Ranak Roy Chowdhury, Rajesh K Gupta, and Jingbo Shang. 2024. Large language models for time series: A survey. *arXiv preprint arXiv:2402.01801*.

Li Zhong and Zilong Wang. 2023. A study on robustness and reliability of large language model code generation. *arXiv preprint arXiv:2308.10335*.

Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. 2023. One fits all: Power general time series analysis by pretrained lm. In *NeurIPS*.

A Appendix

A.1 Numerical Tokenization

We use the LLM-TIME (Gruver et al., 2023) as a baseline for "contextual reasoning" to evaluate LLM's reasoning performance in time series forecasting when captions are provided. The performance of LLM-TIME is partly attributable to their special numerical tokenization method. The original input ($20.88,\ 20.20,\ 20.48, \ldots$ below), is first z-score normalized and then scaled to a constant power of ten ($1e3$ below):

$$20.88,\ 20.20,\ 20.48, \ldots$$
$$\downarrow$$
$$1.0522,\ 1.0178,\ 1.0324, \ldots$$
$$\downarrow$$
$$1052,\ 1017,\ 1032, \ldots$$

Note that there are subtle differences in tokenization for GPT-4 and LLaMA.[2]

B Additional Results

B.1 Training Multimodal Models on Etiological Reasoning Task

Is putting time series into a prompt as text the best way to model these data? Here we experiment with five alternative modeling techniques, each adapted from an existing multimodal architecture. When training models we wanted to keep the results roughly comparable to zero-shot experiments so we reserved the "Health and Medical Data", "Agricultural and Food Production" and "Educational and Public Services" categories for testing and trained on the remainder. Where models are trained the default learning rate from the model's repository was used.

Whisper. Speech-to-text models can be thought of as special cases of time-series-to-text models since microphone-recorded audio is a 1D sensor reading. We modify Whisper (Radford et al., 2022) to compute spectrograms of arbitrary time series and fuse these with GPT-2 inputs via cross attention.

LLAVA-Matplotlib-Zero-Shot. (Liu et al., 2023a) supports visual instruction tuning by training a linear adapter between a vision encoder and a language model's token embedding space. Following Li et al. (2023) we encode time series by plotting them in Matplotlib and saving the results as 224x224 images. These images are fed directly into LLaVA's pretrained CLIP encoder. As the name suggests, this model was not trained and instead relies entirely on the pretrained LLaVA weights.

[2]https://github.com/ngruver/llmtime

Model/Task	Etiological Reasoning
Human	66.1%
Whisper	23.6%
LLAVA-Matplotlib-Zero-Shot	24.3%
LLAVA-Matplotlib	26.1%
LLAVA-TimesNet	23.5%
LLAVA-Spectrogram	26.1%

Table A.1: Perfomance of multimodal models trained on the etiological reasoning task (Section 4)

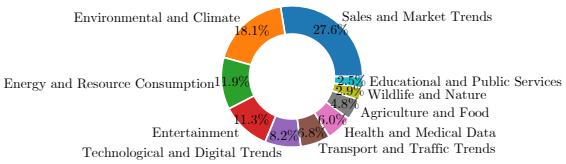


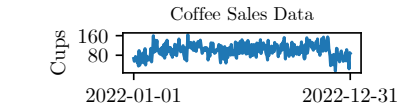
Figure A.1: Portion of scenario categories in our generated dataset (Section 3).

LLAVA-Matplotlib. This experiment is the same as the previous, but we began by tuning LLaVA's adapters using the seven held-out scenario categories.

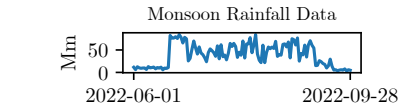
LLAVA-Spectrogram. Spectrograms are 2D representations of a time series and can be passed to standard vision encoder. For this experiment we computed spectrograms and fed them imto LLaVA's clip encoder.

LLAVA-TimesNet. In this experiment we replaced LLaVA's CLIP encoder with the TimesNet (Wu et al., 2023) encoder. TimesNet adaptively maps 1D time series signals into a 2D space that can be interpreted by computer vision kernels and was designed as a general-purpose time series encoder. Since there is no pretrained TimesNet checkpoint in this experiment we freeze only the LLaMA backbone and allow the model to learn weights in the encoder.

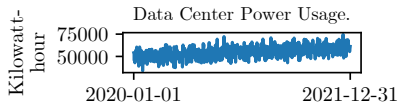
The results show that all models struggle to learn etiological relationships between time series and text. Each model performs within an epsilon of random performance (25%). We conclude that even models finetuned on these data have limited capacity to reason about time series.



