FLAIRR-TS – Forecasting LLM-Agents with Iterative Refinement and Retrieval for Time Series

Anonymous ACL submission

Abstract

Time series Forecasting with large language models (LLMs) requires bridging numerical patterns and natural language. Effective forecasting on LLM often relies on extensive preprocessing and fine-tuning. Recent studies show that a frozen LLM can rival specialized forecasters when supplied with a carefully engineered natural-language prompt, but crafting such a prompt for each task is itself onerous and ad-hoc. We introduce FLAIRR-TS, a test-time prompt optimization framework that utilizes an agentic system: a Forecaster-agent generates forecasts using an initial prompt, which is then refined by a refiner agent, informed by past outputs and retrieved analogs. This adaptive prompting generalizes across domains using creative prompt templates and generates high-quality forecasts without intermediate code generation. Experiments on benchmark datasets show FLAIRR-TS improves forecasting over static prompting and retrieval-augmented baselines, approaching the performance of specialized prompts. FLAIRR-TS provides a practical alternative to fine-tuning, achieving strong performance via its agentic approach to adaptive prompt refinement and retrieval.

1 Introduction

LLMs can, in principle, leverage their vast pretrained knowledge for prediction tasks. Initial studies demonstrated that direct prompting could enable LLMs to achieve competitive zero-shot or few-shot forecasting performance compared to some specialized models, particularly in novel scenarios (Xue and Salim, 2023).

However, the efficacy of LLMs in time series forecasting (TSF) is often stymied by the **prompt engineering bottleneck**. The performance of a frozen, pre-trained LLM is critically dependent on the precise natural language prompt it receives. Crafting optimal prompts is currently a laborious, ad-hoc process requiring significant domain expertise and iterative manual tuning for each new dataset or scenario, thereby limiting scalability and robust generalization((Niu et al., 2024)). This challenge has spurred research into more sophisticated prompting strategies (Liu et al., 2024; Tang et al., 2024) and even methods to reprogram LLMs at inference time without altering weights (Jin et al., 2024).

Given that LLMs can iteratively refine their outputs through feedback (as demonstrated by Madaan et al. (2023) and Chen and others (2025)), we explore their capability to autonomously refineing their prompts at test time to enhance time series forecasts.

We introduce **FLAIRR-TS** - **F**orecasting **L**LM-**A**gents with **I**terative **R**efinement and **R**etrieval, a framework designed to enhance TSF capabilities of LLMs without any training. This approach aims to mitigate the manual prompt engineering burden while simultaneously improving prediction accuracy by grounding forecasts in relevant historical context. FLAIRR-TS synergistically integrates a **Forecaster-agent** (F) for initial predictions, a **Refiner Agent** for Iterative RefinementTuning (IRT), and a **Retrieval agent (R)** that sources semantically similar historical time series segments, akin to Retrieval Augmented Generation (RAG) principles adapted for TSF (Han et al., 2023). This entire cycle of prompt adaptation and forecast refinement occurs without any model weight updates, offering a compelling alternative to costly fine-tuning.

Beyond the adaptive capabilities of FLAIRR-TS for general applicability, we also investigate the upper bounds of performance achievable with highly engineered instructions. To this end, we introduce **Architected Strategy Prompts (ASPs)**: a set of specialized prompts, which include directives for specific analytical procedures or induce particular cognitive approaches. These are developed through a Systematic Prompt Architecting process inspired

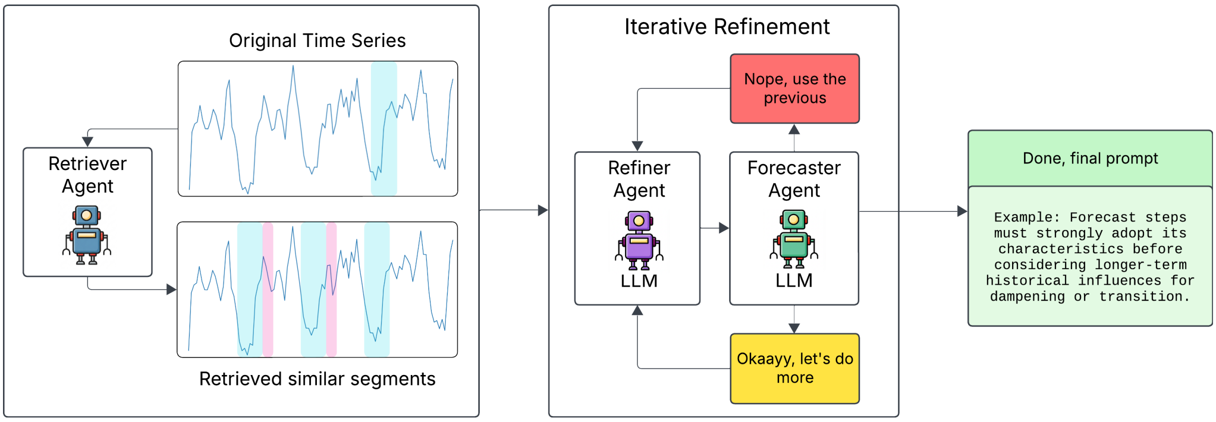

Figure 1: Flowchart of the the proposed method framework, consisting Retrieval, Forecaster and Refiner agents.

