LSTPrompt: Large Language Models as Zero-Shot Time Series Forecasters by Long-Short-Term Prompting

Haoxin Liu[*,†], **Zhiyuan Zhao**[*,†], **Jindong Wang**[§], **Harshavardhan Kamarthi**[†], **B. Aditya Prakash**[†]
[†]Georgia Institute of Technology [§]Microsoft Research Asia
[†]{hliu763, leozhao1997, hkamarthi3, badityap}@gatech.edu [§]jindong.wang@microsoft.com

Abstract

Time-series forecasting (TSF) finds broad applications in real-world scenarios. Prompting off-the-shelf Large Language Models (LLMs) demonstrates strong zero-shot TSF capabilities while preserving computational efficiency. However, existing prompting methods oversimplify TSF as language next-token predictions, overlooking its dynamic nature and lack of integration with state-of-the-art prompt strategies such as *Chain-of-Thought*. Thus, we propose LSTPrompt, a novel approach for prompting LLMs in zero-shot TSF tasks. LSTPrompt decomposes TSF into short-term and long-term forecasting sub-tasks, tailoring prompts to each. LSTPrompt guides LLMs to regularly reassess forecasting mechanisms to enhance adaptability. Extensive evaluations demonstrate consistently better performance of LSTPrompt than existing prompting methods, and competitive results compared to foundation TSF models[1].

1 Introduction

Time-series (TS) data are ubiquitous across various domains, including public health (Adhikari et al., 2019), finance (Deb et al., 2017), and energy (Tay and Cao, 2001). Time-series forecasting (TSF), a crucial task in TS data analysis, aims to predict future events or trends based on historical data. Recent advancements in large Pre-Trained Models (PTMs), a.k.a. foundation models, and Large Language Models (LLMs) have demonstrated their effectiveness for TSF tasks. This is achieved either by training TS foundation models from scratch (Yeh et al., 2023; Kamarthi and Prakash, 2023; Garza and Mergenthaler-Canseco, 2023; Das et al., 2023) or adapting LLMs to TS data as natural language modalities (Jin et al., 2023; Chang et al., 2023; Xue and Salim, 2023; Gruver et al., 2023). These methods leverage powerful generalization capabilities of PTMs or LLMs, proving effectiveness in zero-shot

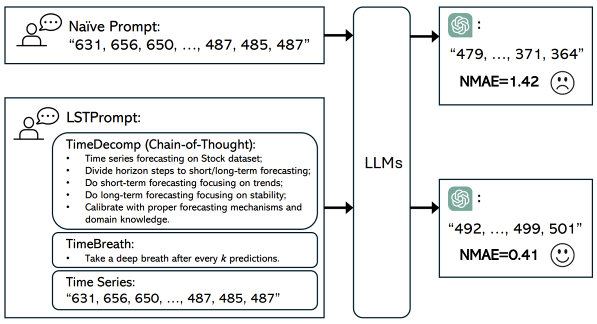
Naïve Prompt:
"631, 656, 650, ..., 487, 485, 487"
LSTPrompt:
TimeDecomp (Chain-of-Thought):
Time series forecasting on Stock dataset;
Divide horizon steps to short/long-term forecasting;
Do short-term forecasting focusing on trends;
Do long-term forecasting focusing on stability;
Calibrate with proper forecasting mechanisms and domain knowledge.
TimeBreath:
Take a deep breath after every k predictions.
Time Series:
"631, 656, 650, ..., 487, 485, 487"
LLMs
:
"479, ..., 371, 364"
NMAE=1.42
:
"492, ..., 499, 501"
NMAE=0.41

Figure 1: Comparison between naive prompt (Gruver et al., 2023) and LSTPrompt.

TSF tasks with promising applications without the need for domain-specific training data.

Designing proper prompting techniques for zero-shot TSF tasks offers notable advantages, which avoids training models from scratch or fine-tuning LLMs for computational efficiency while maintaining forecasting accuracy. Existing approaches (Xue and Salim, 2023; Gruver et al., 2023) prompt LLMs for zero-shot TSF tasks by aligning TS data with natural language sequences and prompting LLMs to perform TSF as sequence completion tasks. However, these methods overlook the dynamic nature of TS data and the intricate forecasting mechanisms inherent in TSF tasks, such as modeling temporal dependencies, which cannot be adequately modeled by simple sequence completion tasks.

