Live-Evo: Online Evolution of Agentic Memory from Continuous Feedback

Anonymous ACL submission

Abstract

Large language model (LLM) agents are increasingly equipped with memory, which are stored experience and reusable guidance that can improve task-solving performance. Recent *self-evolving* systems update memory based on interaction outcomes, but most existing evolution pipelines are developed for static train/test splits and only approximate online learning by folding static benchmarks, making them brittle under true distribution shift and continuous feedback. We introduce LIVE-EVO, an online self-evolving memory system that learns from a stream of incoming data over time. LIVE-EVO decouples *what happened* from *how to use it* via an Experience Bank and a Meta-Guideline Bank, compiling task-adaptive guidelines from retrieved experiences for each task. To manage memory online, LIVE-EVO maintains experience weights and updates them from feedback: experiences that consistently help are reinforced and retrieved more often, while misleading or stale experiences are down-weighted and gradually forgotten, analogous to reinforcement and decay in human memory. On the live *Prophet Arena* benchmark over a 10-week horizon, LIVE-EVO improves Brier score by 20.8% and increases market returns by 12.9%, while also transferring to deep-research benchmarks with consistent gains over strong baselines.

1 Introduction

Large Language Models (LLMs) have increasingly been adopted as the backbone of agent systems, enabling agents to interact with external environments through tool usage and to solve complex, multi-step tasks (Wu et al., 2023; Zhang et al., 2024; Wu et al., 2025, 2024). Recent work has proposed self-evolving agents (ang Gao et al., 2025; Qiu et al., 2025; Long et al., 2025), which allow agents to learn from a training set by constructing tools, knowledge, and task-solving strategies to better accomplish the tasks. Specifically, the knowledge and strategies learnt from past experiences are being recognized as `memory` of agents (Jiang et al., 2024; Zhang et al., 2025b). These memory systems are typically organized into multiple levels, ranging from raw observations and interaction logs (Park et al., 2023; Zhang et al., 2025b; Zhong et al., 2023) to higher-level summarized experiences and abstract guidelines (Zhang et al., 2025a; Xu et al., 2025). During training, the agent can dynamically add, update, or remove memory entries based on its interaction outcomes.(Chhikara et al., 2025a) At test time, the agent leverages the evolved memory to guide decision-making on unseen tasks. These agents equipped with memory learned consistently outperform agents without memory evolution.

At the same time, memory evolution is inherently an online problem. In realistic deployments, an agent's experience accrues sequentially, and its memory must be updated continually by adding new evidence, revising outdated entries, and consolidating recurring patterns, rather than being rebuilt from a static corpus. This perspective is closely related to classic online and continual learning in traditional machine learning (Hoi et al., 2021), though the mechanisms and objectives for agent memory can differ. This shift raises a fundamental question: **how can LLM agents evolve continuously as new data arrives?**

Live benchmarks like *Prophet Arena* (Yang et al., 2025) and *FutureX* (Zeng et al., 2025) exemplify this paradigm by reframing agent evaluation as a longitudinal future prediction problem. In these benchmarks, agents are required to forecast probabilities of upcoming events, and are evaluated using both calibration-based metrics (e.g., Brier scores) and decision-oriented outcomes such as real market returns. In contrast to static retrieval or reasoning tasks, future prediction needs the agent to evolving during test time and continue adapt the memory to totally new tasks. Figure 1 shows the difference between traditional self-evolving memory and live

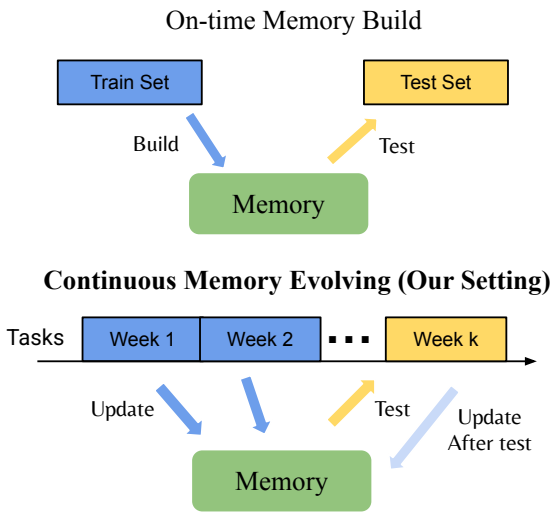


Figure 1: Traditional Self-Evolving Memory System build memory on training dataset and test with the evolved memory. While Live Self-Evolving Memory System build and learn to utilize Memory to tackle continuously new data.

self-evolving memory.

Only a few existing methods study memory for online task streams (Wei et al., 2025b; Wang et al., 2024b). However, they approximate "online" learning by splitting static benchmarks into folds, and therefore largely ignore distribution shift in truly streaming tasks. In contrast, live prediction benchmarks sample tasks from the real world, where environments and markets continually change over time. In this setting, success depends less on retrieving more information and more on *judiciously leveraging past experience over time*. Past experience can provide useful inductive bias, but it can also become stale or misleading as patterns drift or break. Therefore, a self-evolving memory system must go beyond storage: it should actively curate experiences and learn when and how to use them.

We introduce LIVE-EVO, a self-evolving agentic memory system designed for continuous task streams. LIVE-EVO learns not only *what happened before*, but also *how to use experience* by maintaining an *Experience Bank* and a *Meta-Guideline Bank*. For each incoming task, the agent executes a four-stage loop. **Retrieve**: it generates search queries to retrieve relevant question–experience pairs. **Compile**: it compiles retrieved experiences into a task-specific guideline, instructed by *Meta-Guidelines* that encode meta-heuristics for combining historical insights with the current task. **Act**: it performs *ContrastiveEval* by producing and comparing two independent predictions, one guided by the compiled guideline and one as a memory-free baseline, to quantify the contribution of the guideline. **Update**: it updates experience weights based on the observed performance gap; if the guideline underperforms, the agent generates a new entry for the Meta-Guideline Bank. Finally, to control memory growth, LIVE-EVO summarizes trajectories from poorly solved cases into candidate experiences and commits them to the Experience Bank only after re-evaluation confirms an improvement.