by (Sahoo et al., 2025). While FLAIRR-TS excels at automated, test-time prompt refinement without prior domain-specific tuning, ASPs allow us to explore the pinnacle of performance when such meticulous, strategy-driven design is employed. Our main contributions are summarized as:

- We propose **FLAIRR-TS**, a novel prompting and test-time optimization framework for TSF with iterative refinement and retrieval.
- We utilize retrieval augmentation for TSF with LLMs with the introduced **Architected Strategy Prompts (ASPs)**, developed via a Systematic Prompt Architecting process, to reveal the significant impact of specialized, meticulously- engineered instructions and to serve as high-performance benchmarks.
- We demonstrate that FLAIRR-TS consistently improves forecasting accuracy across diverse datasets without model fine-tuning, outperforming static domain agnostic prompting and a non-iterative retrieval-augmented baseline

2 Methodology

2.1 Overall Agentic Architecture

We propose FLAIRR-TS, a framework combining test-time optimization for iterative refinement via prompting by an agentic system, and retrieval-augmented context to enhance TSF with pre-trained LLMs.

It is illustrated in Figure 1 and formally detailed in Algorithm 1, operates as a multi-agent system. The **Forecaster Agent** generates predictions using a prompt that is dynamically improved by the **Refiner Agent** during an **Iterative Tuning** phase. This process is enriched by the **Retrieval Agent** that provides the relevant historical context and augments it to the input provided to the forecaster.

The core iterative cycle (Alg. 1, lines 7-20) involves forecasting, evaluating the forecast against recent ground truth (e.g., via a metric like MSE), and refining the prompt. The Refiner agent can signal early termination if the forecasts are satisfactory. Otherwise, if maximum iterations (N_{iter}) are reached, the system defaults to the prompt that yielded the best observed MSE. This adaptive optimization occurs at test-time without any model training.

2.2 Core Agent Descriptions

Retrieval Agent. Inspired by RAFT (Han et al., 2023), this agent (Alg. 1, line 8) enhances the Forecaster Agent's inputs by retrieving M historical time series segments (S_{retr}) that are most similar to the current context window (X_{Ctx}). These segments, along with their actual outcomes, provide illustrative examples of past pattern evolutions, directly augmenting the context (C_{aug}) given to the Forecaster-agent.

Refiner-agent (R). Functioning as a meta-optimizer (Alg. 1, line 14), the Refiner Agent analyzes the Forecaster Agent's most recent output ($\hat{X}_{\text{cand}}$), its calculated error (mae_{curr}), the prompt (P_{curr}) that generated it, and other contextual information. Based on this, it proposes a refined candidate prompt (P_{next}) and provides a done_signal if the current forecast quality meets termination criteria. Its detailed reasoning, guided by a specific prompt structure (see Appendix B), might yield feedback such as, ‘`Pay closer attention to sudden changes in the last 10% of the input sequence`‘

Algorithm 1 FLAIRR-TS Algorithm

Require: Training data X, Historical series $X_{1:t-1}$, Horizon H, Initial prompt P_0, Context length L, #Segments M, Max iterations N_{iter}, Recent ground truth $X_{t:t+H}$

Ensure: Selected prompt P_{out}

1: $P_{\text{curr}} \leftarrow P_0$; $P_{\text{best}} \leftarrow P_0$; $\text{mae}_{\text{min}} \leftarrow \infty$; $\hat{X}_{\text{best}} \leftarrow$ nil; teacher_stopped $\leftarrow$ false
2: $X_{\text{HistDB}} \leftarrow X_{1:t-L-1}$; $X_{\text{Ctx}} \leftarrow X_{t-L:t}$ ▷ Setup context and historical DB
3: **for** $k \leftarrow 1$ **to** N_{iter} **do**
4: $\quad S_{\text{retr}} \leftarrow \textsc{RetrieveSegments}(X_{\text{HistDB}}, X_{\text{Ctx}}, M)$
5: $\quad C_{\text{aug}} \leftarrow \textsc{AugmentContext}(X_{\text{Ctx}}, S_{\text{retr}})$
6: $\quad \hat{X}_{\text{cand}} \leftarrow \textsc{ForecasterLLM}(P_{\text{curr}}, C_{\text{aug}}, H)$
7: $\quad \text{mae}_{\text{curr}} \leftarrow \textsc{CalculateMAE}(\hat{X}_{\text{cand}}, X_{t:t+H})$
8: $\quad$ **if** $\text{mae}_{\text{curr}} < \text{mae}_{\text{min}}$ **then**
9: $\qquad \text{mae}_{\text{min}} \leftarrow \text{mae}_{\text{curr}}$; $P_{\text{best}} \leftarrow P_{\text{curr}}$; $\hat{X}_{\text{best}} \leftarrow \hat{X}_{\text{cand}}$
10: $\quad$ **end if**
11: $\quad (P_{\text{next}}, \text{done\_signal}) \leftarrow \textsc{RefinerLLM}(P_{\text{curr}}, X_{\text{Ctx}}, S_{\text{retr}}, \hat{X}_{\text{cand}}, \text{mae}_{\text{curr}})$
12: $\quad$ **if** done_signal **then**
13: $\qquad P_{\text{out}} \leftarrow P_{\text{curr}}$; teacher_stopped $\leftarrow$ true; **break**
14: $\quad$ **end if**
15: $\quad P_{\text{curr}} \leftarrow P_{\text{next}}$
16: **end for**
17: **if not** teacher_stopped **then** ▷ Fallback to best MAE if max iterations reached
18: $\quad P_{\text{out}} \leftarrow P_{\text{best}}$
19: **end if**
20: **return** P_{out}

