ForecastCompass: Guiding Agentic Forecasting with Adaptive Factor Memory

Yurui Chang, Yongkang Du, Yuanpu Cao, Jinghui Chen, Lu Lin
The Pennsylvania State University
{yuruic, lulin}@psu.edu

Abstract

Agentic forecasting is important for decision-making in dynamic environments, but it remains challenging because agents must reason from incomplete, time-limited evidence and produce calibrated probabilities before outcomes are resolved. Memory provides a natural mechanism for transferring experience from resolved forecasts to future prediction tasks. However, existing agent-memory methods are not tailored to forecasting, as they typically store past interactions, reflections, or factual associations without explicitly representing reusable predictive factors or calibration knowledge. We propose **Fo**recast**Co**mpass (FoCo), an adaptive factor-based memory framework for agentic forecasting. FoCo organizes forecasting experience with a hierarchical forecasting-task taxonomy, enabling retrieval task-relevant forecasting knowledge. It maintains two complementary memory components: factor memory, which captures reusable predictive dimensions, and reasoning memory, which encodes probability updating, uncertainty handling, and calibration principles. Using retrospective analyses as learning signals, FoCo iteratively revises memory through a verbalized memory-revision procedure, enabling the agent to accumulate transferable forecasting knowledge over time. Experiments on Prophet Arena and FutureX with GPT-5-mini and Gemini-2.5-Flash show that FoCo improves both probabilistic accuracy and calibration. The official code repository for this paper is available at: `https://github.com/yuruic/ForecastCompass`.

1 Introduction

Large language model based agents have emerged as a promising framework for complex real-world tasks that require information seeking [Nakano et al., 2022], tool using [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 and important instance of this setting, since many high-stakes decisions require anticipating uncertain future events, such as market movements [Box et al., 2015, Fama, 1970], policy changes [Tetlock, 2017], and geopolitical developments [Tetlock and Gardner, 2016]. As LLM-based agents become capable of searching the web, using tools, synthesizing evidence, and reasoning over dynamic information, they provide a promising foundation for agentic forecasting [Yao et al., 2022].

Building on this perspective, agentic forecasting can be viewed as a more flexible alternative to classical forecasting pipelines [Hyndman and Athanasopoulos, 2018, Nakano et al., 2022, Yao et al., 2022]. Classical forecasting methods in statistics, machine learning, and decision science typically identify the predictive factors, model their temporal dynamics, and update predictions as new observations become available [Ho and Xie, 1998, Hassan and Nath, 2005, Wu et al., 2021]. However, their modeling process is often constrained by predefined features, structured inputs, and domain-specific assumptions. This limits their flexibility in open-world settings, where relevant evidence may be unstructured, heterogeneous, and distributed across many sources. In contrast, agentic forecasting provides a more flexible paradigm by enabling agents to actively seek information,

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

synthesize diverse evidence, and adapt their reasoning to the forecasting task [Zeng et al., 2025, Yang et al., 2025, Chang et al., 2025].

Given that forecasting fundamentally depends on identifying and modeling the predictive factors, one natural question is: *can agentic forecasters also be guided to reason in a factor-centric manner?* Instead of relying only on generic experiences or event-level forecast outputs, an agent should learn factor-level knowledge from past events and reuse it for future forecasting. Memory provides a natural mechanism for this by storing factor-granular information that can guide factor selection and probability calibration in the future tasks.

However, existing memory mechanisms for LLM agents are largely designed for question-answering or problem-solving settings, where memory helps recover an answer, reuse reasoning traces, or avoid past mistakes [Lewis et al., 2020, Borgeaud et al., 2022, Chhikara et al., 2025]. Forecasting requires memory beyond that used for question answering.

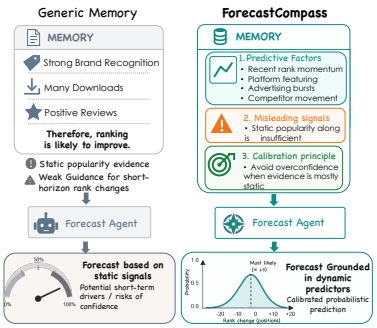


Figure 1: Generic memory v.s. FoCo memory on an example forecasting task.

While QA often relies on retrieving or verifying existing answers, forecasting requires calibrated probabilities for unresolved events under partial and evolving evidence. Moreover, forecasting feedback is noisy and event-specific: a resolved outcome does not directly reveal which predictive factors were missing or misused, and useful lessons may only emerge across multiple trajectories. Forecasting memory must therefore identify recurrent predictive factors, filter misleading or event-specific hindsight, and refine calibration principles. For example, in an app-ranking forecast in Figure 1, a generic memory may imply improved ranking based on stored trace such as "`the app has strong brand recognition, many downloads, and positive reviews.`" While useful, this trace does not identify which signals are informative for short-horizon rank changes. A forecasting-specific memory should instead capture dynamic predictive factors such as recent rank momentum, platform featuring, advertising bursts, and competitor movement, while discouraging overconfidence from static popularity evidence. This motivates a *forecasting-specific memory design* that abstracts reusable predictive factors, misleading signals, and calibration principles from retrospective feedback.