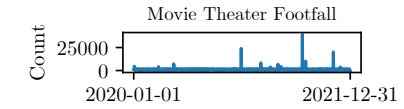
Sales and Market Trends
We track the daily coffee sales in a small café located near a college. Sales might increase during exam periods when students stay up late. The sample rate will be daily over a year (365 observations).



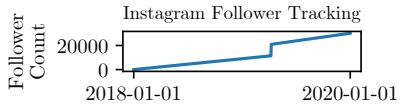
Environmental and Climate
Consider a scenario of rainfall measuring agency operating in a region known to be affected by monsoon yearly. The agency measures daily rainfall, recorded in mm, over the span of one monsoon season (120 days). Significant external events are monsoon onset and monsoon offset which substantially increase and decrease rainfall respectively.



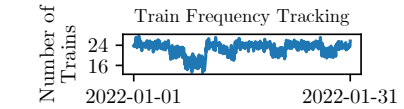
Energy and Resource Consumption
The scenario involves monitoring the daily power usage of a data center over a period of two years, recorded every day at midnight. Any increases in the usage could likely be related to additional servers being added or peak periods of client activity. Summer heat waves could also boost air conditioning use and hence power consumption.



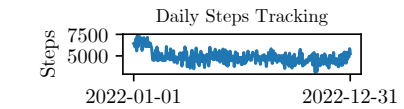
Entertainment
Over a period of two years, the time series is developed to observe and understand the pattern in the footfall count at a movie theater. Here, the external event is the release of blockbuster movies, which greatly increases the footfall. The sample rate of this time series is daily.



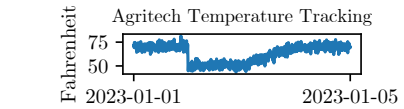
Technological and Digital Trends
This scenario tracks the daily followers count of a rising Instagram influencer over a 2-year period. The influencer starts with modest followers but gets a significant boost after being featured in a viral video on a popular YouTube channel. The time series is sampled daily and the sample rate is 1 sample per day.



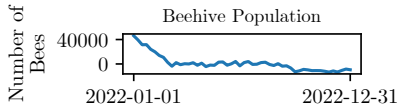
Transport and Traffic Trends
The scenario involves tracking the number of trains passing a particular station per hour over one month. An external event such as a railway strike could result in a decreased number of trains during the strike period. The time series data is sampled every hour.



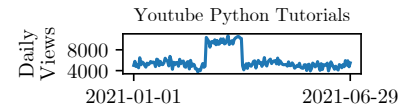
Health and Medical Data
A time series measuring daily steps of a fitness enthusiast. He promised to increase his daily steps due to the new year's resolution starting from January 1. The series will track his steps for 365 days, sampled daily. The external event here is the New Year.



Agriculture and Food
This scenario involves the monitoring of temperature in an agritech controlled environment where special hybrid crops are grown. Let's say on the 3rd day, a power outage occurred, having an impact on the heating system, thus this event will influence an unusual decrease in the temperature.



Wildlife and Nature
This scenario involves the tracking of honeybee population in a particular hive over one year. The external event could be a pesticide spray in nearby fields, resulting in a decline of the population. The sample rate is measured weekly, resulting in about 52 observations in a year.



Educational and Public Services
A tech vlogging YouTube channel decides to create a daily series about coding tutorials in Python. The channel launches the series targeting especially beginners. The series gets popularity upon the introduction of a new trending Python library for Machine learning during the month, causing a spike in viewership. The time series will be recorded on a daily basis over a duration of 180 days (6 months).

Figure A.2: One scenario from each of our ten categories (Section 3).

Model/Generator LM	LLAMA-13B	GEMMA-7B
LLAMA-13B- No TS	88.1%	88.5%
LLAMA-13B	87.0%	87.3%
GEMMA-7B- No TS	86.6%	88.5%
GEMMA-7B	87.2%	88.3%
GPT-3.5- No TS	96.8%	97.0%
GPT-3.5	96.4%	97.1%
GPT-4- No TS	97.5%	97.7%
GPT-4	97.2%	97.4%

Table A.2: Accuracy of LMs on counterfactual MCQs generated using LLAMA-13B and GEMMA-7B.

B.2 MCQ Generation using other LMs

Here, we evaluate the ability of LM other than GPT-4 to generate MCQs. Specifically, we created counterfactual scenarios and the corresponding questions using two LM – LLAMA-13B and GEMMA-7B (Mesnard et al., 2024). Across each setting, we used 100 time series examples and created a set of almost 1000 MCQs for each LLM. The results across these datasets clearly show that GPT-4 achieves significant performance across the MCQs generated using LLAMA-13B and GEMMA-7B, even in the absence of any time series information (Table A.2). This can be attributed to the limited ability of LMs in understanding the dynamics within time series data and creating questions solely based on their textual descriptions. These results reinforce that other LMs may not be suitable for generating time series-specific questions and, consequently, for training models to evaluate time series reasoning ability.