Forecaster-agent (F). This agent (Algorithm 1, line 10) is responsible for generating the time series forecast ($\hat{X}_{\text{cand}}$). It uses the current prompt (P_{curr}) either the initial prompt P_0 or the one refined by the Refiner Agent - along with the augmented context (C_{aug}) provided by the Retriever Agent. FLAIRR-TS allows utilization of a potentially more compact LLMs as this agent, with the behaviors shaped by dynamically optimized prompts. The structure of the prompts are detailed in Appendix C.

2.3 Architected Strategy Prompts (ASP)

Analytical

Deep STL analysis (inspired by (Zhou et al., 2024)): perform an STL decomposition, forecast each component, then recombine them via STL addition.

Thinking–Inductive

Monte-Hall Prompting: frame forecasting as a decision game so the model evaluates several scenarios before committing.

Imaginative

(a) *Many-Worlds Reasoning*: simulate multiple plausible futures and aggregate them.
(b) *D&D Dungeon-Master*: forecast a character's hit-point trajectory over upcoming turns.

3 Experiments

Experiments utilized Informer (Zhou et al., 2021) benchmark datasets[1]: ETT (ETTh1, ETTh2, ETTm1, ETTm2); Electricity; Traffic. We also benchmark some newer dataset ; Weather and ILINet and we test on 2025 data after the knowledge cutoff date of Gemini. More details in Appendix - F. All dataset characteristics (domains, frequencies, evaluated horizons H) and Data integrity are detailed in Section F.

LLM Backbone: FLAIRR was run on Gemini-2.5-Pro and ASP was run on Gemini-2.5-Pro and Gemini-2-Flash, both frozen. For ablation, we also ran the same experiments on DeepSeek-V3

Data & Execution: Inputs normalized via standard scaling; prompt numerical precision controlled. Results are median of $p \approx 5$ runs per experiment for robustness.

[1]Full experimental parameters and any dataset-specific preprocessing are in Appendix or supplementary material.

Dataset	Horizon	Supervised				PTMs		Prompt			
		Informer	DLinear	FEDformer	PatchTST	TTM	Time-LLM	LSTP	FLAIRR (Ours)	ASP(G2.5P) (Ours)	ASP(G2.0F) (Ours)
ETTh1	96	0.76	0.39	0.58	0.41	0.36	0.46	0.15	0.101	**0.078**	0.118
	192	0.78	0.41	0.64	0.49	0.39	0.54	0.22	0.246	**0.208**	0.223
ETTh2	96	1.94	0.35	0.67	0.28	0.26	0.40	0.42	0.156	**0.154**	0.197
	192	2.02	0.41	0.82	0.68	**0.32**	0.42	0.48	0.439	0.332	0.416
ETTm1	96	0.71	0.34	0.41	0.33	0.32	0.38	0.10	0.068	0.043	**0.042**
	192	0.68	0.36	0.49	0.31	0.35	0.46	0.21	0.083	**0.081**	0.099
ETTm2	96	0.36	0.26	0.20	0.26	0.17	0.25	0.25	0.108	0.096	**0.093**
	192	0.52	0.30	0.25	0.29	**0.22**	0.29	0.54	0.370	0.255	0.257
electricity	96	0.53	0.24	0.42	0.22	**0.15**	0.22	0.41	0.250	0.245	0.321
	192	0.62	0.25	0.47	0.24	**0.18**	0.24	0.55	0.263	0.259	0.308
traffic	96	0.69	0.28	0.56	0.25	0.46	0.25	0.32	0.145	**0.143**	0.184
	192	0.58	0.28	0.58	0.26	0.49	**0.25**	0.31	0.326	0.324	0.296

Table 1: Performance comparison (MAE) of supervised models and zero-shot methods on benchmark datasets. FLAIRR (Ours), ASP(G2.5P) (Ours), and ASP(G2.0F) (Ours) are our proposed/evaluated methods.