To address these challenges, we propose FoCo (**Fo**recast**Co**mpass), the first forecasting-specific memory framework that organizes past forecasting experience into hierarchical structured memory for future probabilistic prediction. Rather than storing full trajectories or instance-level conclusions, FoCo maintains two complementary memory components at the subcategory level: *predictive-factor memory*, which captures reusable signals, misleading evidence patterns, and factor-level lessons, and *calibration-oriented reasoning memory*, which captures how confidence should be adjusted under different evidence conditions. At inference time, the agent retrieves relevant memory to guide evidence search, factor selection, and probability estimation for new unresolved questions. To construct and update this memory, FoCo performs contrastive memory revision over resolved forecasting tasks. It contrasts forecasting trajectories with retrospective trajectories, uses their discrepancies as diagnostic signals, and revises the hierarchical memory into reusable predictive factors and calibration principles while avoiding event-level hindsight. Experiments on dynamic forecasting benchmarks show that FoCo improves both accuracy and calibration, with learned memory transferring across time periods and model backbones. Our main contributions are summarized as follows:

- We formulate *factor-centric memory* for agentic forecasting, where memory is organized as subcategory-level predictive factors and calibration-oriented reasoning patterns, rather than event-level trajectories or generic reflections.
- We develop *verbalized factor-memory revision*, a diagnose–aggregate–revise procedure that identifies recurrent lessons across forecasting trajectories and revises memory into transferable predictive-factor and calibration-oriented abstractions while filtering event-specific hindsight.
- We conduct experiments on dynamic forecasting benchmarks including Prophet Arena and FutureX, showing that FoCo improves both forecasting accuracy and calibration. We further analyze its transferability across time and models, demonstrating that forecasting memory provides reusable guidance for future unresolved questions.

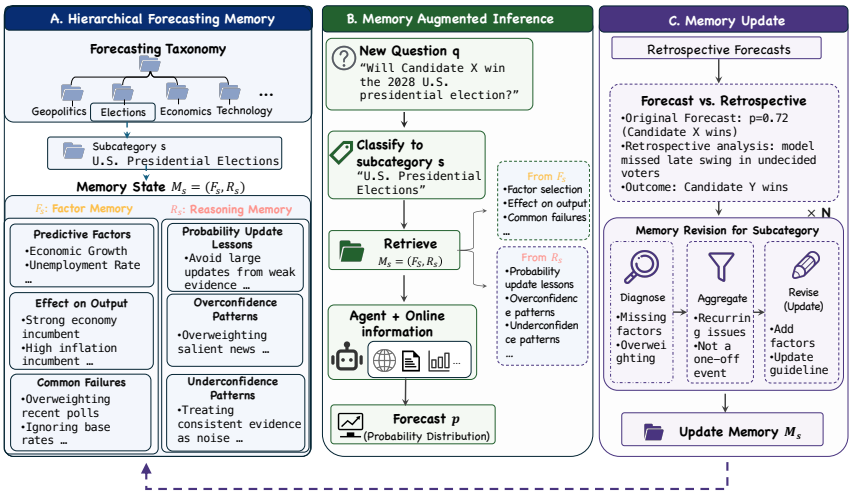


Figure 2: Framework overview. *A*: the hierarchical forecasting memory organizes subcategory-level factor memory and reasoning memory. *B*: a new question is classified into a subcategory, relevant memory is retrieved, and the agent produces a calibrated forecast. *C*: the memory is updated by contrasting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

2 Preliminaries

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

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

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.

3 Method

We introduce FoCo, a framework that builds forecasting memory from retrospective revisions to predictive factors. FoCo organizes memory by a hierarchical taxonomy and subcategory-level states, uses it to guide forecasting, and updates it chronologically as shown in Algorithm 2. Detailed algorithms are provided in Appendix C.

3.1 Hierarchical Forecasting Memory Representation

FoCo organizes experience with a hierarchical taxonomy. At the update step w, the taxonomy T_w consists of a set of categories and fine-grained subcategories. Each forecasting question is associated with one subcategory, and each subcategory serves as the key to a corresponding memory state. For each subcategory $s \in T_w$, FoCo maintains a verbal memory state

$$M_{w,s} = (F_{w,s}, R_{w,s}) \tag{2}$$

where $F_{w,s}$ denotes the predictive-factor memory and $R_{w,s}$ denotes calibration-oriented reasoning memory. The predictive-factor memory $F_{w,s}$ is a set of L_s factor entries,

$$F_{w,s} = \{f_{w,s}^{(1)}, \ldots, f_{w,s}^{(L_s)}\}, \tag{3}$$

where each entry $f_{w,s}^{(i)}$, $i = 1, \ldots, L_s$ describes a reusable predictive factor for subcategory s. Each factor entry contains the factor name, factor-specific evidence checks, common failure modes, and

the factor's typical effect on the forecast probability. These entries guide the agent to identify relevant evidence, avoid misleading uses of individual factors, and estimate how each factor should influence the output distribution.

The reasoning memory $R_{w,s}$ stores calibration-oriented principles that guide how evidence strength, uncertainty, conflicting signals, and missing information should be translated into probabilistic forecasts. Therefore, $M_{w,s}$ is indexed by the forecasting taxonomy and abstracts resolved forecasting records into reusable subcategory-level forecasting principles rather than event-specific conclusions.

3.2 Memory-Augmented Forecasting Inference

For each unresolved question q_i at update step w, FOCO first assigns it to a category-subcategory pair (c_i, s_i) under the current taxonomy $\mathcal{T}_w$. It then retrieves the corresponding subcategory memory M_{w,s_i} and provides it to the forecasting agent as structured guidance.

