ForecastCompass: Guiding Agentic Forecasting with Adaptive Factor Memory

Anonymous Authors[1]

Abstract

Agentic forecasting is important for decision-making in dynamic environments, yet remains difficult because agents must make calibrated predictions from incomplete, time-limited evidence. Memory can transfer lessons from resolved forecasts to future tasks, but existing agent-memory methods rarely capture reusable predictive factors or calibration knowledge. We propose **Fo**recast**Co**mpass (FoCo), an adaptive factor-based memory framework for agentic forecasting. FoCo organizes experience with a hierarchical task taxonomy and maintains two complementary memories: factor memory for reusable predictive dimensions and reasoning memory for probability updating, uncertainty handling, and calibration. Through retrospective memory revision, FoCo accumulates transferable forecasting knowledge over time. Experiments on Prophet Arena and FutureX with GPT-5-mini and Gemini-2.5-Flash show improved accuracy and calibration.

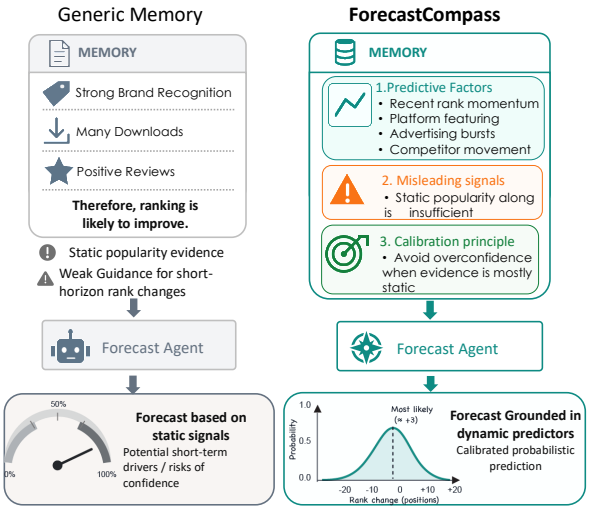


Figure 1. Generic memory v.s. FoCo memory on an example forecasting task.

1. Introduction

LLM-based agents have emerged as a promising framework for complex real-world tasks involving information seeking (Nakano et al., 2022), tool use (Patil et al., 2024; Schick et al., 2023), multi-step reasoning (Yao et al., 2023; Wei et al., 2022), and decision-making under uncertainty (Wang et al., 2023). Forecasting is a natural instance of this setting: many high-stakes decisions require anticipating uncertain future events, including market movements (Box et al., 2015; Fama, 1970), policy changes (Tetlock, 2017), and geopolitical developments (Tetlock & Gardner, 2016). As agents become capable of searching the web, using tools, and reasoning over dynamic information, they offer a promising foundation for agentic forecasting (Yao et al., 2022).

Agentic forecasting can be viewed as a flexible alternative to classical forecasting pipelines (Hyndman & Athanasopoulos, 2018; Nakano et al., 2022; Yao et al., 2022). Traditional methods typically specify predictive factors, model temporal dynamics, and update predictions with new observations (Ho & Xie, 1998; Hassan & Nath, 2005; Wu et al., 2021). However, they often rely on predefined features, structured inputs, and domain-specific assumptions, limiting their use in open-world settings where evidence is heterogeneous, and distributed across sources. In contrast, agentic forecasters can actively gather information, synthesize evidence, and adapt their reasoning to each task (Zeng et al., 2025; Yang et al., 2025; Chang et al., 2025).

Since forecasting depends on identifying predictive factors, we ask: *can agentic forecasters be guided to reason in a factor-centric manner?* Rather than storing only generic experiences or event-level outputs, an agent should learn factor-level knowledge from past events and reuse it in future forecasts. Memory offers a natural mechanism for storing such factor-granular information to support factor selection and probability calibration.

Existing memory mechanisms for LLM agents are mainly designed for question answering or problem solving, where memory helps recover answers, reuse reasoning traces, or avoid past mistakes (Lewis et al., 2020; Borgeaud et al., 2022; Chhikara et al., 2025). Forecasting requires different

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

memory: it must support calibrated predictions for unresolved events under partial and evolving evidence. Useful forecasting memory should abstract predictive factors, identify misleading signals, and guide uncertainty calibration. For example, in the app-ranking task in Figure 1, generic memory may emphasize static evidence such as strong brand recognition, many downloads, and positive reviews. A forecasting-specific memory should instead focus on dynamic factors such as recent rank momentum, platform featuring, advertising bursts, and competitor movement, while discouraging overconfidence from static popularity alone. This motivates a forecasting-specific memory design that captures reusable predictive factors, misleading signals, and calibration principles for future predictions.

We propose FoCo (**Fo**recast**Co**mpass), a forecasting-specific memory framework that stores past experience as hierarchical, subcategory-level memory for future probabilistic prediction. FoCo maintains two memory types: *predictive-factor memory*, which captures reusable signals, misleading evidence patterns, and factor-level lessons, and *calibration-oriented reasoning memory*, which captures how confidence should shift under different evidence conditions. During inference, retrieved memory guides evidence search, factor selection, and probability estimation. To build memory, FoCo contrasts forecasting trajectories with retrospective analyses, using discrepancies to revise reusable factors and calibration principles while avoiding event-level hindsight. Our contributions are:

- We introduce *factor-centric memory* for agentic forecasting, organizing memory as predictive factors and calibration patterns rather than event-level trajectories.
- We propose *verbalized factor-memory revision*, an iterative diagnose–aggregate–revise procedure that converts retrospective signals into transferable forecasting abstractions while excluding event-specific hindsight.
- We evaluate FoCo on Prophet Arena and FutureX, showing improved accuracy and calibration, with memory transferring across time periods and model backbones.

2. Method

We introduce FoCo, a framework that builds forecasting memory from retrospective revisions to predictive factors. FoCo organizes memory with a hierarchical taxonomy, retrieves subcategory-level memory during inference, and updates memory chronologically after questions are resolved, as shown in Algorithm 2. Detailed procedures are provided in Appendix E.

2.1. Hierarchical Forecasting Memory Representation

At update step w, the taxonomy T_w contains categories and fine-grained subcategories. Each forecasting question is assigned to one subcategory s, which indexes a verbal memory state

$$M_{w,s} = (F_{w,s}, R_{w,s}). \tag{1}$$

Here, $F_{w,s}$ stores reusable predictive-factor entries, and $R_{w,s}$ stores calibration-oriented reasoning principles. Each factor entry records the factor name, evidence checks, common failure modes, and typical effect on forecast probability. Thus, memory abstracts resolved forecasting records into reusable subcategory-level principles rather than event-specific conclusions.

2.2. Memory-Augmented Forecasting Inference

For each unresolved question q_i, FoCo assigns it to a category–subcategory pair (c_i, s_i) under T_w and retrieves M_{w,s_i}. Given the question, evidence pool E_i, and retrieved memory, the forecasting agent produces a reasoning trajectory and probability distribution:

$$(z_i^{\text{FoCo}}, p_i^{\text{FoCo}}) = A_\theta(q_i, E_i, M_{w,s_i}), \qquad p_i^{\text{FoCo}} \in \Delta^{K_i - 1}. \tag{2}$$

The factor memory guides evidence search and factor selection, while the reasoning memory guides calibration under uncertainty, conflicting evidence, and missing information.

2.3. Taxonomy and Memory Update Mechanism

After questions from step w are resolved, FoCo updates the taxonomy and memories using only past resolved questions. Previously unmatched questions are grouped into reusable category–subcategory patterns, consolidated with existing taxonomy entries when appropriate, and added as

$$T_w = T_{w-1} \cup \Delta T_w. \tag{3}$$

For memory updates, each resolved question is routed to a subcategory under T_w. Only activated subcategories are updated. For each resolved question q_i, the agent generates a retrospective trajectory using post-resolution evidence:

$$z_i^{\text{retro}} = A_\theta^{\text{retro}}(q_i, E_i^{\text{retro}}). \tag{4}$$

This trajectory is used only as a learning signal. FoCo contrasts the original and retrospective trajectories to obtain

$$\Delta_i = \text{Contrast}(z_i, z_i^{\text{retro}}) = (\Delta_i^F, \Delta_i^R), \tag{5}$$

where Δ_i^F captures factor-level errors and Δ_i^R captures reasoning-level errors.

The LLM aggregates these discrepancies into memory-revision suggestions:

$$\mathcal{U}_i = \left(\bigcup_{f \in \widetilde{F}_{i,s_i}} \{\text{Revise}_F(f, \bar{\Delta}_i^F)\},\ \text{Revise}_R(R_{w-1,s_i}, \bar{\Delta}_i^R) \right). \tag{6}$$

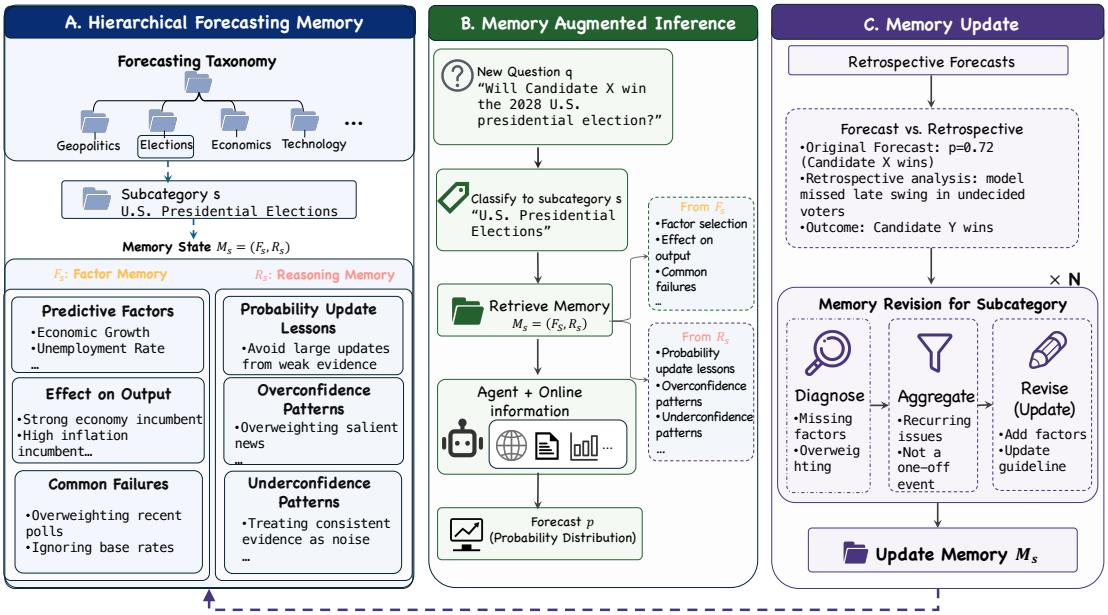


Figure 2. Framework overview. *A*: the hierarchical forecasting memory organizes subcategory-level factor memory and reasoning memory. *B*: a new question is classified into a subcategory, relevant memory is retrieved, and the agent produces a calibrated forecast. *C*: the memory is updated by contrasting original trajectories with retrospective trajectories and iteratively revised.

Only factor entries directly affected by the discrepancy are revised. After N diagnose–aggregate–revise iterations, the updated memory is stored as $M_{w,s}$. This iterative process helps retain recurring forecasting patterns while filtering event-specific hindsight and transient correlations.