Dataset	Horizon	Supervised				Prompt			
		Informer	AutoFormer	FedFormer	PatchTST	LSTP	FLAIRR (Ours)	ASP(G2.5P) (Ours)	ASP(G2.0F) (Ours)
ILI	4	1.54	1.24	2.54	0.43	0.38	0.271	**0.264**	0.189
	12	2.33	1.82	2.67	0.43	0.39	0.249	**0.183**	0.197
	20	2.12	1.90	1.75	1.26	0.73	0.589	**0.564**	0.867
	24	3.99	1.79	1.50	1.72	1.55	0.724	**0.722**	1.004
Weather	24	1.45	1.38	1.95	1.55	0.17	0.110	**0.084**	0.125
	48	1.57	1.43	1.67	1.56	0.24	0.160	**0.142**	0.238
	96	1.48	1.67	1.96	1.12	0.39	0.29	**0.257**	0.243
	120	1.90	1.74	2.02	1.31	0.51	0.383	0.309	**0.369**

Table 2: Performance comparison (MAE) on datasets whose test periods post-date the Gemini 2.5 Pro knowledge cut-off. FLAIRR and both ASP variants are ours; *Informer–PatchTST* are supervised baselines; *LSTP* is a prior prompt-based method.

Results are in Table 1 - which is the evaluation of long horizon datasets Figure 2 - short horizon. We use Mean Absolute error (MAE) as the main metric. We compare with most recent prompt method of LSTPrompt (Liu et al., 2024)(Frozen Gemini as backbone) and two best PTM methods - TTM (Ekambaram et al., 2024) and Time-LLM (Jin et al., 2024). We also compare against non LLM supervised methods like DLinear (Zeng et al., 2022).

Analysis: Our method (FLAIRR and ASP) performed better than LSTP in all of the datasets used, it performed best among all the models in 14 out of 20 times with performing best in all of the smaller horizon cases.

3.1 Ablations

We disentangle the impact of *Retrieval* and *Iterative Refinement (IR)* by successively activating them on top of a *Simple Prompt*. Fig 2 reports mean absolute error (↓) on ETTm2 for Gemini 2.5 Pro, Gemini 2 Flash, and open-source DeepSeek-V3.

Observations: Retrieval alone lowers error by grounding forecasts in analogous history, while IT alone refines outputs through on-the-fly prompt correction. Their combination (**FLAIRR-TS**) delivers the lowest MAE across all three backbones. Crucially, the same trend holds for DeepSeek-V3, demonstrating that our gains are architecture-agnostic and not specific to the Gemini family of models.

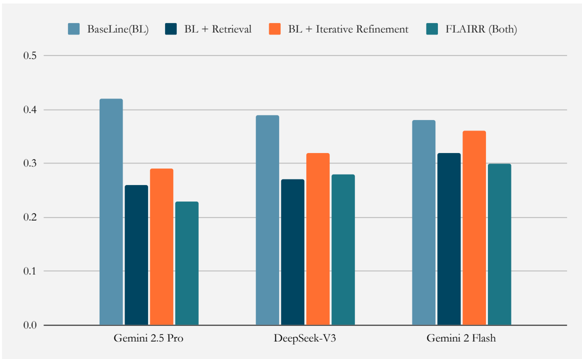

Figure 2: Ablation results, average MAE. Lower MAE is better.

4 Conclusion

The value proposition of FLAIRR-TS lies not necessarily in always surpassing the absolute best, potentially laboriously hand-tuned prompt for every single scenario, but in its ability to automate the refinement process and consistently achieve strong performance starting from generic or moderately good prompts. By iteratively improving instructions based on feedback, FLAIRR-TS aims to elevate the performance baseline achievable with LLMs for TSF without requiring exhaustive manual search for the "perfect" prompt for each new dataset or horizon. The framework offers a pathway to robust performance by adapting the prompt to the task at hand through its agentic interactions.

4.1 Limitations

- **Benchmark coverage.** Empirical validation spans only a handful of public, mostly regularinterval datasets; robustness to irregular sampling, regime shifts, or domain drift remains untested.

- **Analogue-retrieval assumption.** FLAIRR-TS presumes the presence of semantically similar historical segments; when none exist (e.g. novel events), the refinement loop can compound error rather than correct it.
- **Numerical fidelity of LLMs.** Gemini-class models exhibit limited precision on long or out-of-range sequences, and may hallucinate trends under noise or scale shifts, constraining reliability.
- **Inference cost.** Iterative prompting adds multiple LLM calls per forecast; while cheaper than fine-tuning, latency and energy consumption may be prohibitive for real-time, high-frequency settings.

References

Zhaofeng Chen and others. 2025. SETS: Self-verification and self-correction for improved test-time scaling. Anticipated for International Conference on Machine Learning (ICML). Placeholder entry. Details may need updating upon actual publication. Search for preprint by Zhaofeng Chen on SETS.

Vijay Ekambaram, Arindam Jati, Pankaj Dayama, Sumanta Mukherjee, Nam H. Nguyen, Wesley M. Gifford, Chandra Reddy, and Jayant Kalagnanam. 2024. Tiny time mixers (ttms): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series. *Preprint*, arXiv:2401.03955.