Given q_i, the accessible evidence pool E_i, and the retrieved memory M_{w,s_i}, the agent searches or uses tools to gather evidence, conditions its reasoning on the memory, and outputs a forecasting trajectory and probability distribution:

$$(z_i^{\text{FoCo}}, p_i^{\text{FoCo}}) = A_\theta(q_i, E_i, M_{w,s_i}), \qquad p_i^{\text{FoCo}} \in \Delta^{K_i-1}. \tag{4}$$

Here, z_i^{FoCo} denotes the memory-augmented forecasting trajectory, and p_i^{FoCo} denotes the predicted probability distribution over the K_i candidate outcomes.

The retrieved memory guides inference in two complementary ways. The factor memory F_{w,s_i} guides evidence search and factor selection by emphasizing predictive signals and warning against misleading evidence patterns. The reasoning memory R_{w,s_i} guides probability estimation by providing calibration principles for handling uncertainty, conflicting evidence, and incomplete information.

3.3 Taxonomy and Memory Update Mechanism

After questions from update step w are resolved, FOCO updates the forecasting taxonomy and subcategory-level memories using only past resolved questions, preventing future inference from accessing post-resolution information for unresolved questions.

Taxonomy update. Because the taxonomy indexes subcategory memories, it is updated as new forecasting patterns emerge. Let $T_{w-1} = \{(c, s) : c \in \mathcal{C}_{w-1}, s \in \mathcal{S}_{w-1,c}\}$ denote the previous taxonomy. Given newly resolved questions Q_w, the model assigns each question to an existing category–subcategory pair when possible; otherwise, it places the question in an unmatched pool.

For unmatched questions, the model proposes new category–subcategory pairs, verifies and consolidates them, merges overlapping proposals with existing subcategories, and retains only reusable forecasting patterns. The resulting consolidated candidate set is denoted as

$$\Delta T_w = (\Delta\mathcal{C}_w, \Delta\mathcal{S}_w), \tag{5}$$

where $\Delta\mathcal{C}_w$ contains newly introduced categories and $\Delta\mathcal{S}_w$ contains newly introduced subcategories, either under new categories or under existing categories. The taxonomy is then expanded by union:

$$T_w = T_{w-1} \cup \Delta T_w. \tag{6}$$

Memory update. For each resolved question $q_i \in Q_w$, FOCO first routes it to a subcategory s_i under the updated taxonomy T_w and retrieves the corresponding memory M_{w-1,s_i}. Only retrieved memories are updated; memories that are not activated by any resolved question remain unchanged. Let $\mathcal{S}_w^{\text{act}} = \{s_i : q_i \in Q_w\}$ denote the set of activated subcategories at update step w. For each $s \in \mathcal{S}_w^{\text{act}}$, let $Q_{w,s}$ denote the resolved questions routed to s.

After the questions are resolved, FOCO constructs retrospective trajectories as learning signals. For each $q_i \in Q_{w,s}$, the agent is given the post-resolution evidence, and generates a retrospective trajectory

$$z_i^{\text{retro}} = A_\theta^{\text{retro}}(q_i, E_i^{\text{retro}}). \tag{7}$$

The retrospective trajectory is not used as a deployable forecast; instead, it explains which evidence and reasoning steps would have supported the realized outcome.

FOCO then contrasts the original forecasting trajectory z_i with the retrospective trajectory z_i^{retro} to obtain a discrepancy signal:

$$\Delta_i = \text{Contrast}(z_i, z_i^{\text{retro}}). \tag{8}$$

Each discrepancy signal contains two components,

$$\Delta_i = (\Delta_i^F, \Delta_i^R), \tag{9}$$

where Δ_i^F captures factor-level discrepancies, such as missing predictive factors, overused misleading signals, or incorrect factor effects, and Δ_i^R captures reasoning-level discrepancies, such as miscalibration, overconfidence, underconfidence, or incorrect probability updates. The LLM aggregates these discrepancies into final memory-revision suggestions, which are then applied to revise the corresponding memory state. The finalized memory revision suggestions are:

$$\mathcal{U}_i = \left(\bigcup_{f \in \widetilde{F}_{i,s_i}} \{\text{Revise}_F(f, \bar{\Delta}_i^F)\},\ \text{Revise}_R(R_{w-1,s_i}, \bar{\Delta}_i^R) \right), \tag{10}$$

Here, $\widetilde{F}_{i,s_i}$ denotes the subset of factor-memory entries in subcategory s_i that are selected for revision by the aggregated factor-level discrepancy $\bar{\Delta}_i^F$. In other words, FOCO does not revise the entire factor memory F_{w-1,s_i}; it first identifies the factors that are directly affected by the contrast between the original forecasting trajectory and the retrospective trajectory, and only updates this selected subset.

After N iterations of the diagnose–aggregate–revise procedure, the resulting memory is stored as $M_{w,s}$. Forecasting feedback is inherently noisy and event-specific, while reliable predictive factors and calibration patterns are often distributed across multiple resolved forecasts. The iterative procedure is therefore designed to filter spurious or one-off hindsight signals while aggregating recurring evidence into robust, transferable forecasting knowledge. This prevents noisy retrospective information from accumulating in memory and allows useful predictive patterns to be progressively reinforced and refined.

4 Experiments

4.1 Experimental settings

We evaluate FOCO with language model based agents on dynamic forecasting benchmarks, where agents reason over predictive factors, gather evidence, and assign calibrated probabilities to future outcome.