3. Experiments

3.1. Experimental settings

We evaluate FoCo with LM-based forecasting agents on dynamic benchmarks, where agents reason over predictive factors, gather evidence, and output calibrated probabilities. **Models:** We use GPT-5-mini (Singh et al., 2025) and Gemini-2.5-Flash (Comanici et al., 2025) as forecasting backbones. Unless stated otherwise, memory construction and revision use the same backbone. We also test cross-model transfer by applying Gemini-constructed memory to GPT-5-mini. Additional setup details appear in Appendix D.1. **Datasets:** We evaluate on Prophet Arena (Yang et al., 2025) and FutureX (Zeng et al., 2025), which contain time-evolving forecasting questions across domains including sports, technology, and finance. For each dataset, we use the latest five weeks to construct, update, and evaluate memory. Evaluation is chronological: memory is built only from previously resolved questions and used to forecast future unresolved ones, preventing post-resolution leakage. Data statistics are in Appendix C.1. **Evaluation metrics:** We report Brier score (Glenn et al., 1950) and expected calibration error (ECE) (Yang et al., 2025). Brier score measures probabilistic accuracy, while ECE measures calibration across confidence bins. Both are lower-is-better; formal definitions are in Appendix C.4. **Baselines:** We compare against baselines spanning forecasting and memory-design alternatives. Reference baselines include Base, which uses no external memory, and Base (Retro), a non-deployable retrospective diagnostic that analyzes questions after resolution. We include FoCo (Static) to isolate the effect of weekly memory updates. Memory-mechanism baselines include Graphiti (Rasmussen et al., 2025) for structured entity–relation memory; Mem0 (Chhikara et al., 2025) for retrieval-based long-term memory; and Reflexion (Shinn et al., 2024) and A-Mem (Xu et al., 2025) for reflective or experience-driven memory. Details are provided in Appx. C.5.

3.2. Main Results

Table 1 summarizes the average Brier and ECE scores across all evaluation weeks on Prophet Arena and FutureX. Full week-by-week Brier and ECE results are provided in Appendix D.4, respectively. The evolution of the taxonomy, with increasing numbers of categories and subcategories as more events are incorporated, is provided in Appendix D.6. We additionally report bootstrap confidence intervals for Brier score in Appendix D.7 to assess the robustness of different methods.

Factor-centric memory consistently improves forecasting accuracy and calibration. Across both datasets and backbone models, FoCo consistently achieves the best average Brier and ECE scores among all deployable methods. Compared with Base, the gains show that factor-centric memory improves both probabilistic accuracy and calibra-

Method	GPT-5-mini				Gemini-2.5-Flash			
	Prophet Arena		FutureX		Prophet Arena		FutureX	
	Brier↓	ECE↓	Brier↓	ECE↓	Brier↓	ECE↓	Brier↓	ECE↓
BASE (Retro)	0.109	0.079	0.197	0.209	0.187	0.098	0.241	0.279
BASE	0.150	0.114	0.241	0.263	0.202	0.106	0.266	0.299
MEM0	0.149	0.101	0.197	0.217	0.208	0.125	0.272	0.296
REFLEXION	0.150	0.086	0.203	0.222	0.196	0.114	0.252	0.266
A-MEM	0.109	0.092	0.194	0.208	0.204	0.115	0.269	0.287
GRAPHITI	0.134	0.097	0.218	0.244	0.215	0.140	0.275	0.301
FOCO (Static)	0.083	0.089	0.203	0.219	0.134	0.112	0.243	0.237
FOCO	**0.075**	**0.077**	**0.187**	**0.195**	**0.118**	**0.090**	**0.216**	**0.198**

Table 1. Average Brier and ECE results on Prophet Arena and FutureX across backbone models.

tion, while the improvements over FOCO (Static) indicate the importance of iterative memory revision. We include BASE (Retro) only as a retrospective diagnostic reference: although it uses post-resolution information, it is not an upper bound because retrospective analysis may induce hindsight bias and overconfident probabilities. Its role is to help diagnose whether errors stem from missing information, reasoning failures, or calibration issues.

Forecasting-specific memory outperforms general agent-memory baselines. Existing memory methods such as MEM0, REFLEXION, A-MEM, and GRAPHITI can retain and reuse past experience, but their gains are less consistent across datasets and models. In contrast, FOCO obtains the lowest average Brier and ECE scores in all four model-dataset settings in Table 1. This suggests that simply storing past agent experience is not sufficient for forecasting; organizing memory around reusable predictive factors and calibration-oriented reasoning is more effective.

Iterative memory evolution provides additional gains beyond static memory. FOCO (Static) improves over BASE in several settings, indicating that initialized factor-centric memory is already useful. The full FOCO further and consistently improves over this static variant, showing that weekly memory revision refines predictive factors and reasoning patterns as new outcomes become available.

Additional Experiments. Beyond the main comparison, we conduct several additional experiments to analyze the source and robustness of the gains. First, the ablation study in Table 3 shows that predictive-factor memory and reasoning/calibration memory are complementary: factor memory helps select reliable predictive signals, while reasoning memory helps translate them into calibrated probability estimates. Second, the transferability results in Figure 6 show that learned memory is not tied to a single backbone model, as memory constructed with one model still improves both Brier score and ECE when used by another model. Third, the taxonomy-evolution analysis in Figure 7 shows that forecasting categories and subcategories grow over time, supporting the need for an adaptive taxonomy and subcategory-level memory. Finally, the bootstrap analysis in Table 6 and the strengthened retrospective Reflexion baseline in Table 7 further indicate that the improvements are robust and that FOCO benefits from abstracting retrospective diagnostic signals into structured, reusable forecasting knowledge rather than merely storing more experience.

4. Related Work

Our work builds on two lines of research, with a full discussion provided in Appendix A. First, forecasting has been studied through classical time-series, probabilistic, expert, and crowd-based methods (Hyndman & Athanasopoulos, 2018; Tetlock & Gardner, 2016; Wolfers & Zitzewitz, 2004), while recent work studies language-model-based forecasting agents that gather open-world evidence and estimate uncertain future outcomes (Chang et al., 2025; Cheng et al., 2026; Yang et al., 2025; Zeng et al., 2025). Second, memory-augmented agents store reusable experience, reflections, skills, or structured knowledge to improve future task performance (Park et al., 2023; Packer et al., 2023; Shinn et al., 2024; Madaan et al., 2023; Wang et al., 2023; Rasmussen et al., 2025). Unlike these general-purpose memory systems, FOCO develops forecasting-specific memory that captures reusable predictive factors and calibration-oriented reasoning, enabling agents to transfer lessons from resolved forecasts to future unresolved questions.

5. Conclusion

This work studies memory design for agentic forecasting under incomplete, evolving evidence. We propose FOCO, which learns reusable predictive factors and calibration principles from resolved tasks. Across dynamic forecasting benchmarks, it improves probabilistic accuracy and calibration, with memory transferring across time periods and model backbones.

References

Borgeaud, S., Mensch, A., Hoffmann, J., Cai, T., Rutherford, E., Millican, K., Van Den Driessche, G. B., Lespiau, J.-B., Damoc, B., Clark, A., et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pp. 2206–2240. PMLR, 2022.

Box, G. E., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

Cai, T., Wang, X., Ma, T., Chen, X., and Zhou, D. Large language models as tool makers. *arXiv preprint arXiv:2305.17126*, 2023.

Chang, C., Shi, Y., Cao, D., Yang, W., Hwang, J., Wang, H., Pang, J., Wang, W., Liu, Y., Peng, W.-C., et al. A survey of reasoning and agentic systems in time series with large language models. *arXiv preprint arXiv:2509.11575*, 2025.

Chang, Y., Wu, Y., Wu, Q., and Lin, L. Memcollab: Cross-agent memory collaboration via contrastive trajectory distillation. *arXiv preprint arXiv:2603.23234*, 2026.

Cheng, M., Tao, X., Liu, Q., Guo, Z., and Chen, E. Position: Beyond model-centric prediction–agentic time series forecasting. *arXiv preprint arXiv:2602.01776*, 2026.

Chhikara, P., Khant, D., Aryan, S., Singh, T., and Yadav, D. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Comanici, G., Bieber, E., Schaekermann, M., Pasupat, I., Sachdeva, N., Dhillon, I., Blistein, M., Ram, O., Zhang, D., Rosen, E., et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Fama, E. F. Efficient capital markets: A review of theory and empirical work. *The journal of Finance*, 25(2):383–417, 1970.

Glenn, W. B. et al. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

Gutiérrez, B. J., Shu, Y., Gu, Y., Yasunaga, M., and Su, Y. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in neural information processing systems*, 37:59532–59569, 2024.

Hassan, M. R. and Nath, B. Stock market forecasting using hidden markov model: a new approach. In *5th international conference on intelligent systems design and applications (ISDA'05)*, pp. 192–196. IEEE, 2005.

Ho, S. L. and Xie, M. The use of arima models for reliability forecasting and analysis. *Computers & industrial engineering*, 35(1-2):213–216, 1998.

Hyndman, R. J. and Athanasopoulos, G. *Forecasting: principles and practice*. OTexts, 2018.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., et al. Webgpt: Browser-assisted question-answering with human feedback, 2022. *URL https://arxiv.org/abs/2112.09332*, 35, 2022.

Packer, C., Fang, V., Patil, S., Lin, K., Wooders, S., and Gonzalez, J. Memgpt: towards llms as operating systems. 2023.

Park, J. S., O'Brien, J., Cai, C. J., Morris, M. R., Liang, P., and Bernstein, M. S. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Patil, S. G., Zhang, T., Wang, X., and Gonzalez, J. E. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems*, 37: 126544–126565, 2024.

Rasmussen, P., Paliychuk, P., Beauvais, T., Ryan, J., and Chalef, D. Zep: a temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*, 2025.

Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., and Scialom, T. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

Shinn, N., Cassano, F., Berman, E., Gopinath, A., Narasimhan, K., and Yao, S. Reflexion: Language

agents with verbal reinforcement learning, 2023. *URL https://arxiv. org/abs/2303.11366*, 8, 2024.

Singh, A., Fry, A., Perelman, A., Tart, A., Ganesh, A., El-Kishky, A., McLaughlin, A., Low, A., Ostrow, A., Ananthram, A., et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Tetlock, P. E. Expert political judgment: How good is it? how can we know?-new edition. 2017.

Tetlock, P. E. and Gardner, D. *Superforecasting: The art and science of prediction*. Random House, 2016.

Wang, G., Xie, Y., Jiang, Y., Mandlekar, A., Xiao, C., Zhu, Y., Fan, L., and Anandkumar, A. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Wolfers, J. and Zitzewitz, E. Prediction markets. *Journal of economic perspectives*, 18(2):107–126, 2004.

Wu, Y., Ni, J., Cheng, W., Zong, B., Song, D., Chen, Z., Liu, Y., Zhang, X., Chen, H., and Davidson, S. B. Dynamic gaussian mixture based deep generative model for robust forecasting on sparse multivariate time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 651–659, 2021.

Xu, W., Liang, Z., Mei, K., Gao, H., Tan, J., and Zhang, Y. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025.