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 36 (NeurIPS 2023)*, pages 24013–24034. ArXiv:2310.07820.

Seungone Han, Peiyuan Liao, Poming P. Chiu, Jennifer Hobbs, Sungtae An, Min hwan Oh, Vikas K. Garg, Caiming Xiong, and Yoonkey Kim. 2023. Retrieval augmented time series forecasting. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, pages 73654–73670. ArXiv:2310.16227.

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 *The Twelfth International Conference on Learning Representations (ICLR)*. ArXiv:2310.01728.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, pages 9459–9474.

Haoxin Liu, Zhiyuan Zhao, Jindong Wang, Harshavardhan Kamarthi, and B. Aditya Prakash. 2024. LSTPrompt: Large language models as zero-shot time-series forecasters by long-short-term prompting. In *Findings of the Association for Computational Linguistics: ACL 2024*, Bangkok, Thailand. Association for Computational Linguistics.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-Refine: Iterative refinement with Self-Feedback. arXiv preprint arXiv:2303.17651. *Preprint*, arXiv:2303.17651.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. A time series is worth 64 words: Long-term forecasting with transformers. *Preprint*, arXiv:2211.14730.

Peisong Niu, Tian Zhou, Xue Wang, Liang Sun, and Rong Jin. 2024. Understanding the role of textual prompts in llm for time series forecasting: an adapter view. *Preprint*, arXiv:2311.14782.

Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2025. A systematic survey of prompt engineering in large language models: Techniques and applications. *Preprint*, arXiv:2402.07927.

First Name Tan and others.

Jingyi Tang, Zongyao Zhang, Daksh Minhas, Chengzhang Li, Haomin Chen, Minghuan Tan, Chetan Shah, and Joyce C. Ho. 2024. Prompting medical large vision-language models to diagnose pathologies by visual question answering. arXiv preprint arXiv:2407.21368. *Preprint*, arXiv:2407.21368. Verified from.[53] Placeholder ‘tang-etal-2024-enrichingprompts‘ resolved.

Yu-Hsiang Lin Wan, Akshita Agrawal, Chiyu Max Jiang, Eunsol Choi, and Graham Neubig. 2024. Self-supervised prompting for cross-lingual in-context learning in low-resource languages. arXiv preprint arXiv:2406.18880. *Preprint*, arXiv:2406.18880. Verified from.[55] Placeholder ‘wan-etal-2024-incontextexemplars‘ resolved.

Hao Xue and Flora D. Salim. 2023. Prompt-Cast: A new prompt-based learning paradigm for time series forecasting. volume 36, pages 6851–6864. IEEE. Early access 2023, final publication 2024. arXiv:2210.08964, KDD 2022 version: 10.1145/3534678.3539179.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2022. Are transformers effective for time series forecasting? *Preprint*, arXiv:2205.13504.

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 Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI '21)*, volume 35, pages 11106–11115. AAAI Press.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. 2022. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proceedings of the 39th International Conference on Machine Learning (ICML 2022)*, volume 162 of *Proceedings of Machine Learning Research*, pages 27268–27286. PMLR.

Wendi Zhou, Xiao Li, Lin Geng Foo, Yitan Wang, Harold Soh, Caiming Xiong, and Yoonkey Kim. 2024. TEMPO: Temporal representation prompting for large language models in time-series forecasting. arXiv preprint arXiv:2405.18384. Anticipated for NeurIPS 2024. *Preprint*, arXiv:2405.18384.