To address the limitation, we introduce LSTPrompt, a novel prompt strategy of LLMs for TSF tasks by providing specific TSF-oriented guidelines. Our contributions are summarized as follows:

- We propose Long-Short-Term Prompt (LSTPrompt), which decomposes TSF into short-term and long-term forecasting subtasks. Each subtask guides LLMs with distinct forecasting rules and mechanisms, forming a *Chain-of-Thought* reasoning path for predictions.

- We introduce TimeBreath to LSTPrompt, an innovative component that encourages

[*]Equal Contribution
[1]https://github.com/AdityaLab/lstprompt

LLMs to regularly revisit forecasting mechanisms, enabling leveraging different forecasting mechanisms for different time periods.

- We evaluate LSTPrompt on multiple benchmark and concurrent datasets, demonstrating its effectiveness for zero-shot TSF tasks. We show its generalization ability to outperform non-zero-shot methods in specific scenarios.

We provide additional related works in the Appendix A with distinguishing the differences of popular zero-shot TSF methods in Table 3.

2 Methodology

2.1 Problem Formulation and Motivation

Zero-shot TSF aims to predict future TS $\{\hat{y}_i\}_{i=t}^{t+H}$ with a horizon window size H based on a reference TS $\{y_i\}_{i=t-L}^{t}$ with lookback window size L, without prior exposure or training on the target series. Solving zero-shot TSF tasks with LLMs requires aligning TS data with natural language modalities to leverage remarkable generalization abilities and generate predictions based on the provided context.

One approach to align TS data with LLMs is to present TS data as text. Existing zero-shot TSF prompt strategies (Xue and Salim, 2023; Gruver et al., 2023) represent TS data as strings of numerical digits and treat TSF tasks as text-based next-token predictions. However, these strategies overlook the need for sophisticated forecasting mechanisms inherent in dynamic TS data. Without explicit instructions, existing strategies may yield inaccurate predictions with high uncertainty.

To address this, we propose LSTPrompt, tailored for zero-shot TSF tasks through prompting LLMs informatively. LSTPrompt comprises two components: (1) **TimeDecomp**, decomposing TSF tasks into subtasks for systematic reasoning, and (2) **TimeBreath**, facilitating periodic breaks to adapt forecasting strategies within the horizon window. We detail each module in the subsequent sections.

2.2 TimeDecomp

Rather than directly prompting complex questions to LLMs, recent studies advocate decomposing inquiries into simpler, sequential steps (Wei et al., 2022; Kojima et al., 2022). This approach aids LLMs in constructing a coherent reasoning path. However, applying such chain-of-thought or step-by-step strategies to TSF tasks remains unexplored.

To address this, we introduce TimeDecomp, which breaks down TSF tasks into short-term and long-term forecasting subtasks. This is motivated by different forecasting mechanisms for short/long-term forecasting. Particularly, TimeDecomp prompts LLMs to partition horizon time steps into short-term and long-term accordingly. Then, it guides LLMs through each subtask, directing them to focus on specific aspects: short-term forecasting emphasizes trend changes and dynamic patterns, while long-term forecasting highlights statistical properties and periodic patterns. TimeDecomp's chain-of-thought process follows step-by-step cues: it prompts tasks with specific datasets, decomposes tasks into short-term and long-term sub-tasks, and guides LLMs to incorporate appropriate forecasting mechanisms and domain knowledge.

2.3 TimeBreath

In addition to chain-of-thought prompting, recent studies emphasize the importance of incentivizing LLMs to follow step-by-step reasoning, especially when having numerous subtasks (Zhou et al., 2022b; Yang et al., 2023). To facilitate this, Yang et al. propose a strategy that introduces "Take a deep breath" before initiating step-by-step tasks.

TSF tasks involve varying reasoning across different time steps and overly lengthy forecasting horizons can overwhelm LLMs' reasoning abilities. Inspired by the "deep breath" design, we introduce TimeBreath, which prompts LLMs to take "rhythmic breaths" during sequential reasoning for TSF. In the TSF task with H time steps horizon, TimeBreath guides LLMs to rhythmically breathe every k steps, where k is a hyperparameter determining the breath frequency. The intuition of TimeBreath is to encourage LLMs to reassess forecasting mechanisms regularly, particularly for distant time steps that may require different reasonings. By taking breaks, TimeBreath helps LLMs avoid prior irrelevant inferences and fosters adaptive forecasting mechanisms to current forecasts.