Yang, Q., Mahns, S., Li, S., Gu, A., Wu, J., and Xu, H. Llm-as-a-prophet: Understanding predictive intelligence with prophet arena. *arXiv preprint arXiv:2510.17638*, 2025.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., and Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Zeng, Z., Liu, J., Chen, S., He, T., Liao, Y., Tian, Y., Wang, J., Wang, Z., Yang, Y., Yin, L., et al. Futurex: An advanced live benchmark for llm agents in future prediction. *arXiv preprint arXiv:2508.11987*, 2025.

Zhang, Z., Dai, Q., Bo, X., Ma, C., Li, R., Chen, X., Zhu, J., Dong, Z., and Wen, J.-R. A survey on the memory mechanism of large language model-based agents. *ACM Transactions on Information Systems*, 43(6):1–47, 2025.

Zhong, W., Guo, L., Gao, Q., Ye, H., and Wang, Y. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 19724–19731, 2024.

A. Detailed Related Works

Forecasting. Forecasting aims to make probabilistic predictions about future events under incomplete and evolving evidence (Hyndman & Athanasopoulos, 2018; Tetlock & Gardner, 2016). Classical approaches typically rely on structured historical data, predefined variables, or domain-specific assumptions, such as time-series models (Ho & Xie, 1998), probabilistic models (Hassan & Nath, 2005; Wu et al., 2021), and expert or crowd forecasting systems (Tetlock, 2017; Wolfers & Zitzewitz, 2004). In contrast, language-model-based forecasting agents operate in open-world information environments, where they must gather evidence, synthesize heterogeneous information, identify predictive factors, and reason about uncertain future outcomes (Chang et al., 2025; Cheng et al., 2026). Existing studies show that LLMs have promising forecasting capabilities, but still face challenges in temporal reasoning, evidence selection, quantitative estimation, and probability calibration (Yang et al., 2025; Zeng et al., 2025). Our work improves agentic forecasting through memory. Rather than treating each forecasting question independently, FoCo uses resolved forecasting tasks to construct reusable predictive-factor and calibration memories for future unresolved questions under chronological constraints.

Memory-Augmented Agents. Memory has become a key component of language-model agents, enabling them to preserve and reuse information beyond the current context (Zhang et al., 2025; Xu et al., 2025; Chang et al., 2026). Existing memory-augmented agents differ primarily in the content they store. Episodic memory stores past observations, interactions, or trajectories as reusable experience (Park et al., 2023; Packer et al., 2023); reflection-based memory stores verbal feedback or lessons distilled from prior successes and failures (Shinn et al., 2024; Madaan et al., 2023); skill-based memory stores reusable procedures, strategies, or action programs (Wang et al., 2023; Cai et al., 2023); and graph-based memory represents entities, relations, and temporal changes as structured knowledge (Rasmussen et al., 2025; Gutiérrez et al., 2024). At inference time, these memories are typically retrieved or accessed to provide relevant context for new tasks (Zhong et al., 2024; Chhikara et al., 2025). However, while existing memory systems preserve past interactions, reflections, or structured knowledge for future reuse, they are not explicitly designed for forecasting, where memory should capture predictive signals, misleading evidence patterns, and calibration principles under incomplete evidence (Tetlock & Gardner, 2016; Guo et al., 2017). FoCo addresses this gap by maintaining forecasting-specific memory with two complementary contents: predictive-factor memory and calibration-oriented reasoning memory. It further revises these memories by contrasting forecasting trajectories with retrospective trajectories from resolved tasks.

B. Preliminary

Agentic forecasting. We consider a chronological forecasting setting where a language-model agent A_θ makes probabilistic predictions about future events by acquiring evidence and reasoning over it. For a question q_t with K_t possible outcomes, the agent observes a time-valid evidence pool E_t and produces

$$(z_t, p_t) = A_\theta(q_t, E_t), \qquad p_t \in \Delta^{K_t - 1}, \tag{7}$$

where z_t is the generation trajectory and p_t is the forecast distribution over the K_t outcomes.

Information regimes. We distinguish two regimes. In the forecasting regime, the agent can only use pre-resolution evidence E_t and produces a deployable forecast. After the outcome is resolved, the retrospective regime provides post-resolution evidence E_t^{retro}, from which the agent generates a retrospective trajectory z_t^{retro}. This trajectory is not used for deployment; it serves as a learning signal that exposes the strengths and failures of the original forecast.

C. Additional Experimental Details

C.1. Data Statistics

C.2. Case Study

Figure 3 illustrates how FoCo improves individual forecasts. In Event A, factor memory helps the agent focus on ranking-specific signals, such as availability, release timing, chart decay, and competition, rather than relying mainly on generic popularity. This changes the forecast from a broad popularity judgment to a position-specific probability estimate. In Event B, reasoning memory helps the agent distinguish testing or planning signals from full-season institutional adoption, reducing overconfidence in the wrong season. Across both cases, memory improves the forecast by guiding factor selection and probability calibration, rather than by retrieving event-level answers. We further provide representative examples of the learned memory content in Appendix F.1 and report an LLM-as-a-judge evaluation of memory quality in Section C.3.

Table 2. Data statistics for Prophet Arena and FutureX.

Week	Prophet Arena	FutureX
Week 0	100	58
Week 1	178	34
Week 2	136	76
Week 3	88	53
Week 4	38	21
Total	640	242

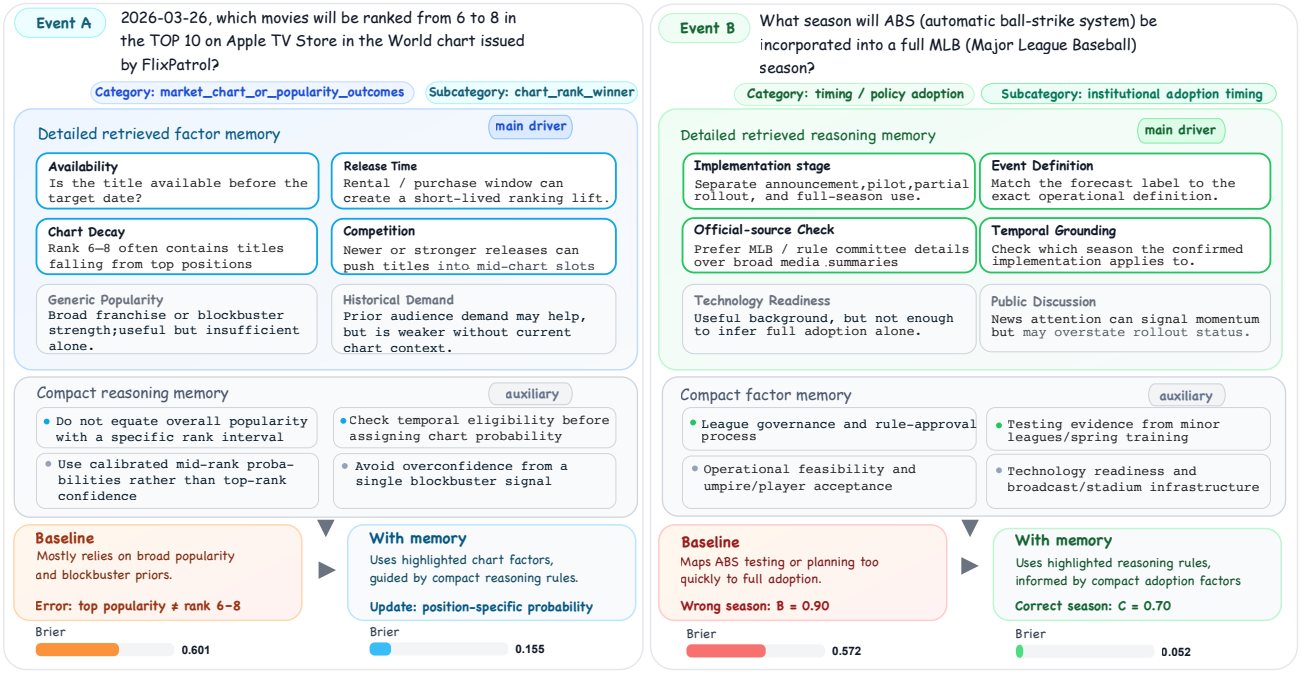


Figure 3. Case Study of how FoCo improves forecasts through factor selection and probability calibration.

C.3. Memory Quality Analysis

We use GPT-5.2 to assess whether memory generated by Gemini-2.5-Flash on Prophet-Arena captures reusable forecasting knowledge rather than event-specific details. Memories are scored on five dimensions: generalizability, novelty, completeness, accuracy, and actionability, using the prompt in Appendix G.3. As shown in Figure 4, FoCo achieves the highest average score and performs strongly across all dimensions. This suggests that its memory abstracts recurring predictive factors and calibration-oriented reasoning patterns, rather than storing event-level forecast records. The strong generalizability and actionability scores further indicate that FoCo avoids relying on specific entities, dates, resolved outcomes, or post-resolution facts, supporting its ability to mitigate information leakage through high-level forecasting principles.

C.4. Evaluation Metrics

We evaluate probabilistic forecasts using both accuracy and calibration metrics. For prediction accuracy, we use the multi-class Brier score:

$$\text{Brier}(\mathbf{p}_t, \mathbf{y}_t) = \sum_{k=1}^{K_t} (p_{t,k} - y_{t,k})^2,$$

where $\mathbf{p}_t \in \Delta^{K_t - 1}$ is the predicted probability distribution and $\mathbf{y}_t \in \{0, 1\}^{K_t}$ is the one-hot encoded realized outcome. Lower Brier score indicates that the predicted probability distribution is closer to the realized outcome.

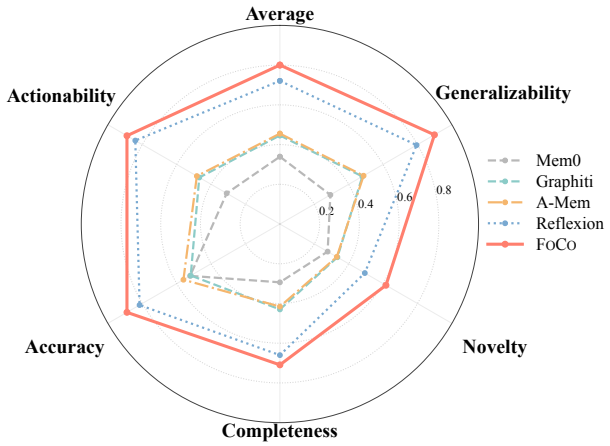


Figure 4. Memory quality comparison on Prophet Arena using LLM-as-a-Judge.

For calibration, we use expected calibration error (ECE). Given N forecasts, let

$$\hat{k}_t = \arg\max_k p_{t,k}, \qquad k_t^\star = \arg\max_k y_{t,k}, \qquad c_t = \max_k p_{t,k},$$

where $\hat{k}_t$ is the predicted class, $k_t^\star$ is the realized class, and c_t is the prediction confidence. We partition predictions into B confidence bins $\{I_b\}_{b=1}^B$. For each bin I_b, the empirical accuracy and confidence are defined as

$$\mathrm{acc}(I_b) = \frac{1}{|I_b|} \sum_{t \in I_b} \mathbf{1}[\hat{k}_t = k_t^\star],$$

$$\mathrm{conf}(I_b) = \frac{1}{|I_b|} \sum_{t \in I_b} c_t.$$

The expected calibration error is then

$$\mathrm{ECE} = \sum_{b=1}^{B} \frac{|I_b|}{N} \left| \mathrm{acc}(I_b) - \mathrm{conf}(I_b) \right|.$$

Lower ECE indicates better calibration, meaning that predicted confidence better matches empirical correctness.