We evaluate LIVE-EVO on the *Prophet Arena* (Yang et al., 2025) benchmark over a 10-week horizon. Our results demonstrate that LIVE-EVO significantly outperforms static baselines, achieving a **20.8%** improvement in Brier Score and a **12.9%** increase in market returns. Furthermore, LIVE-EVO exhibits strong generalization on traditional deep-research benchmarks (Chen et al., 2025), outperforming specialized state-of-the-art methods. Our ablation studies further confirm that the components are essential for maintaining performance in online benchmarks.

2 Related Work

2.1 Self-Evolving Agentic Memory Systems

Memory transforms LLM agents into persistent, adaptive systems capable of long-horizon task-solving (Shan et al., 2025; Qian et al., 2024; Yan et al., 2024) and life-long learning (Zheng et al., 2025; Wang et al., 2024a). Existing solutions follow a hierarchical evolution. Early methods (Zheng et al., 2024), rely on static retrieval. While effective for repetitive tasks, they suffer from experience drift in changing environments (Hu et al., 2025). To address this, recent memory systems support high-level memory operations including forgetting (Liang et al., 2025; Zhong et al., 2024), building knowledge networks(Xu et al., 2025), and introducing heterogeneous memory structures (Chhikara et al., 2025b). Furthermore, some tasks focus on learn experience for task solving, BOT (Yang et al., 2024) synthesize high-level heuristics from past trajectories, EXPEL (Zhao et al., 2024) turns past trajectories into reusable experiences, and Agent Workflow Memory (Wang et al., 2024c) record the action sequence of successful task. However, none of them focus on evolving on live benchmarks.

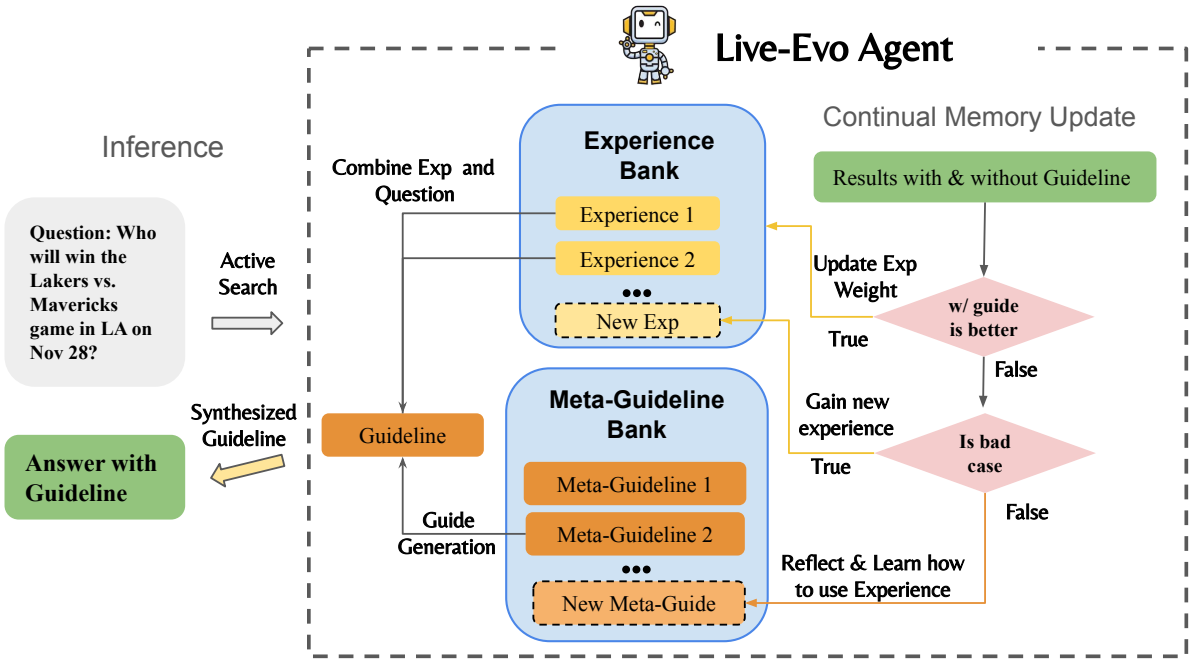


Figure 2: Structure of Live-Evo Agent. Given a question, the Live-Evo Agent will first search relevant experiences and generate a guideline based on the experiences, current task. Also, the generation will augmented by the meta-guideline bank, which teaches the agent how to combine experiences with current task. Inside the agent, the memory update mechanism continually updating experiences' weights and verifying new experiences and meta-guidelines.

2.2 Live Benchmarks

Traditional static benchmarks are released at a fixed time and evaluate models on a closed dataset (Wei et al., 2025a; Zhang et al., 2024). However, such benchmarks inevitably suffer from data leakage over time. To address this issue, several evaluation frameworks adopt a live setting by continuously introducing new tasks to assess models' general capabilities (Contributors, 2023; Xu et al., 2023). Other approaches rely on live human feedback for evaluation (Chiang et al., 2024). More recent live benchmarks release new tasks over time, providing a continuous evaluation stream for specific tasks. For example, LiveCodeBench (Jain et al., 2024) regularly releases new coding problems each quarter. Emerging *future prediction benchmarks*, such as *Prophet Arena* (Yang et al., 2025) and *FutureX* (Zeng et al., 2025), introduce real-world tasks on a weekly basis, offering an ideal testbed for self-evolving agents.

3 Method

We introduce LIVE-EVO, an agentic memory system explicitly designed for true live benchmarks, where tasks arrive sequentially and feedback is revealed over time (See Figure 2). In contrast to prior memory systems that primarily store experiences and abstract them into static summaries, LIVE-EVO continuously optimizes how past experience is used over time, updating its memory policies as new data arrives and grounding each update in ongoing environmental feedback.