In practice, the choice of k significantly impacts LLMs' performance in zero-shot TSF tasks, as demonstrated in the sensitivity analysis provided in Appendix C. A straightforward approach is to align the frequency of breaks with the upper time scale. For example, setting $k = 5$ prompts weekly breaks for daily stock forecasting, while $k = 4$ encourages monthly breaks for weekly Influenza forecasting.

	Dataset	Frequency	Horizon	Supervised				Zero-Shot (PTMs)	Zero-Shot (Prompt)	
				SP	ARIMA	TCN	N-BEATS	TimesFM	LLMTime	LSTPrompt
Darts	AirPassengers	Month	29	34.67	*24.03*	54.96	97.89	14.75	48.96	**13.02**
	MilkProduction	Month	34	*30.33*	37.19	70.86	33.64	22.46	63.15	**7.71**
	BeerProduction	Season	43	102.05	17.13	30.90	*10.39*	**10.25**	20.85	13.29
	Sunspots	Day	141	53.74	*43.56*	51.82	73.15	50.88	59.91	**46.84**
				DeepAR	N-BEATS	WaveNet	Transformer	TimesFM	LLMTime	LSTPrompt
Monash	RiverFlow	Day	30	23.51	27.92	*22.17*	28.06	24.53	28.63	**24.17**
	US Births	Day	30	424.9	*422.0*	504.4	452.9	**408.5**	459.43	429.2
				Informer	Autoformer	FEDformer	PatchTST	TimesFM	LLMTime	LSTPrompt
Informer (ETT)	ETTh1	Hour	96	0.76	0.55	0.58	*0.41*	0.37	0.42	**0.32**
			192	0.78	0.64	0.64	*0.49*	0.49	0.50	**0.36**
	ETTm1	Minute	96	0.71	0.54	0.41	*0.33*	0.25	0.37	**0.19**
			192	0.68	0.46	0.49	*0.31*	**0.24**	0.71	0.55
	ETTh2	Hour	96	1.94	0.65	0.67	*0.28*	**0.28**	0.33	0.31
			192	2.02	0.82	0.82	*0.68*	0.58	0.70	**0.45**

Table 1: Performance comparison of supervised models and zero-shot methods on benchmark datasets: (1) LSTPrompt achieves mostly the best and several second-best results among zero-shot forecasting methods. (2) LSTPrompt outperforms the best supervised models on 6 out of 12 datasets. We bold the best zero-shot results and LSTPrompt with the second-best results is underlined. We italicize/underline the best supervised results.

Dataset	Frequency	Horizon	Supervised				Zero-Shot (PTMs)	Zero-Shot (Prompt)	
			Informer	AutoFormer	FedFormer	PatchTST	LPTM	LLMTime	LSTPrompt
ILI	Week	4	1.64	1.17	2.31	*0.51*	1.54	0.61	**0.42**
		12	2.25	2.10	1.97	*0.52*	0.83	0.81	**0.67**
		20	2.01	1.43	1.67	*1.39*	1.70	4.68	**1.73**
		24	4.29	1.86	*1.30*	2.15	2.18	4.81	**2.08**
Stock	Day	24	5.07	9.94	8.73	*4.52*	0.73	0.51	**0.32**
		48	8.03	9.22	9.56	*4.11*	0.80	0.42	**0.19**
		96	*3.11*	9.61	9.43	4.36	0.87	1.42	**0.41**
		120	*4.07*	10.92	10.59	4.65	1.28	2.61	**0.52**
Weather	Day	24	1.59	*1.54*	1.77	1.77	0.79	**0.31**	**0.31**
		48	1.62	1.63	1.84	*1.25*	1.06	0.66	**0.53**
		96	1.43	1.50	2.34	*1.16*	1.08	0.84	**0.62**
		120	1.45	1.64	1.95	*1.40*	1.18	0.83	**0.69**

Table 2: Performance comparison of supervised models and zero-shot methods on concurrent datasets: (1) LSTPrompt consistently outperforms zero-shot baselines on all evaluations. (2) LSTPrompt outperforms best supervised models on 9 of 12 evaluations. We bold the best zero-shot method and italicize/underline the best supervised results.

2.4 LSTPrompt

We introduce LSTPrompt, which integrates TimeDecomp and TimeBreath to create the comprehensive prompt strategy. The prompt is straightforward: LSTPrompt first guides LLMs through the chain-of-thought steps outlined by TimeDecomp, then instructs them to take regular breaks using TimeBreath. A LSTPrompt demo is shown by Figure 1. We provide a detailed prompting example in Appendix B. LSTPrompt is designed for any TS datasets for zero-shot TSF tasks. It can be easily tailored to different scenarios by adjusting a single hyperparameter, k, as previously discussed.