Models. We use GPT-5-mini [Singh et al., 2025] and Gemini-2.5-Flash [Comanici et al., 2025] as the backbones of the forecasting agent respectively. Unless otherwise specified, memory construction and revision are performed with the same backbone used for forecasting. To test whether the learned memory is tied to a specific model, we also evaluate cross-model transfer by applying memory constructed with Gemini-2.5-Flash to GPT-5-mini. More setup details are provided in Appendix B.1.

Datasets. We evaluate FOCO on two dynamic forecasting datasets: Prophet Arena [Yang et al., 2025] and FutureX [Zeng et al., 2025]. Both datasets contain time-evolving forecasting questions across a broad range of domains, such as sports, technology, and finance. For each dataset, we select the latest five weeks of data to construct, update, and evaluate memory. This setting reflects the dynamic nature of agentic forecasting, where memory is built from recent resolved questions and revised as new weekly outcomes become available. We follow a chronological evaluation protocol: memory is constructed only from previously resolved questions and is used to forecast future unresolved questions. This prevents post-resolution information from leaking into the current forecast.

Evaluation metrics. We use Brier score [Glenn et al., 1950] and expected calibration error (ECE) [Yang et al., 2025] to evaluate the forecasting performance. Brier score evaluates probabilistic accuracy by measuring the distance between the predicted distribution and the realized outcome. ECE evaluates calibration by comparing predicted confidence with empirical accuracy across confidence bins. Both metrics are lower-is-better. Formal definitions are provided in Appendix A.1.

Baselines. We compare FOCO with baselines chosen to cover a broad range of forecasting and memory-design alternatives. **Reference baselines** include BASE, which removes external memory,

and BASE (Retro), a non-deployable retrospective diagnostic reference that analyzes each question after outcome resolution. **Memory-evolution ablations** include FOCO (Static), which removes weekly memory updates and isolates the value of iterative revision. **Memory-mechanism baselines** are further grouped by the memory mechanism they emphasize: *structured memory*, represented by GRAPHITI [Rasmussen et al., 2025], which stores evolving entity–relation graphs; *retrieval-based long-term memory*, represented by MEM0 [Chhikara et al., 2025], which stores and retrieves long-term interaction memories; and *reflective or experience-driven memory*, represented by REFLEXION [Shinn et al., 2024] and A-MEM [Xu et al., 2025], which construct verbal lessons or agent-experience memories from past trajectories. Together, these baselines provide broad coverage of non-memory, retrospective, static-memory, and representative agent-memory settings. We provide detailed adaptation protocols in Appendix A.2, including what information each baseline stores, whether it uses taxonomy information, and how it is updated weekly.

4.2 Main Results

Table 2 reports week-by-week Brier scores on Prophet Arena and FutureX, and Table 1 summarizes the average Brier and ECE scores across all evaluation weeks. Full week-by-week ECE results are provided in Appendix B.3. The evolution of the taxonomy, with increasing numbers of categories and subcategories as more events are incorporated, is provided in Appendix B.5. We additionally report bootstrap confidence intervals for Brier score in Appendix B.7 to assess robustness of different methods. To further verify that these gains are not specific to a short evaluation window, we extend the FutureX evaluation with six additional weeks (Appendix B.2), where FOCO continues to outperform all baselines in both accuracy and calibration.

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.

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 calibration, 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.

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**

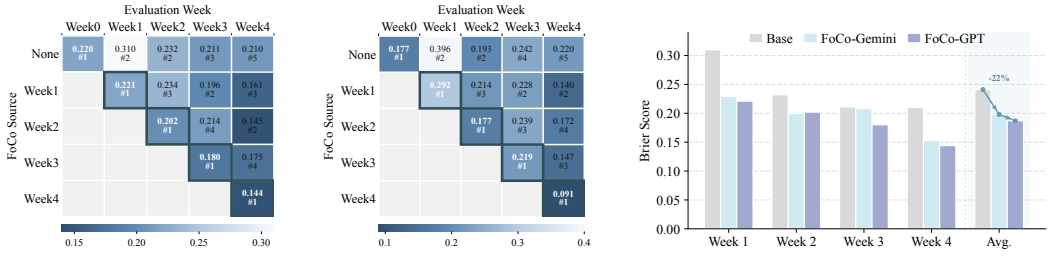
Table 2: Brier Score(↓) results on Prophet Arena and FutureX across different models.

4.3 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 3b and Figure 3a 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.



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

Figure 3: Transferability analysis of FOCO on FutureX. We evaluate both across-time transferability and across-model transferability.

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 3c shows that FOCO provides consistent gains across different backbone models on Brier scores. Full results including the ECE scores are in the Appendix B.4. 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.

4.4 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.

4.5 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 E.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.

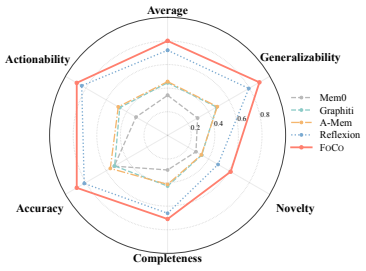


Figure 4: Memory quality comparison on Prophet Arena using LLM-as-a-Judge .

4.6 Case Study

Figure 5 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 D.1 and report an LLM-as-a-judge evaluation of memory quality in Section 4.5.