A Related Work

Time Series Forecasting with LLMs: Traditional time series forecasting has relied on models explicitly trained for the task, from statistical methods to deep architectures like RNN variants and temporal CNNs, up through recent Transformer-based models (e.g. FEDformer (Zhou et al., 2022) and PatchTST ((Nie et al., 2023))) tailored for long-range sequences. These approaches require substantial training on each target dataset. In contrast, emerging research explores using pre-trained LLMs as general-purpose forecasters via prompting at inference time only, without gradient-based fine-tuning. Xue and Salim (2023) pioneered this direction with PromptCast, formulating forecasting as a prompt–completion task: historical values are encoded into a textual prompt (possibly with instructions) and the LLM's next-token predictions are decoded as forecasts. Gruver et al. (2023) similarly represent numerical time series as token sequences and treat extrapolation as language modeling, finding that GPT-3 and LLaMA-2 can zero-shot extrapolate time series with accuracy comparable to or exceeding specialized trained models. TNotably, these LLM-based approaches leverage the models' strong sequence modeling and few-shot generalization for competitive benchmark results, without requiringabilities to achieve competitive results on standard benchmarks without any task-specific training data. Nevertheless, naive prompt formulations might overlook important temporal dynamics and patterns. Recent works therefore propose more advanced test-time prompting strategies. Liu et al. (2024) introduce LST-Prompt, which splits the prediction into short- and long-term sub-tasks and guides the LLM through a chain-of-thought reasoning process; this method outperforms earlier prompt baselines and even approaches the accuracy of dedicated TS models. Tang et al. (2024) report that enriching prompts with external knowledge (e.g. known seasonal periods or contextual clues) and using natural language rephrasings of the input can significantly improve an LLM's forecasting accuracy. Another technique, Time-LLM (Jin et al., 2024), reprograms a frozen LLM by mapping time-series data into textual "patches" and prepending learned prompt tokens, allowing the model to output forecasts that outperform state-of-the-art specialized forecasters without any fine-tuning of the LLM's weights. On the other hand, Tan and others offer a cautionary perspective: through extensive ablations, they found that removing the LLM or replacing it with a simple attention-based network in these pipelines often does not hurt performance (and sometimes improves it), calling into question how much current LLM-for-TS methods truly benefit from the pre-trained language model. To push LLM-based forecasting further, researchers are drawing on insights from prompt optimization and test-time reasoning. For example, Wan et al. (2024) show that intelligently selecting and reusing in-context exemplars can yield larger gains than optimizing instructions alone, suggesting that careful few-shot prompt design is crucial. Chen and others (2025) propose a self-verification and self-correction framework (SETS) that lets the model iteratively refine its outputs at inference, achieving better accuracy scaling on complex reasoning tasks. Incorporating such techniques into zero-shot forecasting prompts is an exciting direction. In summary, the literature demonstrates a nascent but growing paradigm of using pre-trained LLMs directly for time series forecasting, with multiple studies showing that, given the right prompts, foundation models can attain forecast accuracy rivaling traditional specialized models. While these methods demonstrate progress in leveraging LLMs for forecasting, the dynamic and optimal design of prompts—especially those needing to integrate complex reasoning, external knowledge, and iterative feedback—remains a key challenge. Our work, FLAIRR-TS, aims to address this by structuring the forecasting process around specialized agents for dynamic prompt adaptation and refinement.

Agentic Frameworks with Iterative Refinement The concept of employing multiple interacting agents or distinct processing roles for complex problem-solving has gained traction in AI. Such agentic systems can distribute tasks, specialize functionalities, and enable more sophisticated reasoning or generation processes. Iterative refinement, where an output is progressively improved through feedback loops, is a common characteristic of these systems and is also seen in self-correction mechanisms within single LLMs (e.g., Self-Refine by Madaan et al. (2023)). For instance, systems might involve a generator agent and a critic agent, or distinct agents for planning, execution, and verification. FLAIRR-TS draws inspiration from these paradigms by structuring its operation around specialized agents: a Forecaster-agent for initial prediction, a retriever agent for sourcing relevant context, and a refiner agent for iterative prompt refinement. This agentic decomposition facilitates more targeted and adaptable modifications to distinct aspects of the forecasting prompt through these specialized roles. Crucially, unlike traditional multi-agent systems where agents might be independently trained or involve complex coordination protocols, FLAIRR-TS implements these roles using LLMs at test time to dynamically adapt the prompting strategy itself. The "refinement" occurs in the textual instructions and contextual information fed to the LLM, rather than through updates to model weights, distinguishing it from model distillation or training paradigms. This focus on inference-time prompt adaptation through an agentic perspective is a key aspect of our approach. This structured approach also aims to ensure that the LLM's reasoning and generative capabilities are a core component of the forecasting process, addressing concerns about their actual contribution in some prior LLM-for-TS pipelines.

Retrieval Augmented Generation: Retrieval Augmented Generation (RAG) (Lewis et al., 2020) has become a standard technique for enhancing LLMs in knowledge-intensive NLP tasks. RAG systems retrieve relevant documents or passages from an external corpus and provide them as additional context to the LLM, improving factual grounding and reducing hallucination. Recently, Han et al. (2023) adapted this concept to time series forecasting with their Retrieval Augmented Time Series Forecasting (RAFT) approach. RAFT retrieves historical time series segments similar to the current input window and uses them to augment the context provided to a forecasting model (in their case, an LLM). Our work directly builds upon and integrates the RAFT principle within the Retrieval agent component of FLAIRR-TS. We hypothesize that the effectiveness of RAFT can be further enhanced by optimizing the prompt that instructs the LLM on how to utilize the retrieved historical context, which is precisely what the agentic interaction within FLAIRR-TS aims to achieve.