C.5. Baseline Adaptation Details

We provide additional details on how each memory baseline is adapted to the forecasting setting. All baselines use the same backbone forecasting agent and follow the same weekly chronological protocol: memory is updated only from previously resolved questions and is used only for future questions. No baseline receives retrospective information about the current evaluation question. The goal of this protocol is to ensure that all methods operate under the same temporal constraint. Also, for all baselines, we use the same factor-centric forecasting prompt in G.1 as FoCo that the agent is first instructed to decompose each question into predictive factors and then produce the final probabilistic forecast.

Reflexion. Reflexion stores verbal feedback from past trials and retrieves it for future tasks. For a fair comparison, we use the same factor-centric forecasting prompt as FoCo for Reflexion, so the agent first decomposes each question into predictive factors before producing the forecast. For each resolved forecast, we generate a reflection from the original factor-centric forecasting trajectory, the predicted probability distribution, and the ground-truth outcome. The reflection summarizes factor-level successes and failures, such as whether the agent over-relied on weak evidence, missed an important predictive factor, or assigned an overconfident probability. At future forecasting time, relevant reflections are retrieved and prepended to the forecasting prompt. Thus, Reflexion also receives factor-centric outcome feedback from resolved questions, but unlike FoCo, it does not organize the resulting knowledge into structured predictive-factor and calibration memories.

Mem0. Mem0 is adapted as a general long-term memory system over resolved forecasting records. Each resolved question is treated as an interaction containing the question, forecast reasoning, predicted probability, and realized outcome. Mem0 extracts salient memory snippets from these interactions and retrieves relevant memories for future questions. Unlike FoCo, Mem0 does not maintain a category-subcategory taxonomy and does not explicitly separate predictive factors from reasoning or calibration patterns. Its memory is organized by general relevance retrieval over past interactions.

A-Mem. A-Mem is adapted as an experience-level memory baseline. For each resolved forecasting record, we store the question, the agent's reasoning, the predicted probabilities, and the realized outcome as an agent experience. During inference, A-Mem retrieves similar past experiences based on semantic similarity and injects them into the prompt. A-Mem is updated weekly using newly resolved records, following the same chronological protocol as FoCo. However, it does not use the subcategory taxonomy as a core memory structure and does not explicitly aggregate multiple experiences into subcategory-level common factors or calibration rules.

Graphiti. Graphiti is adapted as a graph-structured memory baseline. For each resolved forecasting record, an LLM extracts factual triples from the question, reasoning, evidence, and outcome. These triples are inserted into a dynamic graph as entity and relation nodes. During forecasting, relevant graph context is retrieved based on the current question and provided to the agent. Graphiti can partially support cross-question generalization through shared entities or communities, but it does not explicitly construct subcategory-level predictive-factor memory or calibration-oriented reasoning memory.

D. Additional Experiments

D.1. Experimental setup

For all experiments, we implement the forecasting agent using the OpenAI SDK. Unless otherwise specified, we use GPT-5-mini with medium reasoning effort and set the nucleus sampling parameter top_p to 0.7. To simulate a realistic forecasting setting and avoid information leakage, we restrict the search window to information available no later than one week before the event resolution time. To account for differences in the number of available events, we set the number of memory-revision epochs to three for Prophet Arena and two for FutureX. The detailed discussion for selecting the epochs is in Appendix D.10.

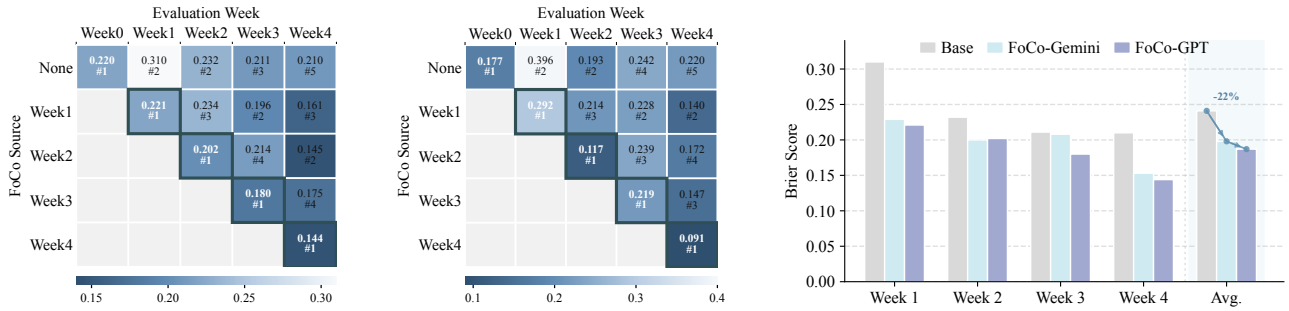
D.2. Transferability

We further evaluate two key properties of FOCO: the generalizability of factor-centric memory and the importance of iterative memory updating. Across-time transferability tests whether memory constructed from earlier weeks remains useful for later forecasting periods, while across-model transferability tests whether the same memory framework provides consistent benefits across different backbone agents.

Across-time Transferability Figure 5b and Figure 5a report across-time transfer results on FutureX. Each row corresponds to the FOCO memory source, and each column corresponds to the evaluation week. The "None" row denotes the non-memory BASE agent, while row Week k denotes using memory constructed up to Week k to forecast later weeks. The left panel reports Brier score, and the right panel reports ECE.

Overall, FOCO provides transferable gains across future weeks. Memory improves later-week Brier and ECE scores over the non-memory baseline, with stronger gains generally appearing after it has incorporated more resolved forecasts. Although transfer is not uniformly monotonic across all source–target pairs due to shifts in topics and evidence conditions, the overall pattern suggests that FOCO transfers both predictive factors and calibration guidance across time, supporting the value of iterative memory evolution.

Across-model Transferability For across-model transfer, we construct FOCO memory with GPT-5-mini and evaluate the same memory with Gemini-2.5-Flash. Figure 5c shows that FOCO provides consistent gains across different backbone models on Brier scores. Full results including the ECE scores are in the Appendix D.5. The results indicate that factor-centric memory is not tied to a single backbone agent. Although different models may use the memory differently, the same memory design consistently improves both probabilistic accuracy and calibration. This supports our claim that FOCO captures reusable forecasting knowledge, including predictive factors and calibration-oriented reasoning patterns, rather than overfitting to a specific model or time period.



(a) Brier score *(b)* ECE score *(c)* Across-model performance

Figure 5. Transferability analysis of FoCo on FutureX. We evaluate both across-time transferability and across-model transferability.

D.3. Ablation Study

Methods	Week 1		Week 2		Week 3		Week 4		Avg.	
	Brier↓	ECE↓	Brier↓	ECE↓	Brier↓	ECE↓	Brier↓	ECE↓	Brier↓	ECE↓
BASE	0.310	0.396	0.232	0.193	0.211	0.242	0.210	0.220	0.241	0.263
FoCo w/o factor	0.233	0.325	0.217	0.196	0.200	0.230	0.178	**0.083**	0.207	0.209
FoCo w/o reasoning	0.246	0.331	0.208	0.179	0.214	0.238	0.150	0.133	0.205	0.220
FoCo	**0.221**	**0.292**	**0.202**	**0.177**	**0.180**	**0.219**	**0.144**	0.091	**0.187**	**0.195**

Table 3. Ablation study on FutureX with GPT-5-mini.

We conduct an ablation study on FutureX with GPT-5-mini to assess the two memory components in FoCo. Table 3 compares the full model with two variants: FoCo w/o factor, which removes predictive-factor memory, and FoCo w/o reasoning, which removes reasoning and calibration memory. Results show that the two components are complementary. Factor memory improves factor selection by emphasizing predictive signals and reducing reliance on weak or misleading ones, while reasoning memory helps translate factors into calibrated probability estimates. The full model performs best by combining accurate factor selection with calibration-oriented reasoning.

D.4. Additional results on ECE scores and Brier scores

We additionally report ECE results to evaluate the calibration quality of different methods. As shown in Table 5, FoCo consistently achieves the lowest average ECE across both datasets and backbone models, indicating that its forecasting-specific memory not only improves prediction accuracy but also leads to better-calibrated probability estimates. These results further support the effectiveness of structured predictive-factor and calibration-oriented memory for probabilistic forecasting.

D.5. Additional results on transferability

We further evaluate whether the learned forecasting memory can transfer across backbone models. Figure 6 reports results on FutureX using GPT-5-mini as the forecasting backbone, comparing the no-memory baseline, memory constructed with Gemini-2.5-Flash, and memory constructed with GPT-5-mini. The transferred memory consistently improves both Brier score and ECE over the no-memory baseline, showing that the learned predictive-factor and calibration-oriented memory captures reusable forecasting knowledge beyond a specific backbone model.

D.6. Taxonomy Evolution

Beyond forecasting performance, we analyze how the forecasting taxonomy evolves during memory construction. Figure 7 shows the number of categories and subcategories over construction weeks on FutureX and Prophet Arena. For both datasets, the number of categories and subcategories increases as more weekly data is incorporated. This indicates that the forecasting data covers an expanding set of topics over time, and that a fixed taxonomy may be insufficient to organize all emerging questions. The result supports our taxonomy update module, which allows new categories and subcategories to be added

Backbone	Method	Prophet Arena					FutureX				
		Week 1	Week 2	Week 3	Week 4	Avg.	Week 1	Week 2	Week 3	Week 4	Avg.
GPT-5-mini	BASE (Retro)	0.096	0.113	0.141	0.084	0.109	0.250	0.217	0.212	0.109	0.197
	BASE	0.130	0.135	0.175	0.161	0.150	0.310	0.232	0.211	0.210	0.241
	MEM0	0.146	0.183	0.138	0.128	0.149	0.232	0.233	0.183	0.138	0.197
	REFLEXION	0.130	0.147	0.155	0.167	0.150	0.246	0.183	0.213	0.170	0.203
	A-MEM	0.117	0.075	0.131	0.114	0.109	0.238	0.195	0.218	**0.124**	0.194
	GRAPHITI	0.128	0.135	0.137	0.134	0.134	0.250	0.233	0.218	0.172	0.218
	FOCO (Static)	0.064	0.066	0.119	0.081	0.083	0.221	0.234	0.196	0.161	0.203
	FOCO	**0.064**	**0.069**	**0.092**	**0.073**	**0.075**	**0.221**	**0.202**	**0.180**	0.144	**0.187**
Gemini-2.5-Flash	BASE (Retro)	0.147	0.188	0.233	0.181	0.187	0.303	0.245	0.230	0.185	0.241
	BASE	0.169	0.226	0.221	0.190	0.202	0.329	0.272	0.237	0.227	0.266
	MEM0	0.180	0.225	0.221	0.205	0.208	0.319	0.281	0.246	0.242	0.272
	REFLEXION	0.179	0.228	0.200	0.178	0.196	0.313	0.259	0.221	0.214	0.252
	A-MEM	0.179	0.220	0.210	0.206	0.204	0.354	0.264	0.266	0.190	0.269
	GRAPHITI	0.175	0.208	0.248	0.230	0.215	0.354	0.261	0.243	0.242	0.275
	FOCO (Static)	0.123	0.099	0.182	0.131	0.134	0.279	0.245	0.230	0.217	0.243
	FOCO	**0.123**	**0.115**	**0.138**	**0.096**	**0.118**	**0.279**	**0.223**	**0.217**	**0.144**	**0.216**

Table 4. Brier Score(↓) results on Prophet Arena and FutureX across different models.