5 Related Work

Forecasting. Forecasting aims to make probabilistic predictions about future events under incomplete and evolving evidence [Hyndman and Athanasopoulos, 2018, Tetlock and Gardner, 2016]. Classical approaches typically rely on structured historical data, predefined variables, or domain-specific assumptions, such as time-series models [Ho and Xie, 1998], probabilistic models [Hassan and Nath, 2005, Wu et al., 2021], and expert or crowd forecasting systems [Tetlock, 2017, Wolfers and Zitzewitz, 2004]. In contrast, language-model-based forecasting agents operate in open-world information

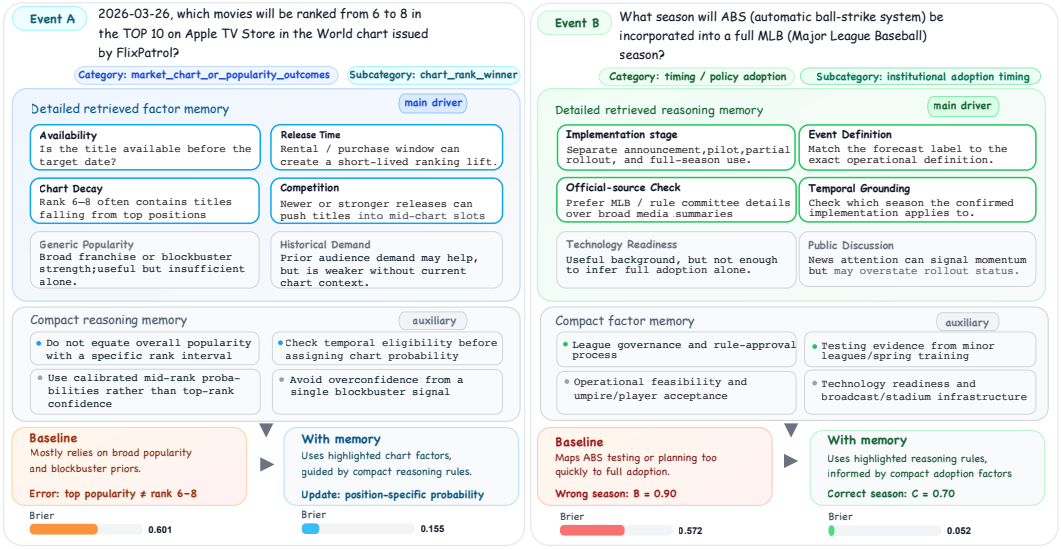


Figure 5: Case Study of how FoCo improves forecasts through factor selection and probability calibration.

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, Ye 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, Yu et al., 2026, Alur 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, Zhang 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 and 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.

6 Conclusion

This work studies how memory should be designed for agentic forecasting, where agents must make calibrated probabilistic predictions from incomplete and evolving evidence. We show that useful forecasting memory should go beyond storing past trajectories and instead abstract reusable predictive factors and calibration principles. FoCo realizes this idea through hierarchical predictive-factor memory and calibration-oriented reasoning memory, revised by contrasting forecasting trajectories with retrospective trajectories from resolved tasks. Experiments on dynamic forecasting benchmarks demonstrate improvements in probabilistic accuracy and calibration, with learned memory transferring across time periods and model backbones.

7 Acknowledgements and Disclosures

The work is supported by the NVIDIA Academic Grant Program. The authors declare no competing interests, including financial relationships outside the submitted work that could be perceived as influencing this research.

References

Rohan Alur, Bradly C. Stadie, Daniel Kang, Ryan Chen, Matt McManus, Michael Rickert, Tyler Lee, Michael Federici, Richard Zhu, Dennis Fogerty, Hayley Williamson, Nina Lozinski, Aaron Linsky, and Jasjeet S. Sekhon. Aia forecaster: Technical report, 2025. URL `https://arxiv.org/abs/2511.07678`.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022.

George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers. *arXiv preprint arXiv:2305.17126*, 2023.

Ching Chang, Yidan Shi, Defu Cao, Wei Yang, Jeehyun Hwang, Haixin Wang, Jiacheng Pang, Wei Wang, Yan Liu, Wen-Chih Peng, et al. A survey of reasoning and agentic systems in time series with large language models. *arXiv preprint arXiv:2509.11575*, 2025.

Yurui Chang, Yiran Wu, Qingyun Wu, and Lu Lin. Memcollab: Cross-agent memory collaboration via contrastive trajectory distillation. *arXiv preprint arXiv:2603.23234*, 2026.

Mingyue Cheng, Xiaoyu Tao, Qi Liu, Ze Guo, and Enhong Chen. Position: Beyond model-centric prediction–agentic time series forecasting. *arXiv preprint arXiv:2602.01776*, 2026.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, 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.

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

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

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

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

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

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

Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback, 2022. *URL https://arxiv. org/abs/2112.09332*, 35, 2022.

Charles Packer, Vivian Fang, Shishir_G Patil, Kevin Lin, Sarah Wooders, and Joseph_E Gonzalez. Memgpt: towards llms as operating systems. 2023.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22, 2023.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems*, 37: 126544–126565, 2024.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: a temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*, 2025.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. *URL https://arxiv. org/abs/2303.11366*, 8, 2024.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

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

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

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

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

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

Yinjun Wu, Jingchao Ni, Wei Cheng, Bo Zong, Dongjin Song, Zhengzhang Chen, Yanchi Liu, Xuchao Zhang, Haifeng Chen, and Susan B Davidson. 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, pages 651–659, 2021.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025.