Model/Task	Question Answering (One TS)
Random baseline	25%
LLAMA-7B- No TS	78.4%
LLAMA-7B	78.8%
LLAMA-13B- No TS	82.6%
LLAMA-13B	82.5%
GPT-3.5- No TS	90.4%
GPT-3.5	88.2%
GPT-4- No TS	**92.6%**[*]
GPT-4	92.3%[*]
GPT-4-Vision	91.8%[*]

[*] GPT-4 generated all data and its performance should be considered an upper bound of true capability (Section 3).

Table A.3: Question Answering (One TS) Performance

B.3 Questions About One Time Series

'What-if' MCQs created for single time series were trivial to answer. An intuitive approach to generate MCQs for time series is to prompt a LM to use the time series and associated scenarios and metadata from Section 3 to generate questions and answers. We again use GPT-4, as questions generated by other LMs were always answerable without the timeseries (Section B.2). First, we prompt GPT-4 with the with all the information generated in Section 3, *i.e.,* time series, short caption, characteristics, generative function, and metadata, to generate a potential counterfactual 'what-if' scenario. Second, we prompt GPT-4 to generate questions around the original time-series and the possible changes due to 'what-if' scenarios and obtain 100k single time series MCQs (full prompt in Section D.1, and examples in Section B.5).

In early experiments, we found that giving the LM access to the full caption consistently led to questions that were entirely dependent on the caption and did not reference the time series. Even after removing the caption from the question generating procedure, all LMs achieved 78-92% accuracy, *even when they were not provided the time series.* This demonstrates that these questions did not actually necessitate time series reasoning (Table A.3).

We further experimented with changing the order of options within MCQs, used prompts with different sets of time-series features, generative functions, metadata, and presented time series as plain text and as tokens using the procedure in LLM-TIME (Gruver et al., 2023). However, none of these attempts produced MCQs that required the time series in order to answer them correctly.

We make the following observations: (1) LM performance overall was high, ranging from 78-92% *without the time series*. This creates a false impression of LM time series reasoning ability, when really the performance stems from text-based parametric LM knowledge instead. (2) Since these data and questions were generated by GPT-4, with GPT-3.5 potentially sharing training data and other components, it is perhaps less surprising that they are significantly better than LLAMA models. We therefore caution to interpret these results as a sign of generalizable time series reasoning ability, which is further called into question by the experiments described next.

Since LMs performed well even in the absence of time series, we deemed this setting unsuitable for evaluating time series reasoning, and did not perform additional human evaluation.

Model/Task	Single TS MCQ	
	Plain Text	LLM-TIME
LLAMA-7B	78.6	**78.8%**
LLAMA-13B	82.4	**82.5%**
GPT-3.5	88.2	88.2%
GPT-4	92.2	**92.3%**

Model/Task	Multiple TS MCQ	
	Plain Text	LLM-TIME
LLAMA-7B	**25.2%**	25.1%
LLAMA-13B	25.7%	**25.8%**
GPT-3.5	27.0%	**27.1%**
GPT-4	52.5%	52.5%

Table A.4: LMs' accuracy on MCQs when time-series are given as comma-separated values in plain text and tokenized using LLM-TIME.

B.4 Using Different Methods to Prompt Time Series

Here, we evaluate different methods of passing a time series to a language model. This task is incredibly important, as recent research has shown that changing the tokenization for time series can lead to it being easily confused by language models and can result in state-of-the-art results in forecasting (Gruver et al., 2023). Therefore, in this section, we compare two methods used in LLM-TIME (Gruver et al., 2023): specifically, passing tokens as comma-separated values and using the tokenization procedure described in Appendix A.1. Our results across both methods show insignificant differences in the ability of LMs to answer MCQs (Table A.4). However, we note that LM with time series encoded as LLM-TIME obtains slightly better performance.