Backbone	Method	Prophet Arena					FutureX				
		Week 1	Week 2	Week 3	Week 4	Avg.	Week 1	Week 2	Week 3	Week 4	Avg.
GPT-5-mini	BASE (Retro)	0.035	0.131	0.075	0.073	0.079	0.309	0.183	0.234	0.109	0.209
	BASE	0.048	0.191	0.100	0.118	0.114	0.396	0.193	0.242	0.220	0.263
	Mem0	0.040	0.149	0.094	0.120	0.101	0.320	0.236	0.216	0.097	0.217
	Reflexion	0.036	0.078	0.102	0.127	0.086	0.336	0.185	0.234	0.132	0.222
	A-Mem	0.052	0.103	0.103	0.111	0.092	0.291	0.180	0.262	0.100	0.208
	Graphiti	0.044	0.129	0.107	0.107	0.097	0.353	0.236	0.261	0.125	0.244
	FOCO (Static)	0.042	0.114	0.098	0.103	0.089	0.292	0.214	0.228	0.140	0.219
	FOCO	0.042	**0.064**	0.101	0.099	**0.077**	0.292	**0.177**	**0.219**	**0.091**	**0.195**
Gemini-2.5-Flash	BASE (Retro)	0.044	0.054	0.172	0.123	0.098	0.393	0.240	0.269	0.215	0.279
	BASE	0.051	0.095	0.152	0.125	0.106	0.409	0.271	0.252	0.265	0.299
	Mem0	0.060	0.141	0.155	0.143	0.125	0.390	0.292	0.289	0.213	0.296
	Reflexion	0.062	0.161	0.116	0.117	0.114	0.363	0.246	0.231	0.223	0.266
	A-Mem	0.064	0.132	0.122	0.140	0.115	0.452	0.245	0.282	0.167	0.287
	Graphiti	0.079	0.162	0.165	0.153	0.140	0.466	0.251	0.272	0.213	0.301
	FOCO (Static)	0.068	0.092	0.152	0.135	0.112	0.232	0.230	0.286	0.200	0.237
	FOCO	0.068	0.119	**0.115**	**0.059**	**0.090**	**0.232**	**0.222**	0.246	**0.091**	**0.198**

Table 5. ECE Score results on Prophet Arena and FutureX across two backbone models.

as more forecasting questions are observed. It also motivates maintaining memory at the subcategory level, so that newly observed groups of related questions can accumulate their own predictive factors and reasoning patterns.

D.7. Bootstrap Confidence Intervals

To assess the robustness of the forecasting results, we conduct a bootstrap analysis over evaluation questions in Table 6. For each dataset and method, we resample forecasting questions with replacement and recompute the mean Brier score. We use 10,000 bootstrap resamples and report the original mean Brier score together with the 95% bootstrap confidence interval, computed from the 2.5th and 97.5th percentiles of the bootstrap distribution. In addition to reporting confidence intervals for each method individually, we conduct a paired bootstrap comparison against the no-memory baseline. For each bootstrap sample, we compute the difference between a method's Brier score and the BASE Brier score on the same resampled set of questions:

$$\Delta_{\text{method}} = \text{Brier}_{\text{method}} - \text{Brier}_{\text{Base}}. \tag{8}$$

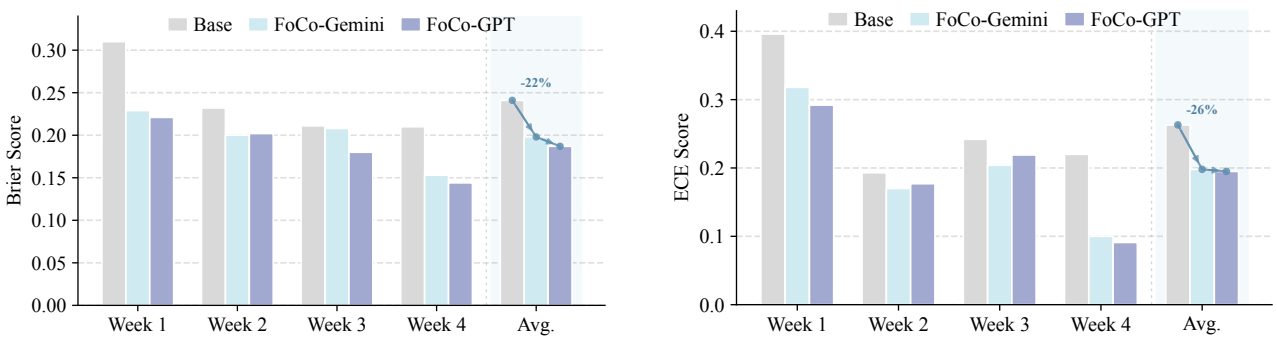


Figure 6. Across-model memory transfer results on FutureX using GPT-5-mini as the backbone. We compare the no-memory baseline, memory transferred from Gemini-2.5-Flash, and memory from GPT-5-mini.

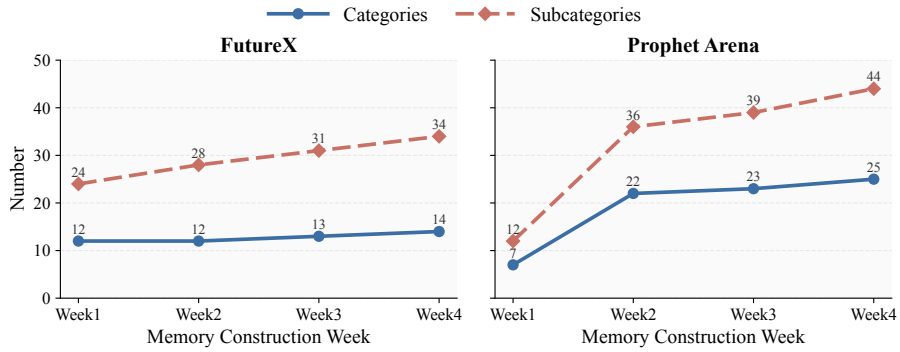


Figure 7. Growth of forecasting taxonomy during memory construction.

The bootstrap analysis supports the robustness of the main results.

D.8. Additional Baseline: Reflexion with Retrospective Trajectory

We further examine whether retrospective trajectories provide useful learning signals for memory-based forecasting. To this end, we introduce a strengthened Reflexion baseline, denoted as REFLEXION + RETRO TRAJECTORY. In contrast to the standard Reflexion baseline, which reflects on the original forecasting trajectory and the resolved outcome, this variant additionally reflects on the retrospective trajectory generated after the outcome is resolved. This comparison tests whether retrospective reasoning can provide useful diagnostic information for future forecasts.

Table 7 shows that retrospective trajectories are indeed valuable. On FutureX, adding retrospective trajectories improves Reflexion from 0.203 to 0.198 in average Brier score and from 0.222 to 0.211 in average ECE. This suggests that retrospective trajectories contain useful information about missing factors, misleading evidence, and calibration errors that can help improve future forecasts.

However, FOCO still achieves the best overall performance. This shows that simply storing retrospective reflections is not sufficient. The benefit of FOCO comes from converting retrospective diagnostic signals into structured, subcategory-level predictive-factor memory and calibration-oriented reasoning memory. In other words, retrospective trajectories provide valuable raw learning signals, while FOCO provides an effective mechanism for abstracting these signals into reusable forecasting knowledge.

D.9. Memory Length Analysis

We also provide the memory statistics for different methods. Table 8 shows that FOCO uses longer memory than the baselines, mainly because it stores both factor memory and reasoning memory. This extra length reflects more structured forecasting knowledge, including predictive signals and calibration rules. The memory remains moderate in size, averaging about 2.2K tokens, and the gains suggest that performance comes from forecasting-specific organization rather than length alone.

Table 6. Bootstrap robustness analysis for Brier score on Prophet Arena.

Dataset	Method	Brier ↓	Δ vs. BASE ↓	p-value
Prophet Arena	BASE	0.1432 [0.1252, 0.1615]	–	–
	MEM0	0.1537 [0.1359, 0.1720]	0.0121 [-0.0130, 0.0364]	0.331
	REFLEXION	0.1435 [0.1255, 0.1627]	-0.0003 [-0.0219, 0.0210]	0.963
	A-MEM	0.1066 [0.0899, 0.1244]	-0.0350 [-0.0599, -0.0101]	0.007
	GRAPHITI	0.1349 [0.1166, 0.1548]	-0.0067 [-0.0314, 0.0179]	0.599
	FOCO	**0.0676 [0.0518, 0.0853]**	**-0.0738 [-0.0923, -0.0560]**	< 0.001

Table 7. Additional comparison with a strengthened Reflexion baseline that reflects on post-hoc trajectories.

Dataset	Method	Week 1		Week 2		Week 3		Week 4		Avg.	
		Brier↓	ECE↓	Brier↓	ECE↓	Brier↓	ECE↓	Brier↓	ECE↓	Brier↓	ECE↓
Prophet Arena	BASE	0.130	0.048	0.135	0.191	0.175	0.100	0.161	0.118	0.150	0.114
	REFLEXION	0.130	**0.036**	0.147	0.078	0.155	0.102	0.167	0.127	0.150	0.086
	REFLEXION + RETRO TRAJECTORY	0.139	0.043	0.136	0.074	0.140	**0.097**	0.137	0.108	0.138	0.081
	FOCO	**0.064**	0.042	**0.069**	**0.064**	**0.092**	0.100	**0.073**	**0.099**	**0.075**	**0.077**
FutureX	BASE	0.310	0.396	0.232	0.193	0.211	0.242	0.210	0.220	0.241	0.263
	REFLEXION	0.246	0.336	0.183	0.185	0.213	0.234	0.170	0.132	0.203	0.222
	REFLEXION + RETRO TRAJECTORY	0.247	0.334	**0.173**	**0.155**	0.222	0.239	0.149	0.116	0.198	0.211
	FOCO	**0.221**	**0.292**	0.202	0.177	**0.180**	**0.219**	**0.144**	**0.091**	**0.187**	**0.195**

D.10. Selection of memory-revision epochs.

Table 9 reports the Brier score and ECE under different numbers of memory-revision epochs, using one resolved week as development and the following week as held-out test. FutureX and Prophet Arena both use Week 0 for development and Week 1 for testing. Figure 8 further shows the held-out Brier–ECE trade-off, where both metrics are lower-is-better. Epoch 2 is selected for FutureX and Epoch 3 for Prophet Arena because they achieve the best held-out accuracy–calibration trade-off and lie on the empirical Pareto frontier.

E. Detailed Algorithms

This appendix provides the detailed procedures omitted from the main text, including initial memory construction and the internal steps of verbalized factor-memory revision. All procedures follow the same chronological constraint as the main pipeline: post-resolution information is used only to revise memory for future forecasts and is never used to modify predictions for the same questions.