Qingchuan Yang, Simon Mahns, Sida Li, Anri Gu, Jibang Wu, and Haifeng Xu. Llm-as-a-prophet: Understanding predictive intelligence with prophet arena. *arXiv preprint arXiv:2510.17638*, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Xiao Ye, Jacob Dineen, Evan Zhu, Shijie Lu, Kevin Song, and Ben Zhou. Hindcast: Replaying prediction markets to evaluate llm forecasters. *arXiv preprint arXiv:2607.14051*, 2026.

Haofei Yu, Yining Zhao, Guanyu Lin, and Jiaxuan You. Building social world models with large language models, 2026. URL `https://arxiv.org/abs/2606.11482`.

Zhiyuan Zeng, Jiashuo Liu, Siyuan Chen, Tianci He, Yali Liao, Yixiao Tian, Jinpeng Wang, Zaiyuan Wang, Yang Yang, Lingyue Yin, et al. Futurex: An advanced live benchmark for llm agents in future prediction. *arXiv preprint arXiv:2508.11987*, 2025.

Yaolun Zhang, Yiran Wu, Yijiong Yu, Qingyun Wu, and Huazheng Wang. Live-evo: Online evolution of agentic memory from continuous feedback, 2026. URL `https://arxiv.org/abs/2602.02369`.

Zeyu Zhang, Quanyu Dai, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. A survey on the memory mechanism of large language model-based agents. *ACM Transactions on Information Systems*, 43(6):1–47, 2025.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19632–19642, 2024.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 19724–19731, 2024.

A Additional Experimental Details

A.1 Evaluation Metrics

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

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

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.

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.

A.2 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 E.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.

B Additional Experiments

B.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 B.10.

B.2 Additional results on more FutureX dataset

	W5		W6		W7		W8		W9		W10		Average	
Method	Brier	ECE	Brier	ECE	Brier	ECE	Brier	ECE	Brier	ECE	Brier	ECE	Brier	ECE
BASE (Retro)	0.215	0.184	0.242	0.189	0.211	0.159	0.206	0.177	0.179	0.196	0.173	0.154	0.204	0.177
BASE	0.225	0.186	0.225	0.253	0.218	0.179	0.230	0.224	0.224	0.216	0.237	0.247	0.227	0.218
Reflexion	0.251	0.198	0.211	0.154	0.214	0.151	0.202	0.210	0.224	0.199	0.218	0.216	0.220	0.188
FOCO	0.209	0.166	0.201	0.136	0.198	0.119	0.183	0.169	0.209	0.176	0.178	0.198	0.196	0.161

Table 4: Results on six additional FutureX weeks (W5–W10) using GPT-5-mini.

To assess whether the observed improvements persist over time, we collected six additional weeks of FutureX data following Week 4 (W5–W10). We evaluated FOCO and the three baselines, Base(Retro), Base, and Reflexion, using GPT-5-mini. Table 4 reports weekly Brier scores and expected calibration errors (ECE), together with their averages across the six weeks.

FOCO achieves the lowest average Brier score and ECE among all evaluated methods. These results demonstrate consistent improvements over Base and Reflexion across the additional weeks, as well as better average performance than all three baselines. We restrict this extended-week analysis to FutureX because it is a live, continuously updated benchmark that had additional resolved weeks available at the time of writing.

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.

B.3 Additional results on ECE 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.

B.4 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.

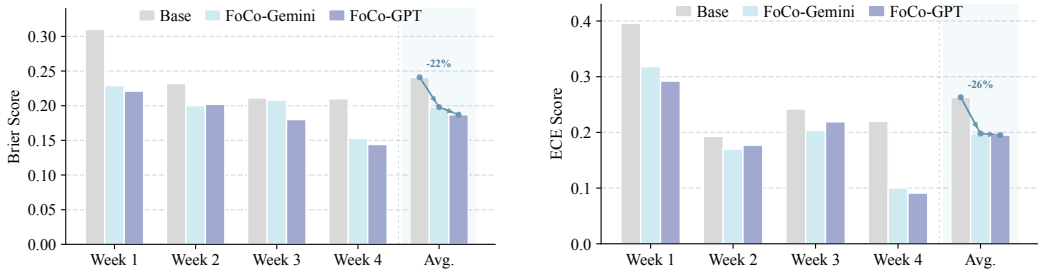


Figure 6: Across-model memory transfer results on FutureX using GPT-5-mini as the backbone. We compare the no-memory baseline, memory transferred from Gemini-2.5-Flash, and memory from GPT-5-mini.

B.5 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 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.

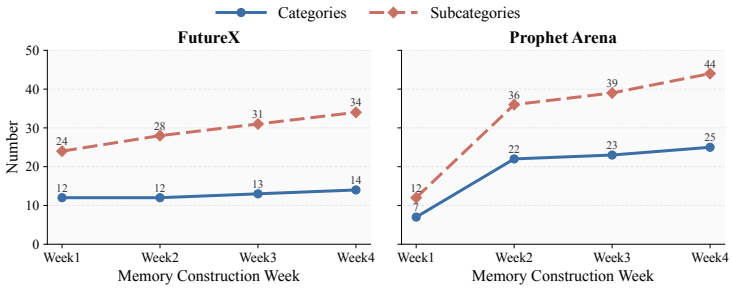


Figure 7: Growth of forecasting taxonomy during memory construction.