B.5 Examples of Single Time Series MCQs

Here we provide a few examples of single-time series MCQs. Specifically, for the time series given in Figure A.3, we queried GPT-4 and obtained the following MCQs.

```
Q. How would the series be affected if the cafe
    started to remain open all night?
   A. If the cafe started to remain open all
       night the timeseries would show no
       change in customer counts as the
       patterns remain the same.
   B. If the cafe remained open all night the
       periods that previously showed zero
       customer counts due to closure would now
       show some level of customer activity.
       However the counts during these late
       hours would typically be lower compared
       to the breakfast and lunch times.
   C. If the cafe started to remain open all
       night the timeseries would show higher
       customer counts during the day and a
       decrease in counts during the night when
        the cafe is closed.
   D. If the cafe started to remain open all
       night the timeseries would show
       significant spikes in customer counts
       throughout the day.

Q. How would the time series be different if the
     TV show started to air on Wednesdays
    instead of Sundays?
   A. If the TV show started to air on
       Wednesdays instead of Sundays the
       timeseries would show no change in
       customer counts as the patterns remain
       the same.
   B. If the TV show aired on Wednesdays instead
       of Sundays the pronounced spikes in the
       customer counts would shift to reflect
       this change. This means we would start to
       see the spikes on Wednesdays and
       continue for the next few days following
       the broadcast.
   C. If the TV show started to air on
       Wednesdays instead of Sundays the
       timeseries would show increased customer
        counts throughout the week.
   D. If the TV show started to air on
       Wednesdays instead of Sundays the
       timeseries would show a decrease in
       customer counts on Wednesdays and an
       increase on Sundays.,
Q. What would the effect on the customer count
    be if the cafe started serving dinner and
    remained busier during evening hours?
   A. If the cafe started serving dinner and
       remained busier during evening hours the
        timeseries would show a decrease in
       customer counts during dinner time.
   B. If the cafe started serving dinner and
       remained busier during evening hours the
        timeseries would show increased
       customer counts only during dinner time.
   C. If the cafe started serving dinner and
       remained busier during evening hours the
        timeseries would show no change in
       customer counts as the patterns remain
       the same.
   D. If the cafe became busier during dinner
       time then the unknown counts during
       evening hours would increase. This could
       introduce another cyclical pattern in
       the time series corresponding to dinner
       hours similar to those observed during
       breakfast and lunch times.
```

B.6 Manually Perturbed MCQs

In this section, we highlight the procedure we use to manually perturb the MCQs generated by GPT-4. In detail, we aim to test the robustness of GPT-4 across slightly modified versions of the same set of MCQs it generated. For this, consider the following MCQs generated by GPT-4 for two independent time series. These questions aim to compare the

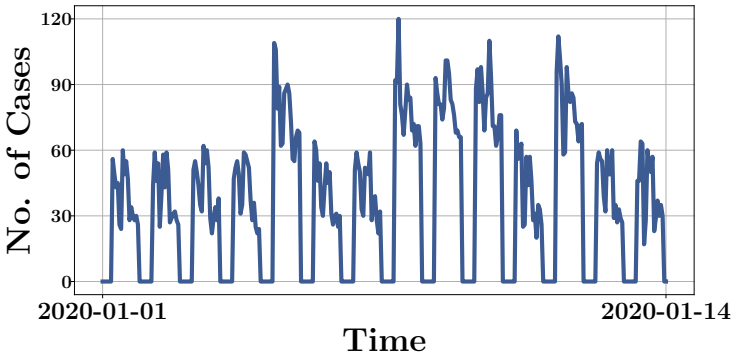


Figure A.3: An example time-series with description: 'Customer counts at a cafe following a TV feature over two week period.'

time series updated by the 'what-if' scenario with the original time series.

```
Q. Do both the original and updated time series
     have the same starting base level of
     pollution?
   A. No the updated series starts at 0 units of
        pollution.
   B. No the base level in the updated series is
        1500 units.
   C. Yes both start with a base level of 1000
        units of pollution.
   D. No the base level in the original series
        is 500 units.

Q. Is there a change in the visitor count during
     the pandemic period in the modified time
     series compared to the original?
   A. The visitor count during the pandemic does
        not drop to 0 in the modified series.
   B. There is no change the visitor count during
        the pandemic period drops to 0 in both.
   C. The visitor count during the pandemic
        becomes 1500 in the modification.
   D. The pandemic period is removed in the
        modification.
```

To change the question, we select the correct option – option C and option B respectively, and create a similarly looking incorrect option. Later, we replace this perturbed option with a randomly selected incorrect option and test the LMs' ability in responding to the MCQ. The following shows the updated MCQs with options D and C being the perturbed options. Upon evaluating both the MCQs, we note that GPT-4 and other LMs selected the perturbed option as their choice of answer. However, we also note that the LMs across different runs selected the correct option, *i.e.,* option C and Option B too. But the goal of the manual perturbation succeeds in showing that LMs cannot understand and select an answer using a time series and mostly select options based on their similarity to the option they originally generated.

```
Q. Do both the original and updated time series
     have the same starting base level of
     pollution?
   A. No the updated series starts at 0 units of
        pollution.
   B. No the base level in the updated series is
        1500 units.
   C. Yes both start with a base level of 1000
        units of pollution.
   D. Yes both start with a base level of 500
        units of pollution.

Q. Is there a change in the visitor count during
     the pandemic period in the modified time
     series compared to the original?
   A. The visitor count during the pandemic does
        not drop to 0 in the modified series.
   B. There is no change the visitor count during
        the pandemic period drops to 0 in both.
   C. There is no change the visitor count during
        the pandemic period drops to 10 in both.
   D. The pandemic period is removed in the
        modification.
```