LIVE-EVO composes of two memory banks: an *Experience Bank* $\mathcal{E}$ and a *Meta-Guideline Bank* $\mathcal{M}$. The Experience Bank stores past task interactions in a structured, reusable form. When queried, the agent does not simply append retrieved trajectories into the prompt; instead, it applies a learned *procedure* that distills retrieved experiences into task-relevant signals and actionable guidance, making the memory system's inductive bias explicit. Complementarily, the Meta-Guideline Bank stores higher-level *composition instructions*, which meta-guidelines that specify how to transform retrieved experiences into a task-adaptive guideline under different conditions. Together, these two banks separate *what happened before* (experience) from *how to use it* (guideline), enabling memory usage to improve over time as new tasks arrive.

We formalize the self-evolving agentic memory system as a closed-loop decision process over the

Algorithm 1: LIVE-EVO

Input: Task stream batch $\mathcal{Q}$; Experience bank $\mathcal{E}$ with weights $\{w_e\}$; Meta-guideline bank $\mathcal{M}$; Bad-case fraction ρ
Output: Updated $\mathcal{E}, \mathcal{M}$

foreach $q \in \mathcal{Q}$ **do**
  $E_q,\ \hat{m} \leftarrow \textsc{Retrieve}(q, \mathcal{E}, \mathcal{M})$ // **Retrieve:** top-k experiences + selected meta-guideline
  $g_q \leftarrow \textsc{CompileGuideline}(q, E_q, \hat{m})$ // **Compile:** LLM produces task-specific memory guideline
  $r_q^{\text{on}},\ r_q^{\text{off}},\ \tau_q \leftarrow \textsc{ContrastiveEval}(q, g_q)$ // **Act:** scores w/ and w/o memory; keep memory-on trajectory τ_q
  // **Update**
  $W_{E_q} \leftarrow \textsc{UpdateWeights}(W_{E_q}, r_q^{\text{on}} - r_q^{\text{off}})$ // update weights of selected experiences
  if $r_q^{on} - r_q^{off} \leq 0$ **then**
    $\mathcal{M} \leftarrow \mathcal{M} \cup \{\textsc{Reflect}(q, g_q, E_q)\}$ // add new meta-guideline on failure
  end
end
// **Update**
$\mathcal{Q}_{\text{bad}} \leftarrow \textsc{SelectWorst}(\mathcal{Q}, \{r_q^{\text{on}}\}, \rho)$ // worst ρ fraction of tasks solved with memory
foreach $q \in \mathcal{Q}_{bad}$ **do**
  $e_q^{\text{new}} \leftarrow \textsc{Summarize}(q, \tau_q)$ // summarize new experience from stored memory-on trajectory
  if $\textsc{Eval}(q, e_q^{new}) > r_q^{on}$ **then**
    $\mathcal{E} \leftarrow \mathcal{E} \cup \{e_q^{\text{new}}\}$ // re-evaluate with new experience and commit if it improves
  end
end
return $\mathcal{E}, \mathcal{M}$

memory banks. For each new task batch, the agent operates through four stages:

$$\{\textsc{Retrieve}, \textsc{Compile}, \textsc{Act}, \textsc{Update}\}.$$

Given a task, the agent first actively search its own memory to retrieve relevant experiences and the meta guideline. Then the agent compiles a guideline based on the meta instruction, the retrieved experiences and the task. The agent then solves the task with the compiled guideline. Finally, the trajectory and result of solving this task will be used to update the memory, including the experience bank and the meta guideline bank. Next we explain each stage of LIVE-EVO in detail (also see Algorithm 1).

3.1 Retrieve

Given a task q, the agent A first retrieves potentially relevant experiences and also a meta guidelines:

$$E_q,\ \hat{m} \leftarrow \textsc{Retrieve}(q, \mathcal{E}, \mathcal{M})$$

We note that the task will not be used directly to query the bank. Instead, the agent generates queries from the given task for both question matching and experience-content matching. While existing systems retrieve through similarity matching (Xu et al., 2025; Park et al., 2023)) or active exploration strategies (e.g., in which the agent probes the memory bank iteratively) (Chhikara et al., 2025a; Long et al., 2025), our **active retrieval** design enables the agent to retrieve relevant information from multiple dimensions, which contrasts with traditional search actions by allowing the agent to seek structural analogies or reasoning patterns rather than simple semantic overlaps, granting the agent higher autonomy in defining what constitutes "relevant" information for a complex forecasting query.

3.2 Compile

The agent transforms retrieved experiences into task-adaptive guidance:

$$g = \textsc{CompileGuideline}(q, E_q, \hat{m}).$$

COMPILEGUIDELINE operationalizes the role of the Guideline Bank: it selects and applies a meta-guideline $\hat{m}$ to turn the retrieved experience set E_q into an executable, task-specific guideline g for the current task q. Concretely, given E_q, LIVE-EVO performs meta-cognitive compilation by (i) extracting cross-experience regularities, (ii) grounding them in the current task context, and (iii) instantiating a guideline g conditioned on $\hat{m}$ to steer downstream decision making.

In contrast, prior approaches typically either concatenate retrieved logs as additional context or rely on fixed abstraction operators (e.g., summaries or heuristic rules) that remain static and do not improve from online feedback (Wang et al., 2024b; Xu et al., 2025).

3.3 Act

Conditioned on the task and the derived guideline, the agent executes a policy:

$$r_q, \tau_q = \textsc{Act}(q \mid g),$$

where τ_q denotes the trajectory, and r_q denotes the resulting outcome signal. The structure of r depends on the evaluation regime. In traditional reasoning or search benchmarks, r is often binary, reflecting task success or failure. In contrast, online benchmarks (e.g. Yang et al. (2025)) yield continuous feedback. These dense signals provide a richer learning substrate for memory evolution than sparse correctness-based rewards.

For every task, we additionally conduct a *contrastive evaluation* to measure the causal impact of retrieved experience at action time. Concretely, we execute the agent again without the compiled guideline. We then compare the resulting outcomes to quantify whether memory usage provides a net benefit on that task. This comparison will later be used to update memory.

3.4 Update

Finally, the agent incorporates new experience into the memory bank. The update mechanism governs how experience accumulates over time and is grounded in objective environmental feedback. Concretely, from *Contrastive Evaluation* we obtain the empirical gain of using the compiled guideline relative to the memory-free baseline. This gain is used to adjust the retrieval weights of the selected experiences: when the guideline improves performance, the corresponding experience weights are increased; when it harms performance, the weights are decreased. This reinforcement-and-decay dynamic is analogous to human memory, where useful experiences are strengthened through repeated success while misleading or outdated ones are gradually suppressed. In addition, failures trigger a reflection step that produces a new meta-guideline, which is added to the meta-guideline bank to improve future guideline compilation.