Algorithm 1: Initial Memory Construction

Require: Initial taxonomy $\mathcal{T}_0$, first-round no-memory forecasting records $\mathcal{R}_0$
Ensure: Initial memory bank M_0
$\mathcal{T}_0 \leftarrow \mathcal{U}_\theta^{\text{tax}}(\mathcal{T}_0, \mathcal{R}_0)$ ▷ *initialize and expand taxonomy*
for each record $r_i \in \mathcal{R}_0$ with question q_i **do**
  $(c_i, s_i) \leftarrow \mathcal{G}_\theta^{\text{tax}}(q_i, \mathcal{T}_0)$ ▷ *assign record to taxonomy*
end for
for each subcategory $s \in \mathcal{S}(\mathcal{T}_0)$ **do**
  $\mathcal{D}_{0,s}^{\text{res}} \leftarrow \{r_i \in \mathcal{R}_0 : s_i = s\}$ ▷ *collect records in subcategory*
  if $\mathcal{D}_{0,s}^{\text{res}} = \emptyset$ **then**
    continue
  end if
  $M_{0,s} \leftarrow \mathcal{I}_\theta^{\text{mem}}(\mathcal{D}_{0,s}^{\text{res}})$ ▷ *initialize verbal memory*
end for
Return $M_0 = \{M_{0,s} : s \in \mathcal{S}(\mathcal{T}_0)\}$

Method	Avg. chars	Approx. tokens	Median	Std.
FoCo	8754	2188	9557	2566
Factor memory	5518	1379	5871	1637
Reasoning memory	3236	809	3430	981
A-Mem	3212	803	2486	2367
Reflexion	3724	931	3767	211
Mem0	1047	262	976	220
Graphiti	202	50	211	45

Table 8. Memory length statistics across methods. Approximate token counts are estimated from character counts.

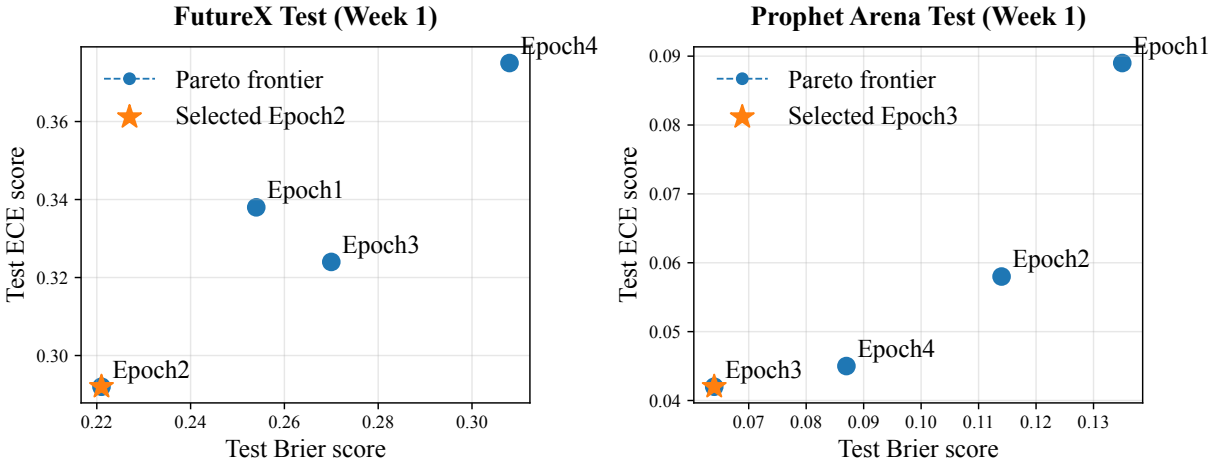


Figure 8. Brier–ECE trade-off for selecting the number of memory-revision epochs. Each point denotes one revision epoch.

The initialization operator $\mathcal{I}_\theta^{\text{mem}}$ summarizes first-round no-memory forecasting records into subcategory-level memory. For each subcategory, it constructs two complementary memory components: predictive-factor memory and calibration-oriented reasoning memory. The initialized memory abstracts recurring forecasting signals and calibration patterns, rather than storing full trajectories or event-level answers.

Algorithm 3 gives the detailed procedure for updating the hierarchical forecasting taxonomy. At week w, the algorithm starts from the previous taxonomy $\mathcal{T}_{w-1}$ and the incoming question set $\mathcal{Q}_w$. Each question is first classified against the existing taxonomy. If the classifier finds a suitable category-subcategory pair, the question is directly assigned to that pair. Otherwise, the question is marked as unmatched and used to construct a candidate subcategory proposal. All unmatched proposals are collected into a proposal pool. The proposal pool is then verified and merged with the previous taxonomy, where redundant, overly narrow, or semantically overlapping proposals are filtered or merged. This produces the updated taxonomy $\mathcal{T}_w$. Finally, all questions are reassigned under $\mathcal{T}_w$ so that every question can be indexed by a subcategory for memory retrieval and revision.

The diagnosis operator $\mathcal{D}_\theta^{\text{mem}}$ compares the original forecast record, the retrospective record, and the realized outcome to identify missing predictive factors, misleading evidence, and calibration errors. The aggregation operator $\mathcal{A}_\theta^{\text{mem}}$ compresses event-level diagnostics into recurring subcategory-level patterns. The revision operator $\mathcal{R}_\theta^{\text{mem}}$ rewrites the memory state into updated predictive-factor memory and calibration-oriented reasoning memory.

The revised memory is constrained to encode transferable forecasting knowledge rather than instance-level answers. Specifically, it should retain recurring predictive signals, factor interactions, failure modes, and confidence-adjustment principles, while excluding copied post-resolution evidence, resolved outcomes, exact event dates, final rankings, and entity-specific conclusions. This constraint allows retrospective records to serve as learning signals for future questions without turning memory into an event-level answer cache.

Dataset	Split	Epoch 1		Epoch 2		Epoch 3		Epoch 4	
		Brier	ECE	Brier	ECE	Brier	ECE	Brier	ECE
FutureX	Train (week 0)	0.213	0.184	0.217	0.182	0.187	0.169	0.190	0.155
FutureX	Test (week 1)	0.254	0.338	0.221	0.292	0.270	0.324	0.308	0.375
Prophet Arena	Train (week 0)	0.114	0.113	0.076	0.111	0.079	0.100	0.079	0.103
Prophet Arena	Test (week 1)	0.135	0.089	0.114	0.058	0.064	0.042	0.087	0.045

Table 9. Effect of memory-revision epochs on development and held-out weeks.

Algorithm 2: FoCo Chronological Update Pipeline

Input: Initial records I_0, weekly questions $\{\mathcal{Q}_w\}_{w=1}^{W}$, revision iterations N.
Output: Final taxonomy $\mathcal{T}_W$ and memory M_W.

$(\mathcal{T}_0, M_0) \leftarrow \text{Initialize}(I_0)$ ▷ *initialize taxonomy and subcategory-level memory*
for $w = 1, \ldots, W$ **do**
  $\mathcal{T}_w \leftarrow \text{TaxUpdate}(\mathcal{T}_{w-1}, \mathcal{Q}_w)$ ▷ *assign questions, propose new subcategories, merge taxonomy*
  for $s \in \mathcal{S}(\mathcal{T}_w)$ **do**
    $\mathcal{Q}_{w,s} \leftarrow \text{Route}(\mathcal{Q}_w, s, \mathcal{T}_w)$ ▷ *select resolved questions in subcategory s*
    $Z_{w,s}^{\text{retro}} \leftarrow A_\theta(\mathcal{Q}_{w,s}, E^{\text{retro}})$ ▷ *obtain retrospective records and realized outcomes*
    $M_{w,s} \leftarrow \text{MemUpdate}(M_{w-1,s}, \mathcal{Q}_{w,s}, Z_{w,s}^{\text{retro}}, N)$ ▷ *diagnose, aggregate, and revise memory*
  end for
end for
$\mathcal{T}_W, M_W$

F. Forecasting Memory Construction Details

F.1. Representative subcategory memories

We provide two representative subcategory-level memories learned by FoCo. Each memory contains two components: *factor memory*, which stores reusable predictive dimensions and their typical effects on forecasts, and *reasoning memory*, which stores calibration, probability-update, and uncertainty-handling lessons.

F.2. Forecasting Question Taxonomy

Table 12 presents partial forecasting question taxonomy used in our analysis on Prophet Arena dataset.

G. Detailed prompts

G.1. Agent Prompts

G.2. Taxonomy-relevant prompt.

Taxonomy classification prompt We use the Prompt in Table 14 to determine whether each forecasting question can be assigned to an existing taxonomy entry. Given the current taxonomy and a new question, the model either matches the question to an existing category–subcategory pair or proposes a new category and subcategory when no suitable match exists. The output is constrained to a fixed JSON schema to ensure consistent downstream taxonomy updates.

Taxonomy revision prompt. We use the prompt in Table 15 to decide whether grouped unmatched questions should be incorporated into the forecasting taxonomy. Given the existing taxonomy and a grouped proposal, the model determines whether the proposal is already covered, should be added as a new subcategory under an existing category, or requires a new top-level category. The output follows a fixed JSON schema to support controlled and consistent taxonomy expansion.

G.3. Memory-relevant prompts

Subcategory memory revision suggestion prompt. We use the prompt in Table 16 to propose the revision for the subcategory-level forecasting memory from paired event trajectories. For each event, the model compares the filtered prediction under realistic forecasting conditions with a no-filter reference and the ground truth, then identifies which parts of the existing memory were useful, missing, overly specific, or misleading. The prompt explicitly separates factor-memory

Algorithm 3: Taxonomy Update

Input: Previous taxonomy $\mathcal{T}_{w-1}$, incoming questions $\mathcal{Q}_w$.
Output: Updated taxonomy $\mathcal{T}_w$ and assignments $\{(c_i, s_i)\}_{q_i \in \mathcal{Q}_w}$.

$\mathcal{P}_w \leftarrow \emptyset$
for $q_i \in \mathcal{Q}_w$ **do**
  $o_i \leftarrow \mathcal{G}_\theta^{\text{tax}}(q_i, \mathcal{T}_{w-1})$ ▷ *match question to existing taxonomy*
  if $o_i = \emptyset$ **then**
    $\tilde{s}_i \leftarrow \text{ProposeSubcategory}_\theta(q_i, \mathcal{T}_{w-1})$
    $\mathcal{P}_w \leftarrow \mathcal{P}_w \cup \{(q_i, \tilde{s}_i)\}$ ▷ *unmatched question induces a proposal*
  end if
end for
$\mathcal{T}_w \leftarrow \mathcal{U}_\theta^{\text{tax}}(\mathcal{T}_{w-1}, \mathcal{P}_w)$ ▷ *merge proposal pool into taxonomy*
for $q_i \in \mathcal{Q}_w$ **do**
  $(c_i, s_i) \leftarrow \mathcal{G}_\theta^{\text{tax}}(q_i, \mathcal{T}_w)$
end for
$\mathcal{T}_w, \{(c_i, s_i)\}_{q_i \in \mathcal{Q}_w}$

Algorithm 4: Detailed Memory Construction and Revision

Require: Taxonomy $\mathcal{T}_w$, previous memory bank M_{w-1}, weekly questions $\mathcal{Q}_w$, forecast records $\{r_i^{\text{fore}}\}$, resolved outcomes $\{y_i\}$, revision iterations N
Ensure: Updated memory bank M_w

Retrospective record construction
for each resolved question $q_i \in \mathcal{Q}_w$ **do**
  Generate retrospective record r_i^{retro} ▷ *uses post-resolution evidence*
  $(c_i, s_i) \leftarrow \mathcal{G}_\theta^{\text{tax}}(q_i, \mathcal{T}_w)$ ▷ *recover taxonomy assignment*
end for
Subcategory grouping
for each subcategory $s \in \mathcal{S}(\mathcal{T}_w)$ **do**
  $I_{w,s}^{\text{res}} \leftarrow \{(q_i, r_i^{\text{fore}}, r_i^{\text{retro}}, y_i) : q_i \in \mathcal{Q}_w,\ s_i = s\}$ ▷ *resolved records for subcategory*
  Verbalized factor-memory revision
  $M_{w,s}^{(0)} \leftarrow M_{w-1,s}$
  for $n = 0, \ldots, N-1$ **do**
    $\Delta_{w,s}^{(n)} \leftarrow \mathcal{D}_\theta^{\text{mem}}(M_{w,s}^{(n)}, I_{w,s}^{\text{res}})$ ▷ *diagnose missing factors and calibration errors*
    $\bar{\Delta}_{w,s}^{(n)} \leftarrow \mathcal{A}_\theta^{\text{mem}}(\Delta_{w,s}^{(n)})$ ▷ *aggregate event-level diagnostics*
    $M_{w,s}^{(n+1)} \leftarrow \mathcal{R}_\theta^{\text{mem}}(M_{w,s}^{(n)}, \bar{\Delta}_{w,s}^{(n)})$ ▷ *revise verbal memory*
  end for
  $M_{w,s} \leftarrow M_{w,s}^{(N)}$
end for
$M_w = \{M_{w,s} : s \in \mathcal{S}(\mathcal{T}_w)\}$

revisions from reasoning and calibration revisions, ensuring that predictive dimensions are updated independently from probability-update rules or confidence-control heuristics.