B Refiner Agent

> You are an expert Time-Series-Forecasting Prompt Engineer acting as a "Teacher LLM". Your goal is to analyse a set of forecasting attempts made by a "Student LLM" and provide specific, actionable "Teacher Learnings" on how to improve the *initial forecasting prompt* used by the Student. The Student uses a base prompt and adds forecasting instructions to it based on your learnings.
>
> **Key Information for Your Analysis for this Iteration** `{it + 1}`:
>
> 1. Current Forecasting Instructions Under Review: `{current_instructions_under_review"}`
> 2. Overall Mean Absolute Error (MAE) for this batch of samples: `{mae_to_report_to_teacher}`
>
> You will also be given a batch of individual samples, where each sample includes:
>
> 1. The full `Prompt` the Student LLM used (includes the instructions above).
> 2. The Student LLM's `Predictions` for the `OT` variable.
> 3. The `Ground-Truth OT` values.
>
> **Your Analysis Task:**
>
> 1. **Identify error patterns.** Compare `Predictions` with `Ground Truths`. Look for systematic errors (over/under-prediction, lagging, volatility mishandling, etc.).
> 2. **Correlate errors with prompts and instructions.** Check whether the current instructions are ambiguous, misleading, too complex, or otherwise harmful.
> 3. **Formulate "Teacher Learnings".** Give concrete, generalisable improvements (e.g. adjust look-back horizon, drop STL decomposition, add weekday feature).
> 4. **Determine "Done" status.**
>   - If the MAE `{mae_to_report_to_teacher}` is low and stable, or no samples were supplied, output `Done: True`.
>   - Otherwise output `Done: False`.
>
> **Output Format—exactly this template**
> Teacher Learnings: `<your concise, actionable suggestions here>`
> Done: `<True or False>`
> Confidence in output: `<High | Medium | Low>` – one-line rationale.

C Forecaster Agent

Prompt-Synthesis Instructions

Example: Forecasting-Instruction Refinement

> **You are an intelligent "Student LLM"** that refines forecasting prompts based on expert feedback. You will receive *Teacher Learnings* that suggest improvements to an initial time-series forecasting prompt. Your task is to turn these learnings into concise and effective *prompt-forecasting instructions*. These instructions will be appended to a base forecasting prompt to guide the

forecasting LLM.

The forecasting instructions should:

- Be a short set of guiding principles (max. 3 actionable items).
- Directly address the issues and suggestions in the Teacher Learnings.
- Be clearly phrased for another LLM to follow.
- **Do not include placeholders such as `{previous_data}` or `{prediction_data}`.
- **Do not change the output format or the forecasting task itself.
- If no actionable learnings exist, output a safe generic set—or state:

  No specific new instructions generated due to lack of actionable learnings.

Example (teacher said "focus on recent volatility"):

Teacher Learnings: The model often misses sudden spikes; the prompt should ask the forecaster to pay more attention to recent volatility and its effect on the next step.

Your Output (forecasting instructions): "Critically assess the volatility in the most recent data points. Your forecast for the next step should reflect whether this volatility is expected to continue, increase, or decrease. Explain this assumption in your reasoning."

Teacher Learnings you received:
`{current_learnings}`

Based on these learnings, generate only the refined prompt-forecasting instructions below (no extra commentary).

Refined Prompt Forecasting Instructions:
<model prediction here>

D Prompt template

Thinking Inducting Prompts

Example: Monte Hall Prompting

Objective
Provide a well-reasoned forecast for the `{target_variable}` value in the next row of the dataset, given the historical data.

Dataset Instructions

- **Dataset**: `data_name`, `data_description`
- **Variable to Predict**: `{target_variable}`.
- **Task**: Predict the `{target_variable}` values for the next `{prediction_length}` steps using the historical data.
- **Constraints**:
  - Adhere strictly to the specified output format.

If `instructions`:
`Forecasting Instructions: {instructions}`

If `raft_context`:
`{raft_context}`

Input Data

- Historical Data: `{previous_sequence_length_data}`

Output Format — exactly this

```
Predicted Values: [predicted_value_1, ...]
Reasoning: [Your detailed reasoning ]
Certainty Estimate: [Percentage certainty]
Certainty Reasoning: [reasoning]
```

E Prompt Library

These are the remaining prompts in the prompt library.

teacher-student-loop

```
ACT I — TEACHER Propose a first-pass forecast for the next {sequence_length} steps.
ACT II — STUDENT Evaluate teacher's forecast against the most recent known data and suggest corrections.
ACT III — TEACHER Incorporate feedback and provide the refined forecast.
```

self-verification-sets

```
Step 1 – Generate candidate forecast A for {sequence_length} steps. Step 2 – Generate independent candidate forecast B. Step 3 – For each horizon h, if the two differ beyond an acceptable tolerance, reconcile them (e.g., by averaging). Provide only the reconciled forecast.
```

meta-prompt-conf-bands

```
Forecast {sequence_length} steps and include 68 % and 95 % confidence bands. Briefly explain the uncertainty assumptions before the numbers.
```

imaginary-python-repl

```
You are ForecastPy, a mental Python REPL. Think then "run code in your head" that derives the forecast for the next {sequence_length} steps.
```

synesthetic-soundtrack

```
Interpret the past sequence as MIDI velocity (0-127) and compose the next {sequence_length} beats that extend the melody. Provide both the MIDI integers and the values rescaled to original units.
```

color-gradient-canvas

```
Map each value to an RGB triplet on a blue-to-red gradient. Produce a grid of HEX colours that encodes the next {sequence_length} points.
```

dungeon-master

```
You are a D&D Dungeon Master. The party's HP over the last turns is shown. Forecast HP for the next {sequence_length} turns, assuming no boss fights and only mild potion use.
```