After processing a batch, we further perform selective experience acquisition rather than indiscriminately storing every trajectory. We identify the worst-performing fraction of tasks under the memory-on setting and generate a candidate experience by summarizing and reflecting on the stored trajectory. We then re-evaluate the task with this candidate experience, and commit it to the experience bank only if it yields a statistically significant improvement over the original memory-on score. This selective write-back controls memory growth while ensuring that new entries are justified by measurable gains.

4 Experiment

4.1 Setup

We evaluate LIVE-EVO on **Prophet Arena** (Yang et al., 2025), a future-prediction benchmark spanning the latest 10 weeks with 500 tasks in total. Each task contains a question, a candidate list, and a bid-price snapshot taken 6 hours before close, which we use to compute returns relative to market consensus. We enforce strict time-based retrieval to prevent information leakage past the close time. We also evaluate on **Xbench-DeepResearch** (Chen et al., 2025) to assess generalization beyond future prediction. We split the benchmark into 10 folds, learn experience sequentially across folds, and report the overall average accuracy.

We use GPT-4.1-mini as the backbone model for most experiments if not specified. All experiments use a `temperature`=0.2, with `bad_case_percentile`=0.3, `min_brier_improvement`=0.05, and `experience_similarity_threshold`=0.5.

Metrics. For XBench-DeepResearch, we use *accuracy* as metrics. For Prophet Arena, we use *Brier Score* and *Market Return* as metrics. Given a query q and a set of candidate outcomes $\mathcal{C} = \{c_1, \ldots, c_K\}$, the agent outputs probabilities of each candidates $\hat{\mathbf{p}} = (\hat{p}_1, \ldots, \hat{p}_K)$ over the outcomes. Let $\mathbf{y} \in \{0,1\}^K$ denote the realized outcome and $\mathbf{m} = (m_1, \ldots, m_K)$ the corresponding prediction market prices. We report the multiclass Brier score as a calibration metric. A lower Brier score indicates that the predicted probabilities are closer to the true real-world outcomes.

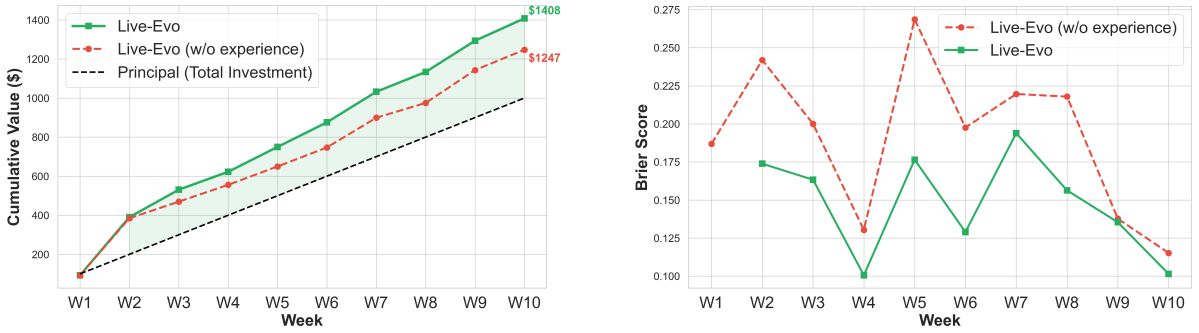
$$\text{BS} = \sum_{k=1}^{K} (\hat{p}_k - y_k)^2, \tag{1}$$

We also compute the market return to quantify the advantage of LIVE-EVO over market-based baselines. The return is obtained by taking a unit long position on outcome c_k whenever the predicted probability $\hat{p}_k$ exceeds the market-implied probability m_k:

$$R = \sum_{k=1}^{K} \mathbb{I}[\hat{p}_k > m_k]\,(y_k - m_k). \tag{2}$$

Model	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	Avg
Base Model											
GPT-4.1-mini	0.18	0.20	0.31	0.18	0.24	0.26	0.23	0.25	0.23	0.15	0.22
Deep Research Methods											
MiroFlow	0.28	0.08	0.53	0.44	0.40	0.43	0.27	0.33	0.26	0.22	0.32
Qwen Deep Research	0.17	0.22	0.23	0.15	0.22	0.22	0.21	0.19	0.20	0.13	0.20
Live-Evo (w/o Experience)	0.18	0.24	0.20	0.13	0.27	0.19	0.22	0.22	0.13	0.11	0.19
Self-Evolving Memory Methods											
ReMem	0.19	0.23	**0.14**	0.11	0.21	0.18	0.19	0.17	0.15	0.11	0.16
Live-Evo **(ours)**	0.19	0.17	0.16	**0.10**	**0.17**	**0.12**	0.19	**0.15**	**0.13**	**0.10**	**0.14**

Table 1: Brier Score (Lower the better) on Prophet-Arena - Weekly Performance Comparison



(a) Cumulative Portfolio Value (Invest $100 Per Week)

(b) Brier Score Comparison

Figure 3: Performance Analysis Comparison. (a) shows the cumulative portfolio value, and (b) shows the Brier score comparison.

Baselines. We compare LIVE-EVO with the following methods as baselines. **(1) Base Models** . We retrieve the top-10 web search results for each query and provide the model with a summarized version of these websites generated by the model itself. The model is then required to output the probability distribution based on this static information. **(2) Deep Research Methods.** We evaluate two representative open-source frameworks, `Qwen Deep Research` (Team et al., 2025) and `MiroFlow` (Team, 2025) which support multiple tools and complex multi-agent workflows. We also evaluate LIVE-EVO without experience, representing the base search agent without evolution. **(3) Self-Evolving Memory Systems.** *ReMem* (Wei et al., 2025b): a self-evolving agent baseline that constructs summarized experiences from raw trajectories and retrieves relevant memories at test time.