3 Experiments

3.1 Benchmark Evaluation

To benchmark the performance of LSTPrompt, we use three common TSF benchmarks: Darts (Herzen et al., 2022), Monash (Godahewa et al., 2021), and Informer datasets (Zhou et al., 2021). While these datasets can potentially be used for training LLMs, evaluating LSTPrompt on these datasets allows fair comparisons within aligned settings, which strictly follows the established setup for zero-shot TSF tasks (Gruver et al., 2023) and are detailed in Appendix C. We use the SOTA prompting method LLMTime (Gruver et al., 2023) and a recent PTM TimesFM (Das et al., 2023) as zero-shot baselines. The results are shown in Table 1. We showcase visualized results in Appendix C.

The results highlight two main benefits of LSTPrompt: First, LSTPrompt achieves the best performance on 8 out of 12 benchmark datasets and the second-best performance on the remaining 4 among zero-shot methods. Notably, LSTPrompt always outperforms the SOTA prompt method LLMTime, while may slightly lag behind TimesFM, which is expected since TimesFM is a TSF-specific PTMs. Second, LSTPrompt can outperform best supervised results under certain scenarios. For instance, LSTPrompt achieves a 74.6% lower MAE compared to the best supervised result on the MilkProduction dataset. This improvement relies on the strong generalization ability of LLMs, which helps mitigate overfitting for supervised models.

3.2 Concurrent Dataset Evaluation

To evaluate the true zero-shot ability of LSTPrompt, we conduct experiments over three concurrent datasets from different domains: influenza-like illness (ILI), Stock, and Weather (Detailed in Appendix C). These datasets ensure that the test data are after June 2023, while most LLMs are trained only up to 2022 (Achiam et al., 2023). Employing these datasets ensures the zero-shot property, even for GPT4. The experiment setup follows Benchmark Evaluations. We omit PromptCast (Xue and Salim, 2023), exclude TimesFM, and include another foundation time-series model, LPTM (Kamarthi and Prakash, 2023), for zero-shot baselines with explanations in Appendix C. We include supervised TSF models, including Informer (Zhou et al., 2021), Autoformer (Wu et al., 2021), FEDformer (Zhou et al., 2022a), and PatchTST (Nie et al., 2022), to show performance disparities between zero-shot methods and supervised models on TSF tasks. The results are shown in Table 2.

The results demonstrate that LSTPrompt consistently outperforms zero-shot baselines on all evaluations. Notably, LSTPrompt consistently outperforms best supervised results on Stock and Weather datasets. This is attributed to heavy distribution drifts on these datasets, which largely degrade the supervised models' performances. In contrast, benefiting from strong generalization abilities of LLMs and zero-shot properties, zero-shot methods mitigate the impacts of distribution drifts and achieve better performance than supervised models.

3.3 Ablation Study

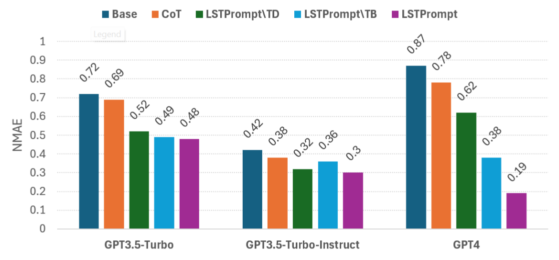

Figure 2: Ablation Study: (1) Enhanced reasoning abilities enable LSTPrompt to perform best on GPT4. (2) Both TimeDecomp and TimeBreath effectively enhance the forecasting accuracy of LSTPrompt.

To understand the significance of various components of LSTPrompt, we conduct two ablation studies: (1) Analyzing the impact of employing different LLMs; (2) Analyzing the effects of TimeDecomp and TimeBreath. We conduct experiments with combinations of different LLMs and various ablated versions of LSTPrompt on the Stock dataset, with results visualized in Figure 2.