Taxonomy saturation on extended weeks. To examine whether taxonomy expansion continues indefinitely or converges as a benchmark's topic space is covered, we extend the taxonomy-evolution analysis to the six additional FutureX weeks (W5–W10). Table 6 reports the number of categories and subcategories at each week. Category count plateaus at 21, which suggests that taxonomy growth saturates as recurring forecasting topics are progressively captured, rather than continuing to expand near-linearly. We attribute this saturation in part to the consolidation step already built into taxonomy update (Section 3), which merges newly proposed subcategories into existing ones before any new entry is created, rather than treating growth as unconstrained. This analysis is nonetheless bounded by the benchmark's finite question set and temporal horizon. Our current mechanism only deduplicates *at creation time*; it does not retroactively merge or retire existing subcategories that may become redundant or inactive over a much longer deployment horizon. Extending taxonomy update with such retroactive consolidation is a natural direction for future work.

	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10
Categories	15	19	19	21	21	21
Subcategories	37	46	46	51	52	53

Table 6: Taxonomy size on FutureX (W5–W10), extending the analysis in Figure 7.

B.6 Evaluation of Task Classification

Prophet Arena and FutureX do not provide ground-truth task categories. We therefore assess routing quality using an independent model, Gemini-2.5-Flash, to judge the subcategory assignments produced by GPT-5-mini. Table below reports the model-judged routing accuracy on Prophet Arena across four weeks.

	Evaluation period			
Metric	Week 1	Week 2	Week 3	Week 4
Accuracy (%)	94.1	87.7	91.8	93.0

Table 7: Classification accuracy on Prophet Arena, judged by Gemini-2.5-Flash for assignments made by GPT-5-mini.

These results suggest that the routing assignments are largely accurate across the evaluated weeks. However, this evaluation reflects an independent model's judgments rather than agreement with ground-truth category labels.

B.7 Bootstrap Confidence Intervals

To assess the robustness of the forecasting results, we conduct a bootstrap analysis over evaluation questions in Table 8. For each dataset and method, we resample forecasting questions with replace-

ment 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:

$$\Delta_{\text{method}} = \text{Brier}_{\text{method}} - \text{Brier}_{\text{Base}}. \tag{11}$$

Dataset	Method	Brier ↓	Δ vs. BASE ↓	p-value
FutureX	BASE	0.2253 [0.1913, 0.2615]	–	–
	MEM0	0.2056 [0.1725, 0.2399]	-0.0197 [-0.0643, 0.0294]	0.457
	REFLEXION	0.2060 [0.1730, 0.2397]	-0.0193 [-0.0496, 0.0095]	0.194
	A-MEM	0.2103 [0.1781, 0.2443]	-0.0150 [-0.0598, 0.0338]	0.592
	GRAPHITI	0.2314 [0.1953, 0.2679]	0.0061 [-0.0346, 0.0483]	0.759
	FOCO	**0.1990 [0.1724, 0.2273]**	**-0.0263 [-0.0443, -0.0020]**	< 0.001
Prophet Arena	BASE	0.1432 [0.1252, 0.1615]	–	–
	MEM0	0.1537 [0.1359, 0.1720]	0.0105 [-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]	-0.0366 [-0.0599, -0.0101]	0.007
	GRAPHITI	0.1349 [0.1166, 0.1548]	-0.0083 [-0.0314, 0.0179]	0.599
	FOCO	**0.0676 [0.0518, 0.0853]**	**-0.0756 [-0.0923, -0.0560]**	< 0.001

Table 8: Bootstrap robustness analysis for Brier score on Prophet Arena and FutureX dataset.

The bootstrap analysis supports the robustness of the main results.

B.8 Additional Baseline: Reflexion with Retrospective Trajectory

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**

Table 9: Additional comparison with a strengthened Reflexion baseline that reflects on post-hoc trajectories.

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 9 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

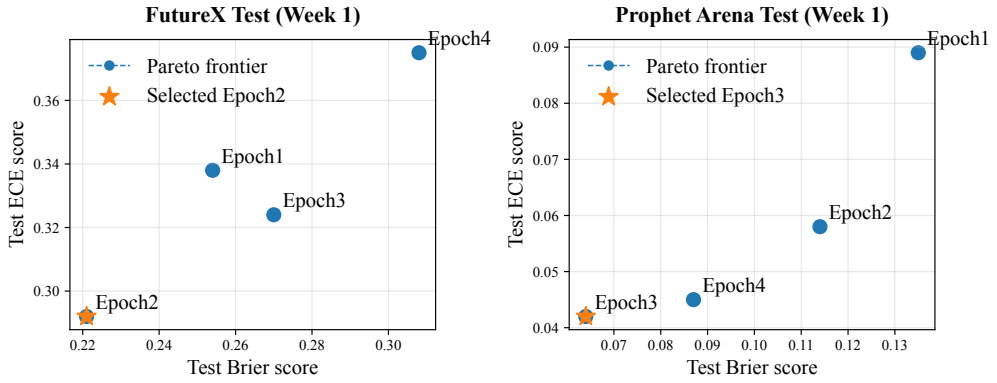


Figure 8: Brier–ECE trade-off for selecting the number of memory-revision epochs. Each point denotes one revision epoch.

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.