4.2 Main Results

Table 1 reports the Brier scores over the most recent 10 weeks of the Prophet Arena benchmark. All methods use GPT-4.1-mini as the foundation model. We compare our method against a Base Model and several open-source Deep Research frameworks. We do not include closed-source Deep Research systems as baselines, because their search tools do not support strict time-based filtering.

Result Analysis The results demonstrate that our agent achieves state-of-the-art performance in terms of the average Brier score, and outperforms all baselines in the majority of individual weeks.

Open-source Deep Research methods perform relatively poorly on this benchmark. This is expected, as they are optimized for discovering partial clues or supporting evidence, rather than producing calibrated probabilistic forecasts of future events. In practice, these methods are often misled by incomplete or temporally fragile signals.

The ReMem baseline shows a consistent improvement over the static Base Model (GPT-4.1-mini), indicating that incorporating self-evolving memory is beneficial for future prediction. However, its performance remains weaker than Live-Evo, highlighting the importance of actively managing and adapting experiences. These results confirm that our design more effectively leverages past experience under continuously evolving, real-world

Table 2: Generalization of LIVE-EVO across different foundation models. We report Brier score (lower is better) and cumulative market return (higher is better), along with relative improvements over the corresponding base agents.

Base Model	Method	Brier Score (↓)		Market Return (↑)	
		Value	Imp.	Value	Imp.
GPT-4.1-mini	Base (w/o Mem)	0.19	–	1.24	–
	Live-Evo	**0.14**	*20.8%*	**1.46**	*12.9%*
GPT-4.1	Base (w/o Mem)	0.18	–	1.13	–
	Live-Evo	**0.17**	*3.0%*	1.18	*4.4%*
GPT-5-mini	Base (w/o Mem)	0.16	–	1.34	–
	Live-Evo	**0.15**	*4.5%*	**1.36**	*1.6%*
Qwen3-8B	Base (w/o Mem)	0.19	–	1.20	–
	Live-Evo	**0.18**	*3.5%*	**1.21**	*0.5%*

conditions.

Performance Comparison We compare LIVE-EVO with its underlying base search agent which isolates the contribution of the proposed experience management system.

Figure 3a illustrates the cumulative market returns under a simplified investment strategy. Assuming an investment of $100 per week, LIVE-EVO achieves a $150 higher return over the 10-week period. Notably, the performance gap between the two agents widens over time, indicating that LIVE-EVO continuously improves its decision quality as more experience is accumulated. Figure 3b further reports the weekly Brier scores. LIVE-EVO consistently outperforms the base agent across all weeks. The improvement is particularly pronounced during periods where the base agent exhibits poor calibration, such as Weeks 5 and 6. These results suggest that LIVE-EVO can stabilize predictions under difficult or volatile conditions.

Table 3: Acc. on Xbench-DeepResearch. All methods are tested with GPT-4.1-mini.

Methods	Acc
Qwen-DeepResearch	0.43
MiroFlow	0.45
ReMem	0.40
Live-Evo (ours)	**0.46**

4.3 Additional Results with Different Models

To evaluate the robustness of LIVE-EVO across foundation models of varying capacity and provenance, we conduct experiments on Prophet Arena with GPT-4.1-mini, GPT-4.1, GPT-5-mini, and Qwen3-8B, covering both closed-source and open-source models (See Table 2). For each model, we compare LIVE-EVO against its corresponding base agent without experience.

Across all evaluated models, LIVE-EVO consistently improves both Brier score and market return. These results demonstrate that the proposed experience management mechanism is broadly compatible with heterogeneous backbone models and does not rely on model-specific heuristics.

Notably, the largest relative improvement is observed with GPT-4.1-mini. This behavior is expected for two reasons. First, weaker base models exhibit greater headroom for improvement. Second, they generate more frequent failure cases during early weeks, which in turn provide richer supervisory signals for experience correction and guideline refinement. In contrast, stronger models such as GPT-5-mini already produce well-calibrated predictions, leaving less room for further gains.

4.4 Results on Deep Research Benchmark

Although LIVE-EVO is not specifically designed for traditional deep research tasks, it nevertheless demonstrates competitive and consistent advantages over both deep research frameworks and prior self-evolving memory methods.

As shown in Table 3, LIVE-EVO achieves the highest accuracy among all evaluated methods. Compared to specialized deep research systems such as Qwen-DeepResearch and MiroFlow, LiveEvo attains superior performance despite lacking task-specific heuristics for evidence exploration. This suggests that experience management learned under live and non-stationary conditions generalizes beyond future prediction, benefiting broader reasoning tasks.

Table 4: Ablation Study Relative to the Full-Memory Model. Color intensity indicates the magnitude of relative change compared to the full-memory setting.

Setting	Avg. Brier	Change (%) ↓	Avg. Return	Change (%) ↑
Live-Evo	**0.14**	–	**1.46**	–
w/o weight-update	0.17	+17.01%	1.34	-8.01%
w/o meta-guideline	0.16	+10.88%	1.41	-3.42%
w/o guideline-compile	0.16	+11.56%	1.16	-20.40%
w/o active-retrieve	0.17	+14.97%	1.22	-16.77%

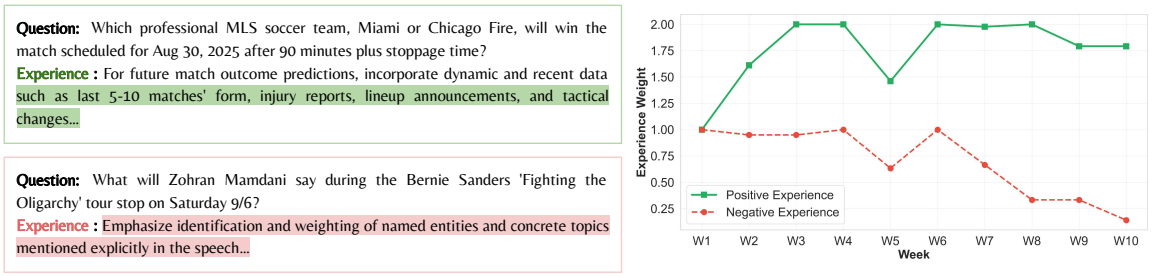


Figure 4: Case Study. The figure contrasts a high-weight experience (green), which provides reusable methods, with a low-weight experience (red), which contains hallucinations, and shows how their weights evolve weekly.