Prompting Different LLMs. In prior experiments, we presented forecasting results based on the most suitable LLMs (e.g., GPT3.5-Turbo-Instruct for LLMTime and GPT4 for LSTPrompt). However, performance differences can arise among zero-shot TSF methods, including LSTPrompt, when evaluated across different LLMs. Thus,we investigate and interpret the potential impacts of utilizing GPT3.5-Turbo, GPT3.5-Turbo-Instruct, and GPT4 with LSTPrompt. The results indicate LSTPrompt coupled with GPT4.0 outperforms instances with GPT3.5-Turbo and GPT3.5 Turbo-Instruct. This finding aligns with expectation, as LSTPrompt prompts LLMs to follow the reasoning path through distinct short-term and long-term forecasting subtasks, each requiring different reasoning mechanisms, while GPT4 is known for its reasoning abilities compared to the remaining two.

Module Effectiveness. To understand the significance of TimeDecomp and TimeBreath, we analyze performance discrepancies over three ablated versions of LSTPrompt: (1) Base, using standard prompts; (2) LSTPrompt\TD, excluding TimeDecomp from LSTPrompt; (3) LSTPrompt\TB, excluding TimeBreath from LSTPrompt. We include the state-of-the-art Chain-of-Thought method (Yang et al., 2023) (referred to as 'CoT') to highlight performance differences with the SOTA prompt strategy for general tasks.

The results demonstrate the effectiveness of both TimeDecomp and TimeBreath. Incorporating TimeDecomp and TimeBreath reduces the average NMAE by 26.8% and 34.1%, respectively, compared to Base prompts. Employing both modules enhances average performance by 46.7% than Base prompts. Moreover, the sole utilization of either TimeDecomp or TimeBreath demonstrates certain advantages in forecasting accuracy over the best CoT method, highlighting the necessity of designing tailored prompts for TSF tasks.

4 Conclusion

In this paper, we introduce LSTPrompt, a novel prompt paradigm for zero-shot TSF tasks through prompting LLMs. LSTPrompt enables LLMs to achieve accurate zero-shot TSF tasks through two innovative modules: TimeDecomp, which decomposes zero-shot TSF tasks into a series of chain-of-thought subtasks, and TimeBreath, which encourages LLMs to periodically reassess forecasting mechanisms. Extensive experiments validate the effectiveness of LSTPrompt, which consistently outperforms the SOTA prompt method and shows generally better performance than SOTA PTMs.

Acknowledgements: This paper was supported in part by the NSF (Expeditions CCF-1918770, CAREER IIS-2028586, Medium IIS-1955883, Medium IIS-2106961, PIPP CCF-2200269), CDC MInD program, Meta faculty gifts, and funds/computing resources from Georgia Tech and GTRI.

References

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

Bijaya Adhikari, Xinfeng Xu, Naren Ramakrishnan, and B Aditya Prakash. 2019. Epideep: Exploiting embeddings for epidemic forecasting. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 577–586.

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

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

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

Chirag Deb, Fan Zhang, Junjing Yang, Siew Eang Lee, and Kwok Wei Shah. 2017. A review on time series forecasting techniques for building energy consumption. *Renewable and Sustainable Energy Reviews*, 74:902–924.

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

Deepanway Ghosal, Navonil Majumder, Ambuj Mehrish, and Soujanya Poria. 2023. Text-to-audio generation using instruction-tuned llm and latent diffusion model. *arXiv preprint arXiv:2304.13731*.

Rakshitha Godahewa, Christoph Bergmeir, Geoffrey I Webb, Rob J Hyndman, and Pablo Montero-Manso. 2021. Monash time series forecasting archive. *arXiv preprint arXiv:2105.06643*.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. 2023. Large Language Models Are Zero Shot Time Series Forecasters. In *Advances in Neural Information Processing Systems*.

Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. 2023. Tabllm: Few-shot classification of tabular data with large language models. In *International Conference on Artificial Intelligence and Statistics*, pages 5549–5581. PMLR.

Julien Herzen, Francesco Lässig, Samuele Giuliano Piazzetta, Thomas Neuer, Léo Tafti, Guillaume Raille, Tomas Van Pottelbergh, Marek Pasieka, Andrzej Skrodzki, Nicolas Huguenin, et al. 2022. Darts: User-friendly modern machine learning for time series. *The Journal of Machine Learning Research*, 23(1):5442–5447.

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. 2023. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*.

Harshavardhan Kamarthi, Lingkai Kong, Alexander Rodríguez, Chao Zhang, and B Aditya Prakash. 2022. Camul: Calibrated and accurate multi-view time-series forecasting. In *Proceedings of the ACM Web Conference 2022*, pages 3174–3185.

