RSI-FORECASTER: A SELF-IMPROVING AGENT FOR EVENT FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Accurate event forecasting across real-world domains requires task-specific strategies whose development has long relied on substantial human expertise and effort. We formulate autonomous forecasting strategy development as a recursive self-improvement (RSI) process, where an LLM iteratively executes, evaluates, and revises its forecasting strategy. Yet forecasting creates a fundamental feedback deadlock: the target outcome remains unknown when a prediction is made. We empirically show that resolving this deadlock is non-trivial: naive historical tasks and imperfect backtesting provide little target improvement, and existing RSI methods improve historical performance in backtesting but struggle to reliably transfer gains to the unresolved targets. We introduce RSI-FORECASTER, an end-to-end RSI system that constructs target-relevant historical proxy tasks, replays them in a leakage-controlled open-web environment, and uses predictive and process feedback to hierarchically improve strategies, followed by held-out selection and transfer the resulting gains to the unresolved target. Across three months of evaluation on two popular living forecasting benchmarks, RSI-FORECASTER maintains leading performance and ranks second among 400 submissions on ForecastBench's preliminary cross-round leaderboard as of the paper submission deadline. Moreover, it substantially outperforms forecasts from the general public and prediction-market crowds, while performing on par with human superforecasters. RSI-FORECASTER consistently improves Brier Index across five LLM backbones, up to 15.54 points.

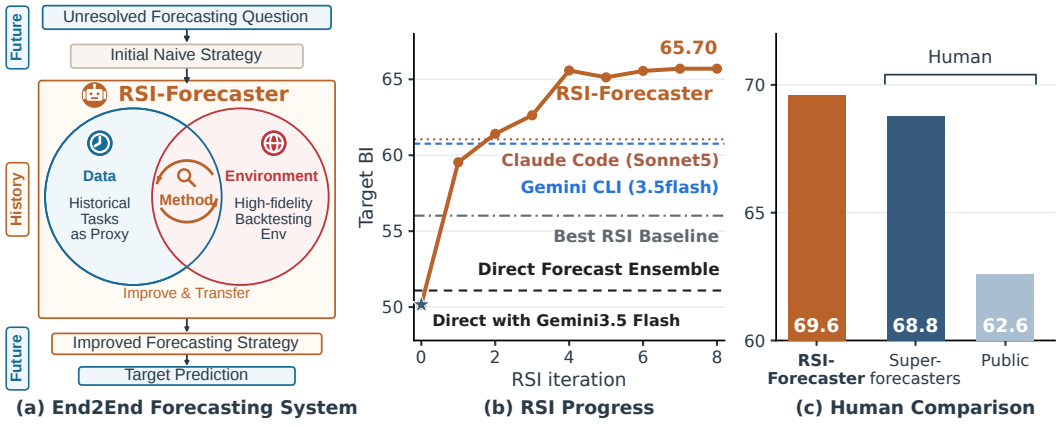


Figure 1: RSI-FORECASTER method and results. (a) RSI-FORECASTER integrates data construction, high-fidelity backtesting, and recursive strategy improvement to build better forecasting agents. (b) It progressively improves prediction performance from 50.16 to 65.70, surpassing existing RSI methods and strong coding agents. (c) It outperforms the general public and performs comparably to human superforecasters on ForecastBench's adjusted score.

1 INTRODUCTION

Event forecasting is important across many real-world domains, such as market, climate, politics and sports (Tsokos et al., 2019; Hegre et al., 2019; Ham et al., 2019; Liu et al., 2026a; Karger et al., 2025). Producing high-quality forecasts often requires a complex and highly task-specific research process, as the relevant information sources, historical analogues, analytical methods, predictive models, and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

required domain knowledge vary substantially between target questions. As a result, developing effective forecasting strategies has traditionally required substantial human expertise and manual effort. This creates a clear need for AI systems that can autonomously develop effective forecasting strategies tailored to each target question and be on par with human forecasting performances.

In this work, we formulate the development of the forecast strategy as a recursive self-improvement (RSI) process that starts from an initial naive strategy and progressively improves it through repeated execution, evaluation, and revision. However, RSI for forecasting faces a fundamental feedback deadlock: strategy improvement requires verification against the target outcome, yet that outcome remains unknown when the forecast must be issued. Our key idea is to break this deadlock by using resolved historical forecasting tasks, which share predictive structure with the target, as proxy evaluation tasks for RSI. However, this surrogate approach is non-trivial due to three systematic challenges: RSI Data requires autonomously constructing historical tasks that meaningfully guide the target; RSI Environment necessitates replaying these tasks within a high-fidelity backtesting framework; and RSI Method demands iteratively improving strategies so that performance gains transfer to the unresolved target instead of overfitting the historical data.