4.5 Ablation Study

We conduct ablation studies to assess the contributions of key components in LIVE-EVO (Table 4). Removing any single module consistently degrades performance in both Brier Score and market return, indicating that each component is non-trivial: w/o weight-update fixes all experience weights, w/o meta-guideline removes meta-guidance bank for guideline generation, w/o guideline-synthesis directly uses retrieved experiences, and w/o active-retrieve queries memory using only the question. Among these variants, disabling guideline synthesis causes the largest drop in market return, underscoring the importance of converting accumulated experience into actionable guidance, while removing active retrieval or adaptive weight updates also leads to substantial degradation. Overall, the results show that Live-Evo's improvements stem from the **synergistic interaction** of its components rather than any single design choice.

4.6 Case Study

To illustrate how the weight-update mechanism operates in practice, we analyze experiences with the **lowest** and **highest** learned weights. In Figure 4, the red box highlights a low-weight experience that contains a clear hallucination: the experience suggests retrieving the content of a speech, whereas the task requires predicting the outcome of the speech. Such mismatches lead to consistently poor downstream performance and are therefore downweighted over time.

In contrast, the green box corresponds to a high-weight experience that provides a reusable and task-aligned guideline, recommending the analysis of recent match forms. This experience consistently supports accurate predictions and is thus reinforced by the weight-update mechanism.

This case study demonstrates that LIVE-EVO can progressively filter out low-quality or misleading experiences by reducing their weights, while amplifying high-quality, transferable experience. As a result, the agent learns to rely on increasingly reliable guidance, leading to improved future prediction performance.

5 Conclusion

We introduced Live-Evo, the first online-evolving agentic memory system specifically designed for benchmarks with continuous, real-world feedback. By employing a four-stage evolutionary loop:Retrieve, Compile, Act, and Update, the system dynamically learns to optimize how past experiences are transformed into task-adaptive guidance. Our evaluation on the Prophet Arena benchmark demonstrates that LIVE-EVO achieves significant improvement over strong baselines. Furthermore, the system exhibits robust generalization on deep-research benchmarks. These results underscore the vital role of feedback-driven experience management in building persistent, adaptive agentic systems for non-stationary environments.

6 Limitations

While LIVE-EVO achieves strong performance, its design introduces several potential constraints. First, its reliance on the dense environment feedback ensures robust calibration but may limit applicability in settings with sparse or subjective feedback. Second, the **Verify Before Update** protocol strictly admits new experiences only with statistically significant gains, which can delay the adoption of subtle or emerging heuristics.

References

Huan ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Yiran Wu, Hongru Wang, Han Xiao, Yuhang Zhou, Shaokun Zhang, Jiayi Zhang, Jinyu Xiang, Yixiong Fang, Qiwen Zhao, Dongrui Liu, and 8 others. 2025. A survey of self-evolving agents: On path to artificial super intelligence. *Preprint*, arXiv:2507.21046.

Kaiyuan Chen, Yixin Ren, Yang Liu, Xiaobo Hu, Haotong Tian, Tianbao Xie, Fangfu Liu, Haoye Zhang, Hongzhang Liu, Yuan Gong, Chen Sun, Han Hou, Hui Yang, James Pan, Jianan Lou, Jiayi Mao, Jizheng Liu, Jinpeng Li, Kangyi Liu, and 14 others. 2025. xbench: Tracking agents productivity scaling with profession-aligned real-world evaluations. *Preprint*, arXiv:2506.13651.

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

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

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. 2024. Chatbot arena: An open platform for evaluating llms by human preference. *Preprint*, arXiv:2403.04132.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Steven CH Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. 2021. Online learning: A comprehensive survey. *Neurocomputing*, 459:249–289.

Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, Senjie Jin, Jiejun Tan, Yanbin Yin, Jiongnan Liu, Zeyu Zhang, Zhongxiang Sun, Yutao Zhu, Hao Sun, Boci Peng, and 28 others. 2025. Memory in the age of ai agents. *Preprint*, arXiv:2512.13564.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *Preprint*, arXiv:2403.07974.

Xun Jiang, Feng Li, Han Zhao, Jiahao Qiu, Jiaying Wang, Jun Shao, Shihao Xu, Shu Zhang, Weiling Chen, Xavier Tang, and 1 others. 2024. Long term memory: The foundation of ai self-evolution. *arXiv preprint arXiv:2410.15665*.

Xuechen Liang, Yangfan He, Yinghui Xia, Xinyuan Song, Jianhui Wang, Meiling Tao, Li Sun, Xinhang Yuan, Jiayi Su, Keqin Li, Jiaqi Chen, Jinsong Yang, Siyuan Chen, and Tianyu Shi. 2025. Self-evolving agents with reflective and memory-augmented abilities. *Preprint*, arXiv:2409.00872.

Lin Long, Yichen He, Wentao Ye, Yiyuan Pan, Yuan Lin, Hang Li, Junbo Zhao, and Wei Li. 2025. Seeing, listening, remembering, and reasoning: A multimodal agent with long-term memory. *Preprint*, arXiv:2508.09736.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. *Preprint*, arXiv:2304.03442.

Chen Qian, Yufan Dang, Jiahao Li, Wei Liu, Zihao Xie, YiFei Wang, Weize Chen, Cheng Yang, Xin Cong, Xiaoyin Che, and 1 others. 2024. Experiential co-learning of software-developing agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5628–5640.

Jiahao Qiu, Xuan Qi, Tongcheng Zhang, Xinzhe Juan, Jiacheng Guo, Yifu Lu, Yimin Wang, Zixin Yao, Qihan Ren, Xun Jiang, Xing Zhou, Dongrui Liu, Ling Yang, Yue Wu, Kaixuan Huang, Shilong Liu, Hongru Wang, and Mengdi Wang. 2025. Alita: Generalist agent enabling scalable agentic reasoning with minimal predefinition and maximal self-evolution. *Preprint*, arXiv:2505.20286.