Harshavardhan Kamarthi and B Aditya Prakash. 2023. Large pre-trained time series models for cross-domain time series analysis tasks. *arXiv preprint arXiv:2311.11413*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 95–104.

Bryan Lim and Stefan Zohren. 2021. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209.

Kevin Lu, Aditya Grover, Pieter Abbeel, and Igor Mordatch. 2021. Pretrained transformers as universal computation engines. *arXiv preprint arXiv:2103.05247*, 1.

Elizbar A Nadaraya. 1964. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142.

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

Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. 2019. N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*.

Francis EH Tay and Lijuan Cao. 2001. Application of support vector machines in financial time series forecasting. *omega*, 29(4):309–317.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Christopher Williams and Carl Rasmussen. 1995. Gaussian processes for regression. *Advances in neural information processing systems*, 8.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430.

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*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2023. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*.

Chin-Chia Michael Yeh, Xin Dai, Huiyuan Chen, Yan Zheng, Yujie Fan, Audrey Der, Vivian Lai, Zhongfang Zhuang, Junpeng Wang, Liang Wang, et al. 2023. Toward a foundation model for time series data. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4400–4404.

G Peter Zhang. 2003. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175.

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

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

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

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022b. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.

A Additional Related Works

Time-Series Forecasting. Traditional time series methods approach forecasting from a statistical standpoint, treating it as standard regression problems with time-varying parameters (Nadaraya, 1964; Williams and Rasmussen, 1995; Zhang, 2003). Recent advancements in deep learning have led to significant breakthroughs in this field, exemplified by deep models like LSTNet and N-BEATS (Lai et al., 2018; Oreshkin et al., 2019). Many state-of-the-art deep learning methods, such as Informer, Autoformer, PatchTST, and CAMul (Zhou et al., 2021; Wu et al., 2021; Nie et al., 2022; Kamarthi et al., 2022), build upon the success of self-attention mechanisms, popularized by transformer-based architectures (Vaswani et al., 2017). These transformer-based models excel at capturing long-range dependencies, surpassing the capabilities of traditional Recurrent Neural Network (RNN) models, owing to their effective use of self-attention mechanisms.

Large Language Models. The augmentation of language model parameters and training data size has been shown to enhance generalization ability (Brown et al., 2020). Consequently, researchers have developed Large Language Models (LLMs) like GPT (Brown et al., 2020; Achiam et al., 2023) and Llama (Touvron et al., 2023). These models excel at identifying patterns in prompts and extrapolating them through next-token prediction, achieving remarkable success in few-shot or zero-shot generalization and in-context learning. Beyond natural language tasks, LLMs exhibit effectiveness in transfer learning across diverse modalities, including images (Lu et al., 2021), audio (Ghosal et al., 2023), tabular data (Hegselmann et al., 2023), and time-series data (Zhou et al., 2023). These accomplishments underscore the importance of aligning modalities appropriately to enable LLMs to comprehend tokenized patterns across different domains beyond traditional language processing tasks.

Large Models for Time-Series Forecasting. In addition to the success of large models in language tasks, researchers in the field of time-series forecasting (TSF) have pursued the development of large models from two main perspectives: First, they train Pre-Trained Time-Series Models from scratch (Garza and Mergenthaler-Canseco, 2023; Das et al., 2023; Kamarthi and Prakash, 2023; Yeh et al., 2023), utilizing extensive time-series datasets and tailoring them specifically for TSF tasks. Alternatively, researchers harness the generalization capabilities of Large Language Models (LLMs) by aligning time-series data with language modalities through techniques such as reprogramming (Jin et al., 2023; Chang et al., 2023; Zhou et al., 2023) or prompting (Gruver et al., 2023; Xue and Salim, 2023). To better understand the similarities and differences between all zero-shot methods mentioned in this work, we list the property comparisons of all zero-shot methods in Table 3.

Method	Type	Cost	Use CoT or Guidelines	Evaluated on GPT4
TimesFM	PTMs	High	N/A	N/A
LPTM	PTMs	High	N/A	N/A
PromptCast	Prompt	Low	No	No
LLMTime	Prompt	Low	No	Partial
LSTPrompt	Prompt	Low	Yes	Yes

Table 3: Summary of similarities and differences of related works on zero-shot TSF tasks.