C Prompt For Scenario Generation

We used the following prompt to generate the time series scenarios described in Section 3.

```
1.  Describe a scenario that might produce a
    time series. This scenario should include an
     external event and how it might influence
    the reading. Be sure to describe the sample
    rate of the time series and the duration
    over which it is sampled. The description
    should be less than 100 words in length.
    Delimit this description with the XML tag <
    description>.

    The time series must be less than 1000
        observations in length, be a single
        variable, have no values greater than 1
        e6, and have no missing values.

    Also add a summary of the description, no
        more than 25 words in length with the
        tag <description_short>. Also add
        summary, no more than three words in
        length with the tag <description_tiny>.
        The scenario should be as different as
```

possible from any of the following: [<previous_descriptions>]

2. You will generate a list of up to five characteristics of this specific time series, including patterns that you might expect to see in the series and how external events might cause distribution shifts in the data generating process. Delimit these characteristics with the XML tag <characteristics>.

3. You will write a numpy function called `generate_series` that takes no arguments and outputs a time series that matches the description. All parameters from the data generating process should be drawn from reasonable distributions. The function must return a single numpy array. Place this code inside a python markdown block and delimit your code with the XML tag <generator>. Do not call the function, simply define it. You should also make sure that the scale of time series is realistic. For example, a time series of a quantity like stock price should never be less than zero.

4. Return a json string, delimited by the tag <metadata> that contains the units of the time series and the timestamps corresponding to the first and last values. Remember that in JSON format datetimes must be passed as strings. Also include a string that relects the frequency of the time series.

Here is an example of a complete response:
<description> *your description* </description>
<description_short> *your description* </description_short>
<description_tiny> *your description* </description_tiny>
<characteristics> *your characteristics* </characteristics>
<generator>
```python
def generate_series():
    # your code here
    return x
```
</generator>
<metadata>
    {
    "start": x,
    "end": y,
    "units": z,
    "frequency" : freq
    }
</metadata>

D Prompt For MCQ Generation

D.1 Prompt for Single Time-Series MCQs

We use the following prompt to generate the MCQs around single-time series described in Section 5.

1. Given a description of a time-series, a set of sentences describing its characteristics, and a python code segment that generates this time-series. You have to create five counterfactual question-answer pairs. Counterfactual reasoning questions involve exploring hypothetical scenarios by considering what would have happened if certain events or conditions had been different from what actually occurred.
2. For example, 'What will the time-series look like if some event occured?'. Generate a wide-range of questions. Create questions and answers that avoid referencing or directly quoting code or the description. Avoid asking questions specifically tied to the description or the Python code. The questions should require an understanding of time-series dynamics for accurate answers.

3. The answers should not mention the description or the code at all. Provide the questions and answers in the following exact format: '{'category':'"+et+"', 'question':'', 'answer':''}'. Ensure that each question and its corresponding answer are presented on the same line, with each new question starting on a new line for a clear and organized format.

4. Using the set of question-answer pairs, create three incorrect answer options for each question. Your incorrect answers should have similar lengths compared to the correct answers. The input format is: '{'question':'', 'answer':''}'. In the output, you should copy the question and answers from the input and provide incorrect options in the following format: '{'question':'', 'answer':'', 'incorrect answer 1':'', 'incorrect answer 2':'', 'incorrect answer 3':''}\n'. Each new question should start on a new line. Do not separate question, its answer and options into different lines. Ensure that each question, its corresponding answer and incorrect answers are presented on the same line. Do not use any double quotations within the text.

5. Avoid the use of contractions in all kinds of notations. Instead, use the full forms for greater clarity. If there exists any contraction in the question or answer, then replace it with the full-form. Do not generate any additional text.

D.2 Prompt for Multiple Time-Series MCQs

For generating MCQs that operate at the intersection of multiple time-series, we employed the following steps:

D.2.1 Creating a list of 'what-if' scenarios for a time series

1. You have been given a description of a time series and a code that generates the time-series. Your task is to create five counterfactual questions that someone can ask regarding this time series.

2. Try to formulate questions that are distinct from each other. Additionally, ensure that the questions aim to bring about significant changes to the time series. Make sure that the new time series can be easily generated by modifying the code and do not ask extremely difficult questions.