Lianlei Shan, Shixian Luo, Zezhou Zhu, Yu Yuan, and Yong Wu. 2025. Cognitive memory in large language models. *arXiv preprint arXiv:2504.02441*.

MiroMind AI Team. 2025. Miroflow: A high-performance open-source research agent framework. https://github.com/MiroMindAI/MiroFlow.

Tongyi DeepResearch Team, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, and 1 others. 2025. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2024a. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. 2024b. Agent workflow memory. *arXiv preprint arXiv:2409.07429*.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. 2024c. Agent workflow memory. *Preprint*, arXiv:2409.07429.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. 2025a. Browsecomp: A simple yet challenging benchmark for browsing agents. *Preprint*, arXiv:2504.12516.

Tianxin Wei, Noveen Sachdeva, Benjamin Coleman, Zhankui He, Yuanchen Bei, Xuying Ning, Mengting Ai, Yunzhe Li, Jingrui He, Ed H. Chi, Chi Wang, Shuo Chen, Fernando Pereira, Wang-Cheng Kang, and Derek Zhiyuan Cheng. 2025b. Evo-memory: Benchmarking llm agent test-time learning with self-evolving memory. *Preprint*, arXiv:2511.20857.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation. *Preprint*, arXiv:2308.08155.

Yiran Wu, Mauricio Velazco, Andrew Zhao, Manuel Raúl Meléndez Luján, Srisuma Movva, Yogesh K Roy, Quang Nguyen, Roberto Rodriguez, Qingyun Wu, Michael Albada, and 1 others. 2025. Excytin-bench: Evaluating llm agents on cyber threat investigation. *arXiv preprint arXiv:2507.14201*.

Yiran Wu, Tianwei Yue, Shaokun Zhang, Chi Wang, and Qingyun Wu. 2024. Stateflow: Enhancing llm task-solving through state-driven workflows. *arXiv preprint arXiv:2403.11322*.

Liang Xu, Anqi Li, Lei Zhu, Hang Xue, Changtai Zhu, Kangkang Zhao, Haonan He, Xuanwei Zhang, Qiyue Kang, and Zhenzhong Lan. 2023. Superclue: A comprehensive chinese large language model benchmark. *Preprint*, arXiv:2307.15020.

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

Yikuan Yan, Yaolun Zhang, and Keman Huang. 2024. Depending on yourself when you should: Mentoring llm with rl agents to become the master in cybersecurity games. *Preprint*, arXiv:2403.17674.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E Gonzalez, and Bin Cui. 2024. Buffer of thoughts: Thought-augmented reasoning with large language models. *Advances in Neural Information Processing Systems*, 37:113519–113544.

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

Zhiyuan Zeng, Jiashuo Liu, Siyuan Chen, Tianci He, Yali Liao, Yixiao Tian, Jinpeng Wang, Zaiyuan Wang, Yang Yang, Lingyue Yin, Mingren Yin, Zhenwei Zhu, Tianle Cai, Zehui Chen, Jiecao Chen, Yantao Du, Xiang Gao, Jiacheng Guo, Liang Hu, and 12 others. 2025. Futurex: An advanced live benchmark for llm agents in future prediction. *Preprint*, arXiv:2508.11987.

Guibin Zhang, Haotian Ren, Chong Zhan, Zhenhong Zhou, Junhao Wang, He Zhu, Wangchunshu Zhou, and Shuicheng Yan. 2025a. Memevolve: Meta-evolution of agent memory systems. *Preprint*, arXiv:2512.18746.

Yaolun Zhang, Yinxu Pan, Yudong Wang, and Jie Cai. 2024. Pybench: Evaluating llm agent on various real-world coding tasks. *Preprint*, arXiv:2407.16732.

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

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024. Expel: Llm agents are experiential learners. *Preprint*, arXiv:2308.10144.

Junhao Zheng, Chengming Shi, Xidi Cai, Qiuke Li, Duzhen Zhang, Chenxing Li, Dong Yu, and Qianli Ma. 2025. Lifelong learning of large language model based agents: A roadmap. *Preprint*, arXiv:2501.07278.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. 2024. Synapse: Trajectory-as-exemplar prompting with memory for computer control. *Preprint*, arXiv:2306.07863.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2023. Memorybank: Enhancing large language models with long-term memory. *Preprint*, arXiv:2305.10250.

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

A Appendix

A.1 Implementation Details

Base Search Agent. To isolate the efficacy of the Live-Evo, We build a simple search agent and equipped the agent with basic google-search and web fetch tool. We use serper api as the google-search api and apply the time filter by editing the queries. The max turns of one task is set to 20. For web content that exceed the agent's max sequence length, we call the llm to summarize the content.

Retrieve. We calculate the semantic similarity based on all-MiniLM-L6-v2 model from the sentence-transformers library. The system enforces a minimum weighted similarity threshold of $\tau = 0.3$. Only experiences has higher relativity will be retrieved.

A.2 Example Case

In this section, we present a comprehensive execution trajectory of the LIVE-EVO system on a specific future prediction task from the *Prophet Arena* benchmark. This case study illustrates how the agent retrieves historical failures, synthesizes a dynamic guideline, executes actions based on that guideline, and achieves a superior Brier score compared to the baseline.

A.2.1 Task Definition

The agent is presented with a binary prediction task regarding an NFL game.

Task Input

Question: Which professional football team, Cincinnati or Pittsburgh, will win the game scheduled for Oct 16, 2025?
Ground Truth: Cincinnati (1), Pittsburgh (0)

A.2.2 Phase 1: Retrieve

Upon receiving the task, the agent queries the Experience Bank ($\mathcal{E}$). The system retrieves relevant past experiences where the agent previously failed due to over-reliance on betting odds or missed schedule changes. Two representative retrieved experiences are shown below:

Selected Retrieved Experiences

Experience ID: 46f126ef (Sports/NCAAF)
Failure Reason: The agent over-relied on pre-game betting odds and recent season trends without accounting for roster changes or "home advantage" dynamics closer to the game date.
Improvement: Incorporate dynamic, up-to-date info (roster, coaching) as the event approaches. Avoid static betting odds.
Experience ID: 50fe0d0c (Sports/MLS)
Failure Reason: The agent failed to update its prediction to reflect the **rescheduling** of the match, basing probabilities on outdated timing.
Improvement: Always verify the event date and confirm the prediction is relative to the current schedule.