micro-essay-poisson

```
Write a ≤60-word micro-abstract describing the generative mechanism, then list {sequence_length} λ parameters for a Poisson baseline.
```

reverse-sudoku

```
Think of the next {sequence_length} points as filling a 9 × 11 Sudoku-like grid whose row sums match the recent history. Provide the grid and a flattened list.
```

many-worlds-ensemble

```
Create forecasts for four parallel universes (A-D) shifted by −2σ, −1σ, +1σ, +2σ, each {sequence_length} steps long, then provide a consensus median forecast.
```

haiku-seeded

```
Compose a three-line haiku that metaphorically describes the upcoming pattern, then list the {sequence_length} numeric forecasts, one per line.
```

F Datasets

Experiments were performed on a diverse set of widely-used time-series-forecasting (TSF) benchmark datasets spanning multiple domains, sampling frequencies, and statistical characteristics (e.g., seasonality, trend, noise levels). All datasets are normalized with StandardScaling from sklearn package. The datasets are:

- **ETT** (`ETTh1`, `ETTh2`, `ETTm1`, `ETTm2`) – Electricity Transformer Temperature data recorded at hourly (`h`) or 15-minute (`m`) intervals; widely used for long-sequence forecasting with OT as target variable (ETTh: 17,420 total data points, ETTm: 69,680 total data points)

- **Electricity** – Hourly household electricity data of customers with electricity comsumption as target variable (26,304 total data points)

- **Traffic** – Hourly occupancy rates from California road-traffic sensors (2021-2025 March) with traffic volume as target variable (17,544 data points)

- **ILINet** – Weekly Influenza-Like-Illness counts from the CDC (2002-2025 April) with total ILI patients as target variable (1,441 total data points)[2]

- **Weather** - Hourly weather data from Chicago with temperature as target variable (35,052 total data points)[3]

F.1 Data Integrity

A significant consideration when utilizing Large Language Models (LLMs) for time series forecasting is the potential for the model's pre-training data to inadvertently include samples from the test set, which could lead to an overestimation of predictive performance. To rigorously uphold data integrity in this study, we employed ILINet and weather datasets as benchmarks, with a specific focus on temporal data separation. Our experimental design ensures that all data samples within the test set originate from dates strictly subsequent to the known training data cut-off date of the LLM employed for inference. This chronological separation mitigates the risk of test data contamination, providing a robust and fair evaluation of the LLM's ability to generalize and forecast genuinely unseen future values.

F.2 Evaluation Metrics

Forecasting performance was assessed with two standard error metrics:

[2] https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html
[3] https://www.kaggle.com/datasets/curiel/chicago-weather-database

$$\mathrm{MAE} = \frac{1}{H}\sum_{i=1}^{H}\left|\hat{X}_{t+i} - X_{t+i}\right|, \tag{1}$$

Where H is the prediction horizon, $\hat{X}_{t+i}$ is the predicted value, and X_{t+i} is the ground-truth value. Lower values indicate better performance for both metrics. These metrics were computed directly from the experimental results.

G Future directions

There are several avenues for future work. One direction is to incorporate quantitative validation in the loop: currently, the Refiner-agent's feedback quality is not directly measured. If we had a small hold-out set or could use the model's own likelihood of the data, we might select or weight feedback. This leans towards techniques in automatic prompt optimization where a reward is defined. Additionally, while FLAIRR-TS currently uses natural language for feedback from the Refiner-agent, one could imagine hybrid approaches where the Refiner-agent suggests pseudo-code or formulaic adjustments (if the LLM agents are equipped with a calculator tool). That could improve handling of scale and magnitude issues. On the retrieval side, exploring more advanced analog search (perhaps using learned embeddings or matching not just on raw values but pattern descriptors) might yield even more relevant cases to show the Refiner-agent, especially for complex multivariate data.

From an application perspective, deploying FLAIRR-TS in an interactive forecasting system would be very interesting. Because FLAIRR-TS's intermediate steps (the prompts, the retrieved analogs, the feedback) are human-readable, a human analyst could intervene in the loop – agreeing or disagreeing with the Refiner-agent's critique, or adding their own feedback. This could turn forecasting into a collaborative dialog between human, Forecaster-agent, and Refiner-agent. In settings like supply chain or epidemiology forecasting, such a system could help build trust as well, since each refinement step can be scrutinized.

H Potential Risks

- **Decision-critical misuse.** Deployment in safety- or finance-critical contexts without rigorous calibration could propagate spurious forecasts, leading to systemic harm.
- **Bias amplification.** Retrieval from historical data can embed and magnify demographic or regional skews, potentially disadvantaging underrepresented groups.
- **Privacy leakage.** Sending raw time-series to external LLM APIs risks exposing sensitive patterns; secure on-prem or encrypted inference is required for confidential data.
- **Environmental footprint.** Although we avoid training, repeated large-model inference still incurs non-trivial energy costs; batching and lighter models are possible mitigations.