Batch memory-suggestion summarization prompt. We use the prompt in Table 17 to aggregate per-event memory revision suggestions within the same forecasting subcategory. The model identifies revision signals that recur across multiple events and summarizes them into reusable factor, calibration, and reasoning-failure lessons. One-off event-specific observations are discarded so that the resulting summary can be reliably merged with other batch summaries before the final memory update.

Final subcategory memory revision prompt. We use the prompt in table 18 to produce the final memory update for each forecasting subcategory. The model synthesizes batch-level revision summaries from the current epoch and updates the existing memory only when changes are supported by repeated evidence across events. The prompt enforces conservative factor preservation, separates predictive factors from reasoning and calibration rules, and outputs a structured JSON memory used in later forecasting episodes.

Memory artifact evaluation prompt. We use the following prompt to evaluate the quality of generated memory artifacts. Each artifact is scored along five dimensions: generalizability, novelty, completeness, accuracy, and actionability. The

Representative Memory Example 1: `team_sports_match_winner`	
Factor Memory	**Typical Effect on Output**
Market-implied expectation	Anchors forecasts close to market consensus for liquid events; as liquidity declines or independent model signals strengthen, forecasts move away from market priors with proportional uncertainty.
Event-state and canonical instance mapping	Determines whether the task remains probabilistic or should collapse toward a recorded outcome; ambiguous or resumed instances require branching and retained uncertainty.
Provenance strength, independence, and corroboration count	High-tier, independently corroborated primary sources justify large probability shifts and smaller residuals; single secondary or aggregated reports produce only modest updates.
Evidence freshness, retrievability, and cutoff admissibility	Fresh, provably pre-cutoff primary records produce the strongest updates; post-cutoff or archival-only evidence should not drive pre-cutoff collapse.
Personnel availability, role leverage, and replacement depth	Verified availability or absence of high-leverage personnel often produces the largest conditional shifts; uncertain availability should be scenario-branched.
Venue, travel burdens, and tactical matchup fit	Produces moderate adjustments for home side, travel, venue micro-effects, or tactical fit; strong mismatches can materially alter expectations.
Base rates, competition stage, structural tier, and short-run form	Stabilizes priors near long-run expectations absent strong event-specific evidence; structural tier gaps set upset floors and regularized short-run signals shift posteriors modestly.
Outcome-space completeness and residual finality risk	Prevents implausible absolute certainty by enforcing a documented nonzero residual when collapsing; residual size depends on correction history and corroboration strength.
Reasoning Memory	**Learned Reasoning / Calibration Patterns**
Calibration experiences	Run event-state/provenance preflight before decisive updates; use vig-removed market probabilities as the default prior; branch on high-leverage uncertainties; require provenance metadata for deterministic collapse or large moves; prohibit exact 1.0/0.0 and retain documented nonzero epsilon.
Overconfidence patterns	Avoid collapsing on secondary or social reports without primary provenance; do not treat syndicated reposts as independent corroboration; do not use post-cutoff archival captures or access-time metadata as contemporaneous evidence.
Underconfidence patterns	Do not maintain broad uncertainty after timestamped primary confirmations; avoid defaulting to 50/50 when a documented model or thin-market blend is available; branch when high-quality sources support a decisive scenario.
Probability-update lessons	Use publisher-declared publication time or archived proof for admissibility; scale update magnitude and residual size by source tier and corroboration count; blend market-implied and model-implied priors when markets are thin or divergent; label evidence-limited forecasts and widen uncertainty when authoritative evidence is inaccessible.
Common reasoning failures	Skipping event-state/provenance preflight; counting syndicated reports as independent confirmations; omitting epsilon rationale when collapsing; double-counting information already embedded in market priors.

Table 10. Representative learned memory for the `team_sports_match_winner` subcategory.

evaluator is instructed to return a fixed JSON object with integer scores and a one-sentence rationale.

H. Limitations and Social Impact

Limitations This work studies memory-augmented agentic forecasting under a chronological evaluation protocol. There remain challenges in interpreting and verifying stored memories, especially when memories are abstracted from many prior experiences. Future work could develop more transparent memory inspection and validation mechanisms for forecasting agents.

Societal Impact. Automated forecasting systems may create risks if users over-rely on their predictions, or if forecasts are used in high-stakes settings without appropriate human oversight. Forecasting agents should therefore be deployed with transparency about uncertainty, careful evaluation in the target domain, and safeguards against using generated probabilities as sole decision criteria.

Representative Memory Example 2: `commodity_price_threshold_by_date`	
Factor Memory	**Typical Effect on Output**
Market-implied crude-price expectation	Sustained upward momentum shifts mass toward higher thresholds; elevated volatility increases dispersion and raises tail probabilities on both sides.
Refining capacity, outages, and regional throughput constraints	Large or persistent outages increase the probability of higher thresholds, especially regionally; restored capacity lowers those probabilities.
Demand seasonality and short-term mobility patterns	Rising seasonal demand increases probability of exceeding higher thresholds; falling demand or abrupt mobility drops reduce those probabilities.
Inventory levels and short-term stock draws/builds	Consecutive draws increase probability of higher thresholds, while builds reduce it; inventories moderate upstream shock pass-through.
Institutional reporting alignment and sampling sensitivity	Same-day authoritative postings tightly anchor the distribution; absent or differently averaged series require modeling systematic offsets and larger uncertainty.
Inter-series sampling and averaging differences	Known consistent offsets shift the central estimate slightly and indicate that small gaps may be measurement differences rather than substantive moves.
Cross-source consensus and dispersion	Tight cross-source clustering compresses the distribution and lowers outside-tail probabilities; broad dispersion widens uncertainty and raises tail mass.
Short-window tail/event penetration probability	Credible events increase tail probabilities only if they can affect the target sampling window; otherwise tail mass remains near baseline volatility.
Reasoning Memory	**Learned Reasoning / Calibration Patterns**
Calibration experiences	Anchor tightly on authoritative same-day postings; build one continuous short-term distribution and derive threshold probabilities from it; model cent-scale inter-series offsets as measurement bias; use source clustering or dispersion to tighten or widen uncertainty.
Overconfidence patterns	Avoid near-certainty from a single source without accounting for sampling or rounding; do not stack correlated corroborative signals as independent evidence; avoid large tail probability when clustered authoritative observations contradict it.
Underconfidence patterns	Do not maintain broad uncertainty when same-day authoritative observations exist; avoid diluting strong late-window signals across many thresholds; downweight distal signals that cannot materially affect near-term outcomes.
Probability-update lessons	Large shifts require high-reliability event-specific evidence; explicitly handle strict threshold wording and rounding; cap probabilities for outcomes contradicted by contemporaneous authoritative observations; collapse repeated reports of the same data stream into one effective update.
Common reasoning failures	Confusing reporting artifacts with true price movement near cent-scale thresholds; double-counting outlets that repeat the same source; averaging target and proxy series without correcting timing or sampling differences.

Table 11. Representative learned memory for the `commodity_price_threshold_by_date` subcategory.

Category	Subcategories
Ambiguous or underspecified	Missing resolution criteria
Commodity and financial prices	Commodity price threshold by date
Competitive events	Individual competition winner; Playoff series winner; Team sports match winner
Diplomatic meetings and encounters	Which named leader meets subject
Environmental or meteorological outcomes	Global temperature index thresholds; Weather precipitation thresholds
Geopolitical or foreign policy actions	Sanctions or coercive measures
Government policy or executive actions	Presidential executive orders by deadline; Presidential trade policy action
Legal or judicial outcomes	Arrest or charges by deadline; Lawsuit liability or verdict
Legislative rules or parliamentary procedures	Filibuster threshold or rule change
Legislative votes	Individual legislator vote on nomination
Macroeconomic indicators	Monthly inflation index thresholds
Market chart or popularity outcomes	Chart rank winner; Content engagement threshold; Content release by named artist; Critic aggregate score thresholds; Sales or streams numeric
Monetary policy or central bank actions	Central bank policy rate move by date
Multi-option winner markets	Candidate withdrawal or dropout by date; Corporate leadership appointment; Election margin or vote share thresholds; Political nominee or election winner; Tournament or championship winner
Personal physical challenge	Individual challenge completion by date
Public statements or behavior	Named individual attendance; Phrase mention; Physical action during live stream; Public appearance or sighting; Specific quote or topic
Reality TV elimination	Elimination outcome
Team season outcomes	Regular season win totals; Season win totals banded; Team regular season wins thresholds; Season wins thresholds; Season wins total thresholds; Season total wins prediction; Season wins threshold; Team win total thresholds

Table 12. Partial taxonomy of forecasting question categories and subcategories used in our analysis.