A.2.3 Phase 2: Compile

Using the retrieved experiences, the agent synthesizes a task-specific guideline ($\mathcal{G}$) via the REUSE operator. This guideline explicitly warns against the specific pitfalls identified in the retrieved memory (e.g., verifying dates and ignoring early odds).

Synthesized Guideline

- **Dynamic Information:** Prioritize authoritative sources (e.g., official injury reports, press releases) close to the game date over early betting odds or historical reputation.
- **Schedule Verification:** Implement a workflow step to confirm the exact game date and update contextual data to avoid outdated inputs (derived from Exp 50fe0d0c).
- **Scenario Analysis:** Explicitly model the impact of key player absences (e.g., injury reports) and home vs. away advantages.

A.2.4 Phase 3: Act

Guided by the synthesized instructions, the agent executes a search strategy. Unlike a standard agent that might immediately look up win probabilities, the LIVE-EVO agent follows the guideline to first verify the schedule and check specific injury reports.

Agent Execution Trajectory (Abbreviated)

Action 1: Web Search
`Query: "Cincinnati vs Pittsburgh NFL game Oct 16 2025 injury report roster updates"`
Observation 1 (Search Results): Results indicate the game date matches the query. Snippets show specific injury reports: "Bengals TE Mike Gesicki removed from injury report," "Bengals DE Trey Hendrickson ruled OUT."

Action 2: Visit Page
Observation 2 (Web Content): *[Content summarized for brevity]* Detailed tables confirm:

- **Cincinnati:** Trey Hendrickson (DE) is OUT; Tanner Hudson (TE) is OUT.
- **Pittsburgh:** Calvin Austin III (WR) is OUT; Miles Killebrew (S) is OUT.

Date confirmation: Thursday, Oct 16, 2025.
Action 3: Web Search
`Query: "Cincinnati Bengals vs Pittsburgh Steelers recent performance 2025 season stats"`
Observation 3 (Search Results): Results confirm the game conclusion: "Bengals 33-31 Steelers (Oct 16, 2025) Final Score."

A.2.5 Phase 4: Result & Update (Evaluation)

The agent synthesizes the gathered evidence. **A detailed comparison between the baseline and the LIVE-EVO agent is presented in Table 5.** While the Baseline agent (without memory) relied on Pittsburgh's superior record (4-1) and betting odds, the LIVE-EVO agent incorporated the specific game-day dynamics and injury resilience found during the guided search.

The LIVE-EVO system achieved a **Brier Score improvement of 0.2829**. Following this success, the weight of the retrieved experiences is increased, reinforcing the guideline to "verify schedule" and "ignore early odds" for future sports prediction tasks.

A.3 Prompts

Prompt: Retrieve Query Generation shows the prompt that guide the agent to generate retrieve queries for the experience bank and meta-guideline bank. Prompt: Guideline Compile shows the prompt that guide the agent generate guideline based on experiences, meta-guideline and current tasks. Prompt: Base Agent Prediction shows how the base search agent will act given the task and the guideline.

Prompt: Search Query Generation

You are exploring an experience database to find relevant past predictions that can help with a new task.

Current Task: Which professional football team, Cincinnati or Pittsburgh, will win the game scheduled for Oct 16, 2025?

The experience database contains past prediction experiences with these fields:

- `question`: The prediction question/task title
- `improvement`: Key insights on how to improve similar predictions
- `failure_reason`: What went wrong in past predictions
- `missed_information`: Information sources that were missed
- `category`: Domain category (politics, technology, etc.)

Generate 2-3 search queries. For each, specify the text and search type: "question" or "experience".

Output as JSON:

```
{
  "queries": [
   {"query": "...", "search_target": "..."}
  ]
}
```

Table 5: Performance Comparison: Baseline vs. Live-Evo

Baseline Agent (No Memory)	Live-Evo Agent (With Memory)
Prediction: Pittsburgh (0.65), Cincinnati (0.35) **Reasoning:** Relied on Pittsburgh's 4-1 record and winning streak. Heavily weighted betting odds favoring Steelers. **Brier Score:** 0.5329	**Prediction:** Cincinnati (0.55), Pittsburgh (0.45) **Reasoning:** Identified close 33-31 victory conditions. Weighed resilience despite injuries and home advantage more heavily than static odds. **Brier Score:** 0.2500

Prompt: Guideline Compile

You are synthesizing insights from past prediction experiences to create a guideline for a new prediction task.
Current Task: Which professional football team, Cincinnati or Pittsburgh, will win the game scheduled for Oct 16, 2025?
Relevant Past Experiences Found: [Experience 1 Summary: Over-relied on betting odds, missed key defensive injuries...] [Experience 2 Summary: Over-relied on historical trends, missed roster changes...]
CRITICAL: Experience Applicability Check Before applying lessons from past experiences, you MUST assess whether each experience is truly applicable to this specific task type. Identify which lessons are directly applicable vs. need adaptation. Based on these experiences AND the applicability check, generate a FOCUSED and ACTIONABLE guideline (3-5 bullet points) for this prediction task.
Output ONLY the bullet points.

Prompt: Base Agent Prediction

You are tasked with predicting the probability of different outcomes for the following event:

Event: Which professional football team, Cincinnati or Pittsburgh, will win the game scheduled for Oct 16, 2025?
Possible outcomes: "Cincinnati", "Pittsburgh"

Task-Specific Guideline

- Incorporate dynamic, up-to-date information close to the game date, including granular injury reports (e.g., Trey Hendrickson status).
- Avoid over-reliance on early betting odds or historical reputation.
- Explicitly model and quantify the impact of key player absences and home-field advantage.

CRITICAL: How to Properly Use This Guideline

1. **Verify Applicability**: Assess if the current task matches the context of the lesson.
2. **Trust Your Current Research**: If fresh findings contradict the guideline, prioritize fresh evidence.

Your task:

1. Research this event by searching for relevant information online.
2. Analyze the information to assess the likelihood of each outcome.
3. Provide a probability estimate (between 0 and 1).

Output as JSON.