B Prompt Details

Below, we introduce a template prompt for LSTPrompt, designed to be adaptable to various time-series datasets for zero-shot time-series forecasting tasks. The template is outlined as follows:

```
f"Please continue the following input
    sequence by addressing the task of
    forecasting {dataname}. You should
    break down the task into short-term
    and long-term predictions, following
     a three-step plan. First,
    adaptively and reasonably identify
    the ranges for short-term and long-
    term predictions. Then, design
    distinct and correct forecasting
    mechanisms for both short-term and
    long-term prediction tasks. For
    short-term predictions, focus on
    trends and the last few steps of the
     input sequence. For long-term
    predictions, emphasize cyclical
    patterns and statistical properties
    of the entire input sequence. You
    may further optimize the forecasting
     mechanisms based on your
    observations and domain knowledge.
    Finally, correctly implement the
    forecasting mechanisms, completing
    predictions one-time step at a time.
Remember to take a deep breath after
    every {breath_steps} time steps of
    prediction. The input sequence is as
     follows:\n"
```

C Additional Experiment Details

Experiment Setup. Following the established setups in LLMTime and with the consideration of evaluating costs, we limit our focus to univariate time series forecasting tasks. However, LSTPrompt can readily extend to the multivariate forecasting domain by employing multiple univariate forecasting techniques (Gruver et al., 2023; Lim and Zohren, 2021). We strictly followed LLMTime's data-splitting and data-preprocessing method for benchmark datasets, where the test set comprises the last 20% of each time series.

In addition to well-known benchmark datasets such as Darts, Monash, and ETT, our zero-shot evaluations encompass three concurrent datasets: ILI, Stock, and Weather. This selection ensures that the test data have never been exposed to LLMs training. All these datasets are publicly accessible. We use data after June 2023 for testing, thereby guaranteeing that GPT-3.5 and GPT-4 models have not been trained on these sets. Further details on these datasets are provided below:

- **ILI[2]:** The ILI dataset provides the reported influenza-like illness patients with age divisions. The dataset covers from 2002 to 2023. The forecasting target is the weekly number of ILI patients.
- **Stock[3]:** The Stock dataset provides daily historical data of Alphabet Inc. (GOOG). The Stock dataset set has 7 columns, including the stock's opening price, closing price, highest price of the day, etc. The dataset covers from 2013 to 2024 (Jan). The forecasting target is the daily opening price.
- **Weather[4]:** The Weather dataset provides historical weather record of Chicago. This dataset set has 10 columns, including date, temperature, precipitation, humidity, wind speed, and atmospheric pressure. The dataset covers from 2021 to 2023. The forecasting target is the daily average temperature.

Baseline. In supervised baselines, we adopt various models depending on the benchmark's official evaluation criteria. Concurrent datasets utilize transformer-based supervised models, same as the ETT benchmark, known for their remarkable performance in TSF evaluations.

For zero-shot baselines, we categorize methods into pre-trained Time-Series Foundation Models (PTMs) and prompting methods. In benchmark evaluations, we utilize TimesFM (Das et al., 2023) for PTMs, as it asserts not being trained on these datasets, while LPTM (Kamarthi and Prakash, 2023) does. Conversely, for concurrent dataset evaluations, we employ LPTM, as it is open-source compared to TimesFM. Reprogramming methods are omitted, such as TimeLLM (Jin et al., 2023) and LLM4TS (Chang et al., 2023), due to their inapplicability to our zero-shot setting.

For prompting methods, we compare LLMTime (Gruver et al., 2023) with our proposed LSTPrompt. Promptcast (Xue and Salim, 2023) is omitted, as LLMTime consistently outperforms it, and LSTPrompt demonstrates uniformly better performance across all evaluations than LLMTime. The implementation of our method was inspired by the prompt trick in TimeLLM (Jin et al., 2023) and filtered outliers of predictions using standard deviation.

Evaluation Metric. Following the established setups, we evaluate the Mean Absolute Error (MAE) on Darts and Monash datasets between predictions and raw target sequences. For ETT, ILI, Stock, and Weather datasets, we evaluate the MAE based on the normalized predictions and target sequences according to the mean and variance of the training data. The formulation of MAE $= \frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t|$.

Hyperparameter Sensitivity Study. As previously mentioned in Section 2, we conduct experiments using LSTPrompt on the Stock dataset with varying values of breath frequency k. The results are shown in Fiugure 3. Note that $k = 0$ denotes LSTPrompt without employing TimeBreath.