Session 1: Factor Decomposition

```
This is session 1.
Decompose the forecasting task into the key factors you would investigate. The max
number of factors is 5, but use fewer if that seems sufficient. Focus on the most
important factors that will drive the forecast.
If factor memory is provided, treat it as weak prior structure rather than as an
authoritative template.
Use retrieved factor memory only to suggest candidate factors that may be relevant.
You may discard retrieved factors if they do not fit the current case.
Do not simply mirror the retrieved factor list; adapt it to the current question.
For each decomposed factor, also generate a short statement of its typical effect on the
forecast output or probabilities.
For each decomposed factor, also generate a short possible error or reasoning trap that
should be avoided when using that factor.
If factor memory is provided, use it as background context, but do not copy or manually
select an error pattern from it. Infer the most relevant error-to-avoid yourself.
Factor memory may help rank or surface plausible drivers, but it must not by itself
determine the final decomposition or imply strong confidence.
Do not use any tools.
Do not give a final answer or probabilities yet.
Return only valid JSON with a top-level key 'factors'.
'factors' must be a list of objects with keys 'factor_name', 'rationale',
'typical_effect_on_output', and 'potential_common_error_pattern'.
Set 'typical_effect_on_output' to a concise description of how the factor usually shifts
probabilities or changes the forecast.
Set 'potential_common_error_pattern' to a concise possible error to avoid for that
factor, or an empty string only if none is relevant.
Use short reusable factor names rather than full-sentence descriptions.
```

Session 2: Evidence Gathering and Forecasting

```
This is session 2.
Use the prior factor decomposition from the conversation as your search plan.
If reasoning memory is provided, use it as a reusable prior rather than as ground truth.
Treat the retrieved reasoning patterns as explicit guardrails for how to reason about
the task.
You should actively use them to avoid the listed common errors, distribution mistakes,
calibration mistakes, and other reasoning traps.
You CAN ONLY use web search at most {max_search_calls} times total.
You MUST finish the task within {max_turns} turns total for this session.
Budget your turns carefully and keep enough remaining turns to produce the final answer.
If the evidence is already sufficient, stop searching early rather than risking a
max-turns failure.
Do not spend your last available turn on search or extra reasoning; reserve it for the
final JSON answer.
Do not use all searches unless necessary.
Once you have enough evidence, stop searching and return the final JSON immediately.
If reasoning memory is provided in the user input, use it as a checklist for reasoning
and evidence gathering, and explicitly check your forecast against it before answering.
```

Table 13. Prompt templates for the two-session forecasting agent. Session 1 decomposes the forecasting task into predictive factors, while Session 2 uses the factor decomposition and reasoning memory for evidence gathering and probabilistic forecasting.

```
Taxonomy:
{taxonomy_text}

Question:
{question}

Return a JSON object with these keys:
- matched_existing_taxonomy: boolean
- category_name: string or null
- subcategory_name: string or null
- confidence: number between 0 and 1
- rationale: short string
- proposed_category_name: string or null
- proposed_subcategory_name: string or null
- proposed_category_rationale: string or null

Rules:
- If the question fits an existing category and subcategory, set matched_existing_taxonomy=true
and fill category_name/subcategory_name.
- If it does not fit, set matched_existing_taxonomy=false and propose a new category and
subcategory.
- Do not invent multiple options.
- Keep rationale concise.
```

Table 14. Prompt template for matching a forecasting question to an existing taxonomy or proposing a new taxonomy entry.

```
Existing taxonomy:
{taxonomy_text}

Grouped proposed new category candidate:
{proposal_json}

Decide whether this grouped proposal should be promoted into the taxonomy.

Return a JSON object with these keys:
- promote_to_taxonomy: boolean
- target_category_name: string or null
- target_subcategory_name: string or null
- create_new_category: boolean
- create_new_subcategory: boolean
- rationale: string
- category_summarized_patterns: array of strings
- subcategory_summarized_patterns: array of strings
- examples: array of strings

Rules:
- If the proposal is actually covered by an existing category/subcategory, do not
promote it as new.
- If it should map to an existing category but needs a new subcategory, set
create_new_category=false and create_new_subcategory=true.
- If it needs a genuinely new top-level category, set create_new_category=true and
create_new_subcategory=true.
- Use concise snake_case names for category and subcategory.
- Keep at most 3 summarized patterns and at most 3 examples.
- Examples should be selected from the proposal examples when possible.
- Always return valid JSON only.
```

Table 15. Prompt template for deciding whether grouped proposed taxonomy candidates should be promoted into the forecasting taxonomy.

```
You are reviewing one forecasting event to suggest how a subcategory memory should be
revised.

Category: {category_name}
Subcategory: {subcategory_name}
Existing memory: {existing_memory_json}

You are given two bundles for the same event:
1. FILTERED bundle: inference under a date-based search cutoff.
2. NO-FILTER bundle: inference without date restriction, used as an oracle reference.

Filtered event bundle: {filtered_event_bundle}
No-filter reference bundle: {nofilter_event_bundle}
Ground truth: {ground_truth}

Task:
Compare the filtered inference with the no-filter reference and ground truth. Identify
which parts of the existing memory were useful, missing, too concrete, or misleading.
Suggest reusable subcategory-level revisions, not event-specific facts. Separate
factor-memory issues from reasoning and calibration issues.
Return valid JSON only with:
- category_name: string
- subcategory_name: string
- memory_strengths: array of strings, at most 5
- memory_gaps: array of strings, at most 5
- suggested_factor_revisions: array of objects, at most {max_factors}, each with:
  - factor_name: string
  - action: one of ["keep", "revise", "add", "deprioritize"]
  - description: string
  - typical_effect_on_output: string
  - factor_specific_checks: array of 2-4 short strings
  - common_failures: array of 2-4 short strings
- suggested_calibration_revisions: array of strings, at most 6
- suggested_reasoning_failure_revisions: array of strings, at most 6
- event_specific_notes: string
Rules:
Factor memory describes broad predictive dimensions that matter in this subcategory.
Reasoning memory describes how to update probabilities, calibrate confidence, interpret
evidence, and handle uncertainty. Each factor revision must be a broad latent
predictive dimension, not a named entity, search query, raw metric, threshold,
confidence cap, provenance rule, or procedural update rule. Put calibration rules,
confidence-control policies, late-window verification rules, and evidence-strength
thresholds into calibration or reasoning revisions instead.

Suggested factors should remain meaningful even if numeric update rules are removed.
Good factor names resemble market-implied expectation, structural opportunity and
role, institutional signaling, macro trend alignment, timing feasibility, or electorate
alignment. Avoid factor names such as evidence credibility, provenance, update caps,
thresholds, historical priors, regularization, or late-window verification cadence.

Focus especially on calibration: distinguish evidence that only helps rank outcomes
from evidence strong enough to justify extreme confidence. If the no-filter reference
also fails to approach ground truth, note that the missing signal may be genuinely hard
to capture rather than a clear memory gap.

Before finalizing each suggested factor revision, check whether it is truly a
predictive dimension. If it is mainly a reasoning or calibration rule, move it to
suggested_calibration_revisions or suggested_reasoning_failure_revisions.
```

Table 16. Prompt template for revising subcategory-level forecasting memory from filtered and no-filter event bundles.

```
You are summarizing a batch of per-event memory revision suggestions for one forecasting
subcategory.

Category: {category_name}
Subcategory: {subcategory_name}
Existing memory, built from {n_existing_questions} questions: {existing_memory_json}
Per-event revision suggestions for this batch, {batch_size} events:
{event_suggestions_json}

Task:
Identify signals that recur across multiple events in this batch. Summarize recurring
factor revision signals, calibration lessons, and reasoning failures. Discard one-off,
event-specific observations that appear in only one event. Be concise, since this
summary will be combined with summaries from other batches before final memory revision.

Return valid JSON only with:
- batch_size: integer
- recurring_factor_signals: array of objects, at most {max_factors}, each with:
  - factor_name: string
  - action: one of ["keep", "revise", "add", "deprioritize"]
  - frequency: integer
  - consensus_description: string
  - key_checks: array of strings, at most 3
  - key_failures: array of strings, at most 3
- recurring_calibration_signals: array of strings, at most 6
- recurring_reasoning_failure_signals: array of strings, at most 6
- common_memory_gaps: array of strings, at most 4
- common_memory_strengths: array of strings, at most 4

Rules:
Only include factor, calibration, and reasoning-failure signals supported by at
least two events in the batch. Omit one-off event-specific observations. Keep all
descriptions short, reusable, and independent of individual event details.
```

Table 17. Prompt template for summarizing recurring memory-revision signals across a batch of forecasting events.

```
You are revising reusable forecasting memory for one subcategory.

Category: {category_name}
Subcategory: {subcategory_name}
Existing memory, built from {n_existing_questions} questions: {existing_memory_json}
Batch-merged revision summaries from the current epoch, {n_new_questions} new questions:
{event_revision_suggestions_json}

Each suggestion was produced by comparing filtered inference with a no-filter oracle reference and ground truth.

Task:
Synthesize all revision summaries. Revise memory only when supported by multiple events. Keep factors reusable,
high-level, and stable. Preserve useful existing content unless repeated evidence supports revision.

Factor preservation rules:
Every existing factor must be explicitly addressed; KEEP is the default. Modify a factor only if at least
one suggestion explicitly targets it with action="revise". Drop a factor only if at least two suggestions
deprioritize or contradict it and it is not independently useful. If {n_new_questions} < {n_existing_questions},
prefer extending or merging over replacing. Add a factor only if it captures a genuinely absent predictive
dimension. Silent replacement by a renamed equivalent factor is not allowed.

Return valid JSON only with:
- category_name: string
- subcategory_name: string
- revised_common_factors: array of objects, at most {max_factors}, each with factor_name, description,
typical_effect_on_output, factor_specific_checks, and common_failures
- revised_common_reasoning_patterns: object with calibration_experiences, overconfidence_patterns,
underconfidence_patterns, probability_update_lessons, and common_reasoning_failures
- revised_representative_examples: array of strings, at most 5
- revised_notes: string, 1-3 sentences
- update_rationale: string, 1-3 sentences

Core separation:
Factor memory describes broad predictive dimensions. Reasoning memory describes probability updating,
calibration, confidence control, evidence interpretation, and uncertainty handling. Calibration rules,
confidence caps, provenance rules, thresholds, regularization policies, and late-window verification rules must
be placed in revised_common_reasoning_patterns, not revised_common_factors.

Revision rules:
Preserve useful content by default; merge, refine, or extend rather than drop unless repeated evidence supports
removal. Prioritize stable cross-event patterns over one-off anomalies. Each revised factor must be a broad
latent predictive dimension, not a named entity, search query, raw metric, threshold, confidence cap, or
procedural update rule. Good factors resemble market-implied expectation, structural opportunity and role,
institutional signaling, macro trend alignment, timing feasibility, or electorate alignment.

Reasoning rules:
Store cross-factor lessons in revised_common_reasoning_patterns. Focus on calibration: when to stay near base
rates, when weak signals should not be stacked, when large probability shifts require direct high-reliability
evidence, and how to handle sparse, conflicting, or uncertain evidence. Distinguish ranking evidence from
evidence strong enough for extreme confidence.

Final check:
Before finalizing each revised factor, check whether it is truly a predictive dimension. If it is mainly a
reasoning or calibration rule, move it to revised_common_reasoning_patterns.
```

Table 18. Prompt template for final subcategory-level forecasting memory revision.

```
You are an expert evaluator of AI-generated memory artifacts for a probabilistic
forecasting system.  Be concise, critical, and accurate.

A memory artifact is a stored knowledge unit that an AI forecasting agent retrieves to
improve future probability estimates.

Rate the following artifact on 5 dimensions, each scored 1-10:

1.  generalizability - applies broadly across question types, not overfit to specific
examples
2.  novelty - contains non-obvious insights beyond common knowledge
3.  completeness - covers the key factors needed for forecasting this question type
4.  accuracy - factually correct and based on sound reasoning
5.  actionability - directly guides forecasting decisions, e.g., "look for X" or "weight
Y"

Memory artifact:
--
{memory_text}
--

Reply ONLY with valid JSON, with no markdown fences:
{{"generalizability":  <int>, "novelty":  <int>, "completeness":  <int>, "accuracy":
<int>, "actionability":  <int>, "reasoning":  "<one sentence>"}}
```

Table 19. Prompt template for evaluating generated memory artifacts across five quality dimensions.