3. Format the output as follows:'{'question'}'\n, with each new question starting on a new line. The counterfactual questions should explore hypothetical scenarios and involve 'What-if' type inquiries. The questions should not include values directly from the original time series or code. For instance, 'What if the start was 25 units' is preferred over 'What if the start was 25 units instead of 20 units?'.

4. Avoid referencing random noise, the random number generator, its mean, or variance in any question. Do not generate any additional text.

D.2.2 Creating a new time-series

For each time series x (Section 3) and a 'what-if' scenario outlined in the previous paragraph, we employ GPT-4 to generate the corresponding generative function. This function simulates a second time series, denoted as $\overline{x}$, reflecting the 'what-if' scenario. We used the following prompt to generate the updated time series

1. Generate a new Python code for a time series based on the given code and description. The user will specify a change in the time series, and you should produce the updated code using the function name 'generate_series'.

2. Always ensure the length of the time series remains unchanged. This is hard constraint that should not be violated. Keep realistic expectations and ensure the length of the time series remains unchanged. For example, (1) keep the rate of change consistent rather than the actual values. (2) Understand what changes the user's suggestion can make to the time series and then update the code accordingly. (3) Given a time series code, you have the freedom, and in some cases, the obligation, to modify any pre-defined maximum or minimum values specified in the original code to accurately represent the desired change.

3. Ensure that the new time series adheres to real-world principles; for instance, maintaining a consistent rate of change under typical conditions. If the change demands that the time series has an offset by some units, then modify this value in the code as well.

4. Return the output in the format ```new code ```, where the 'new code' is replaced by the updated code. Try to create code that generates a time-series that is significantly different from the time-series produced by the original code, but with same lengths.

5. Always return the code in a format that can be executed directly using the exec() function. Avoid additional text.

D.3 Creating MCQs

Utilizing the 'what-if' scenario, brief captions, and both time series x and $\overline{x}$, along with their generating functions, we construct multiple-choice questions (MCQs). These MCQs aim to evaluate the similarities and differences between the two time series. We used the following prompt to generate the MCQs around single-time series described in Section 5.

1. Given two Python codes for generating time series, the first representing the original time series with a description, and the second presenting a modification of the original time series under specific conditions.

2. Your task is to ask five questions regarding the differences between both time series. Also ask five questions regarding the simmilarity between both time series. Additionally, provide answers to all the questions and three negative or incorrect options. Ask questions regarding the patterns within both time-series, such as how they appear, the rates of change, and any specific differences in trends. Format the output as follows:'{'category':' difference/simmilarity', 'question':'', 'answer':'', 'incorrect answer 1':'', 'incorrect answer 2':'', 'incorrect answer 3':''}'.

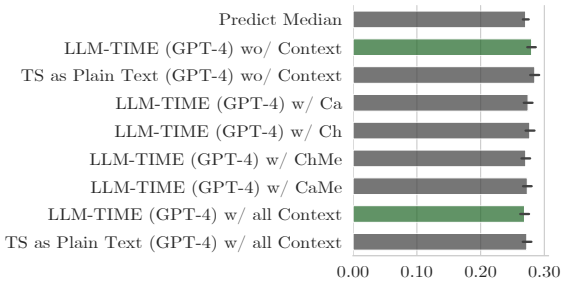
3. Make sure you follow the following rules: (1) Do not ask question regarding the lengths or the number of data-points within both the time-series. (2) Ensure that the questions and answers give the impression of being created independently, in the absence of the code, solely by examining the time series. (3) Do not mention anything regrarding the random noise or random number generator in both the answers and questions . (4) Try to keep the answers short and not very detailed. (5) Ensure that each question and its corresponding answer are presented on the same line, with each new question starting on a new line for a clear and organized format. (6) Try to add numerical values to answers wherever possible, but make sure you use words such as 'seems to be ' or 'around value' so that they appear to be approximate. Avoid unnecessary text and focus on precision.