The results suggest that setting $k = 5$, enabling LSTPrompt to breathe weekly in forecasting the stock prices, achieves the best performance compared to other breath frequencies. This optimal frequency aligns with the Stock dataset's structure, which includes daily stock prices for 5 weekdays. Intuitively, setting $k = 5$ encourages LSTPrompt to reassess its reasoning and forecasting strategy on a weekly basis, fitting well with the inherent weekly cycles in stock data. By appropriately adjusting the breath frequency in TimeBreath, LSTPrompt can

[2] https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

[3] https://www.kaggle.com/datasets/jillanisofttech/google-10-years-stockprice-dataset

[4] https://www.kaggle.com/datasets/curiel/chicago-weather-database

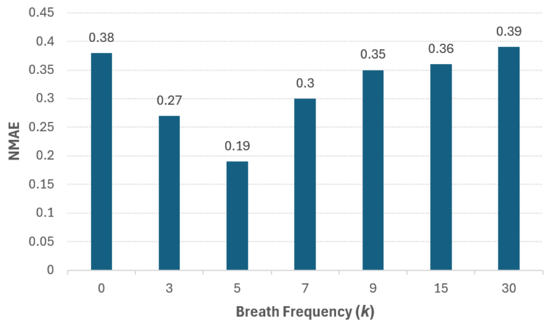

Figure 3: Hyperparameter Sensitivity: The best breath frequency $k = 5$ (weekly) aligns with the upper time scale of the Stock data (daily).

dynamically infer patterns while effectively adapting to the data's periodic nature, leading to more accurate forecasts.

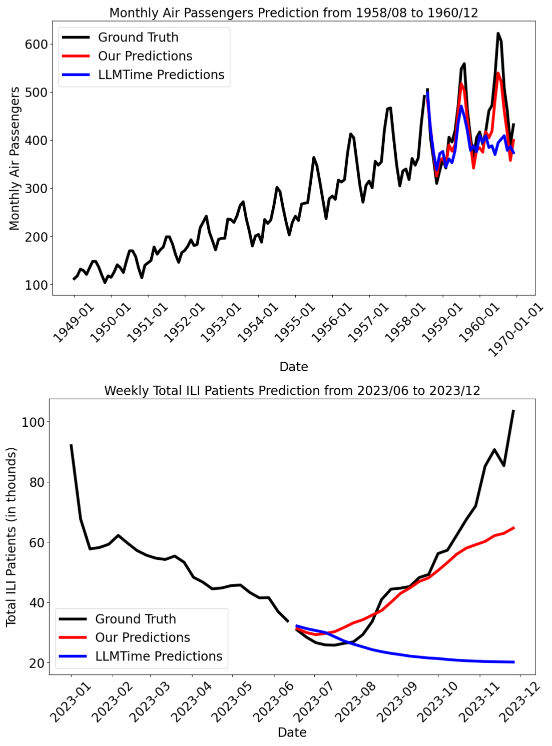

Figure 4: Result visualizations on the AirPassengers (top) and ILI (bottom) datasets. LSTPrompt exhibits better performance than LLMTime, demonstrating enhanced long-term prediction stability and improved ability to capture trend changes.

Result Visualization. We present the result visualizations for the AirPassengers dataset in the Benchmark Evaluation and the ILI dataset in the Concurrent Dataset Evaluation. These visualizations are shown in Figure 4.

The visualizations demonstrate clear benefits from two perspectives: First, the predictions of LSTPrompt exhibit greater long-term stability and accuracy compared to LLMTime, as evidenced by the AirPassengers predictions. Notably, LSTPrompt effectively maintains the periodic properties inherent in the dataset. Second, LSTPrompt demonstrates better capability in capturing accurate trend changes compared to LLMTime, as illustrated by the ILI predictions. In particular, LSTPrompt accurately captures trends in increasing predictions where LLMTime fails to detect them.

D Limitation Discussion

While LSTPrompt has demonstrated effectiveness in zero-shot TSF tasks by employing simple prompts for LLMs, its limitations should be acknowledged from two perspectives. First, the interpretability of LSTPrompt may be compromised. The evaluation of LSTPrompt heavily relies on existing LLMs, the mechanisms and response behaviors of which are currently challenging to interpret. Consequently, LSTPrompt may suffer from reduced interpretability due to our limited understanding of LLMs. Second, incorporating additional instructions in the prompts, such as the names and properties of time-series datasets, could potentially introduce information leaks that are exploited by the LLMs. We advocate for further research within the safe AI community to investigate the trustworthiness of LLMs, ensuring that LSTPrompt can be deployed without concerns regarding information leakage issues.