B.9 Cost Analysis

Method	Factor decomposition	Forecasting reasoning	Total
Base	696	3,301	3,997
Reflexion	690	3,067	3,757
FoCo	751	3,514	4,265

Table 10: Average output-token usage for factor decomposition and forecasting reasoning. Total denotes the sum across both stages.

Table 10 reports the average output-token usage for factor decomposition and forecasting reasoning. FoCo achieves substantial forecasting improvements with a modest increase in generated output relative to BASE and Reflexion. The component ablations further support the contribution of both memory types: removing either factor memory or reasoning memory degrades forecasting performance, while the full method performs best. Together, these results demonstrate the effectiveness of combining factor and reasoning memory at a modest additional generation cost.

B.10 Selection of memory-revision epochs.

We adopt a chronological evaluation protocol in which Week 0 is used to initialize memory and Week 1 serves as a validation period for selecting the number of memory-revision epochs. Table 11 reports Brier scores and ECE across different epoch counts on these two weeks. Figure 8 illustrates the validation Brier–ECE trade-off, with lower values indicating better performance on both metrics. We select two revision epochs for FutureX and three for Prophet Arena, as these settings achieve the lowest validation Brier score and ECE among the evaluated configurations. The selected epoch count is fixed for all subsequent weeks, while memory continues to be updated using questions whose outcomes have become available before each forecast. Week 1 results are reported as validation performance, and subsequent weeks are used to evaluate forecasting performance under the fixed hyperparameter setting.

Why not validate each memory edit locally. An alternative to our aggregation-based revision would be to validate each individual memory edit against a held-out set of resolved questions before accepting it. We find this is not reliable in our chronological setting, where the topic distribution of resolved questions shifts substantially from week to week. Table 12 reports validation performance on a held-out 20% split of Week 4 alongside subsequent Week 5 performance, for memories revised with different numbers of epochs: the epoch favored by this local validation (Epoch 1) is not the epoch that performs best on Week 5 (Epoch 2), showing that acceptance decisions from a single-week

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 11: Effect of memory-revision epochs on development and held-out weeks.

validation split can overfit to transient topic composition rather than reflecting genuine memory quality. This is consistent with prior memory-augmented agents such as Reflexion [Shinn et al., 2024] and ExpeL [Zhao et al., 2024], which likewise revise textual memory without independently validating each edit, relying instead on aggregate downstream performance. FoCo mitigates noisy or erroneous updates through aggregation across multiple resolved trajectories and continuous, non-append-only revision, which lets earlier inaccuracies be corrected as new evidence arrives. This is consistent with the absence of accumulated degradation observed across repeated chronological updates in our experiments.

Revision Epoch	Week 4 Validation Brier↓	Week 4 Validation ECE↓	Week 5 Brier↓	Week 5 ECE↓
Epoch 1	**0.142**	**0.152**	0.240	0.200
Epoch 2	0.168	**0.152**	**0.209**	**0.166**
Epoch 3	0.183	0.160	0.232	0.185

Table 12: Local validation on a held-out 20% split of FutureX Week 4, versus subsequent Week 5 performance, across memory-revision epochs.

C 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 $M_0 = \{M_{0,s} : s \in \mathcal{S}(\mathcal{T}_0)\}$

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

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
return $\mathcal{T}_W, M_W$

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
return $\mathcal{T}_w, \{(c_i, s_i)\}_{q_i \in \mathcal{Q}_w}$

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.

D Forecasting Memory Construction Details

D.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.

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
return $M_w = \{M_{w,s} : s \in \mathcal{S}(\mathcal{T}_w)\}$

D.2 Forecasting Question Taxonomy

Table 15 presents partial forecasting question taxonomy used in our analysis on Prophet Arena dataset.

E Detailed prompts

E.1 Agent Prompts

E.2 Taxonomy-relevant prompt.

Taxonomy classification prompt We use the Prompt in Table 17 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 18 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.

E.3 Memory-relevant prompts

Subcategory memory revision suggestion prompt. We use the prompt in Table 19 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 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 20 to aggregate per-event memory revision suggestions within the same forecasting subcategory. The model identifies

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 13: Representative learned memory for the `team_sports_match_winner` subcategory.

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

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 14: Representative learned memory for the `commodity_price_threshold_by_date` subcategory.

F 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.

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 15: 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 16: 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 17: 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 18: 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 19: 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 20: 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 21: 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 22: Prompt template for evaluating generated memory artifacts across five quality dimensions.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately summarize the paper's scope and contributions

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitation section is in Appendix F.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The detailed experimental settings are provided in Appendix B.1, and prompts are shown in Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We will release full code when the paper is accepted.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The detailed experimental settings are provided in Appendix B.1, and prompts are shown in Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The robustness of our method is shown in Appendix B.7.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports the model backbones, inference settings, memory-revision epochs, and token usage. Since the experiments use API-based LLM inference rather than local training, token counts and LLM call counts serve as the primary measure of computational cost.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We follow the NeurIPS Code of821 Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The social impact is discussed in Appendix F

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release high-risk data or models, such as pretrained language models, image generators, or scraped datasets. The experiments use existing open-source forecasting datasets and model APIs, and we do not redistribute proprietary model weights or newly collected high-risk data.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The forecasting datasets used in this paper are open-source and are cited in the experimental setup.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: LLMs are used as core components of the method. The paper reports the backbone models and experimental setup, and provides the prompts for the LLM-based taxonomy and memory-revision modules in the appendix.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.