E Additional Results for Context-Aided Forecasting

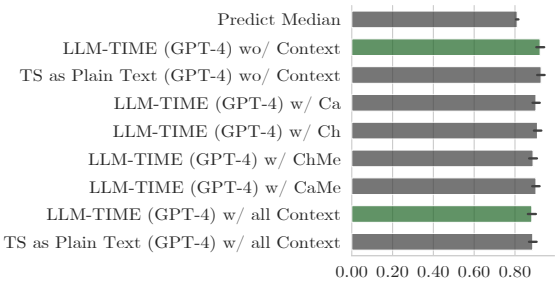
In this section, we will present more results and examples on how LM reasons through context in forecasting. Figure A.4 shows the full results for two metrics, MAE and MSE, both derived from the average of 2000 samples. Each result will be independently normalized before calculating the metrics. Overall, it can be seen that as more captions are provided, LM's reasoning in forecasting only improves slightly. Even when all captions are provided, the aid remains quite *marginal*. Two examples of how LM integrates context into forecasting are shown in Figure A.5, where figure (a) demonstrates that LM can reason out difficult-to-forecast distribution shifts from captions. However, as seen in figure (b), even when highly-relevant caption are provided, it still does not enhance the forecasting. Even though current LMs show quite limited zero-shot reasoning ability about time series, they still demonstrate *some potential*. Examples in Figure A.6 illustrate some successful cases. Therefore, we believe that with the development of general models, LMs reasoning ability on numerical sequences, especially with natural language context, will gradually improve.

F Participant Details

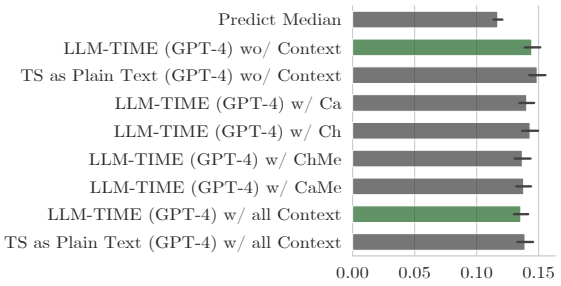
Participants were recruited from a major Computer Science department. They were paid at a rate equivalent to their hourly rate commensurate with their seniority, as determined by the department.



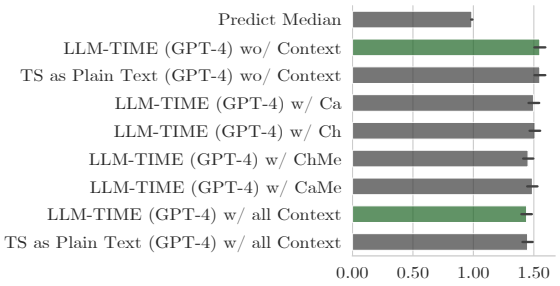
(a) MAE with max-min normalization



(b) MAE with z-score normalization



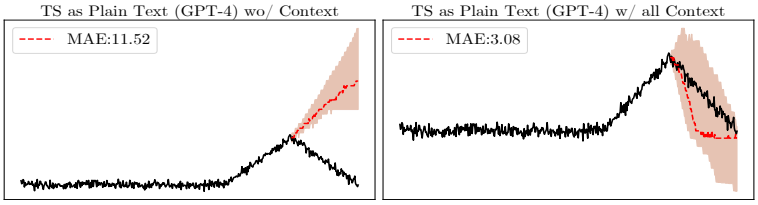
(c) MSE with max-min normalization



(d) MSE with z-score normalization

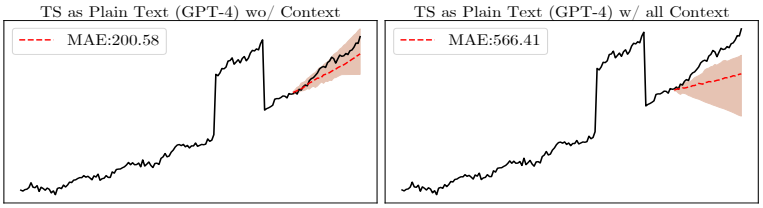
Figure A.4: These figures indicate that after adding various context relevant to the time series, the forecast results improved *marginally*. We use "Predict Median", "LLM-TIME (Gruver et al., 2023) (GPT-4)", and "TS as Plain Text (GPT-4)" as our baselines. In the baseline, LLM forecasts without context (wo/ Context). It can be observed that whether providing Caption (Ca), Characteristics (Ch), or Metadata (Me) individually, such as "LLM-TIME (GPT-4) w/ Ca", or combining all captions, for example, "LLM-TIME (GPT-4) w/ all Context", the overall improvement remains very limited.

Context:
✅ (1) Slow decrease of temperature once power is restored until it reaches ...
✅ (2) Stable temperature values before the outage and after temperature ...
✅ (3) Overall time series has slow ascending and descending trends during ...



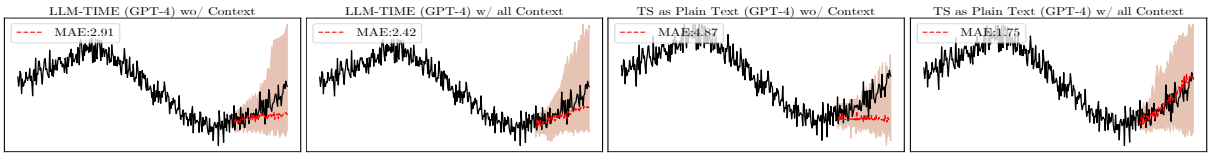
(a) LLM reasoned out the distribution shift in the time series from the captions.

Context:
✅ (1) Overall increasing trend in website visits due to growing popularity.
✅ (2) Daily seasonality due to increased visits during peak hours.
✅ (3) A large spike in visits during the discount event.

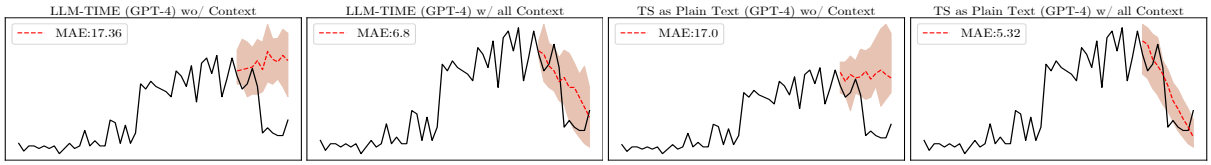


(b) Even in a relatively simple pattern, the LLM fails to effectively understand captions.

Figure A.5: Figures (a) and (b) are two typical examples showing that LLM can reason out difficult-to-forecast distribution shifts from captions. However, in a simple pattern, even when accurate captions are provided, it still fails to reason effectively.



(a) For simple distribution shifting pattern, captions improves reasoning during forecasting.



(b) For very difficult to forecast time series, captions still provide significant help to LLM reasoning.

Figure A.6: Examples (a) and (b) show that integrating captions into forecasting, whether utilizing the LLM-TIME method or directly using GPT-4, helps with LLM reasoning.

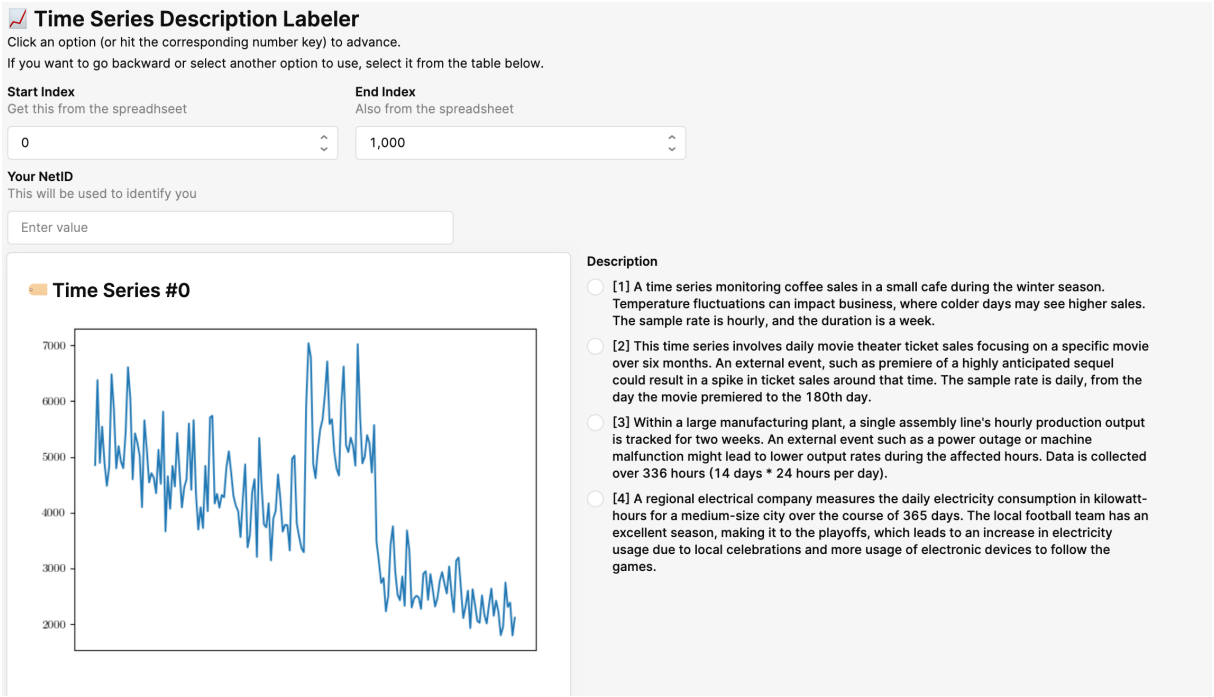


Figure A.7: A screenshot of the tool used by human annotators in the etiological reasoning task (Section 4)