To tackle these bottlenecks, we develop targeted strategies for each dimension and introduce RSI-FORECASTER, an end-to-end RSI system for forecasting. Our motivating experiments (details in § 3) quantify the substantial technical gaps along all three dimensions. For RSI Data, using a naive dataset under a standard RSI method improves the target set Brier Index (BI) from 46.33 to just 47.35; in contrast, our agentic data solution reaches 52.65 BI. For RSI Environment, archived and leaky backtesting yield marginal target BI improvements (reaching 47.32 and 47.89, respectively), compared to the 52.65 enabled by our environment. For RSI Method, even when provided with our robust data and environment, only three out of five existing RSI methods outperform the non-RSI baseline (a direct forecast ensemble). These methods yield a target BI gain of just 1.57–4.94 points, falling far short of the 14.62-point gain achieved by our method. Our contributions are threefold:

- We formulate autonomous forecasting strategy development as a recursive self-improvement (RSI) problem. To enable this improvement loop, we leverage historical proxy tasks and empirically identify three systematic bottlenecks: RSI Data, RSI Environment, and RSI Method.
- We propose RSI-FORECASTER, an end-to-end system designed to resolve the bottlenecks. It features a data agent to construct target-relevant historical tasks; an open-web backtesting environment that balances information sufficiency with leakage control; and a robust RSI method that uses predictive and process feedback to guide hierarchical strategy refinement, followed by held-out selection and target adaptation.
- We conduct extensive live evaluations on ForecastBench (Karger et al., 2025) and FutureX (Liu et al., 2026a;b) over three months, covering $\sim$1,000 questions. Our submissions consistently maintain top-three performance, competitive with private systems from frontier labs and specialized forecasting companies. Comprehensive ablations and cross-LLM experiments further validate our design choices and demonstrate the broad applicability of our approach.

Code is available at `anonymous.4open.science/r/RSF/`. See related work in App. A, future work and limitations in App. B.

2 PROBLEM SETUP

Forecasting task. A forecasting question $q_0 = (s_0, t_0, h_0, \rho_0)$ specifies its semantics, forecast cutoff, horizon, and resolution rule. An executable strategy π produces a forecast $\hat{y}_0^\pi$ using information available by t_0, before the outcome y_0 is known. Once outcomes are observed, performance on a question set S is measured by $J_S(\pi) = f((\hat{y}_i^\pi)_{i\in S}, (y_i)_{i\in S})$. Here, f is a task-appropriate evaluation metric, such as the Brier Index for probability forecasts (§ 5). **Autonomous strategy improvement.** Given q_0, an initial executable strategy π_0, and budget B, we study how a system can autonomously acquire feedback and improve π_0 before issuing the target forecast. The target outcome is unobserved, so target predictive scores are unavailable during improvement. The target question, evaluation metric, and backbone LLM weights remain fixed.

3 GAPS: WHAT DOES RSI REQUIRE FOR FORECASTING?

Given an unresolved target question q_0, we aim to enable RSI to recursively transform a naive forecasting strategy into a strong strategy specialized to that target, without per-target human effort.

Table 1: Controlled experiments for RSI Data and Environment. We fix the backbone LLM and RSI method. Our proposed Data and Environment (Secs. 4.1, 4.2) produce the largest Target BI improvement, from 46.33 to 52.65, while all alternatives remain below 48.

LLM: Gemini-3.5-Flash
RSI Method: Meta-Harness

RSI Data	RSI Env	Target BI ↑
Before RSI		46.33
Naive	Our	47.35
Our	Archived	47.32
Our	Leaky	47.89
Our	**Our**	**52.65**

Table 2: Comparison of RSI methods using the same backbone LLM, data, and environment. Each method runs for eight iterations. Direct Forecast Ensemble averages forecasts from eight runs. All five existing RSI methods improve both Research and Target BI, but only three exceed the non-RSI method, by 1.57–4.94 BI points. Our RSI-FORECASTER (Sec. 4.3) achieves the highest Target BI (65.70), with a 14.62-point gain.

LLM: Gemini-3.5-Flash *RSI Data and Env: Ours*

Method	Research Set BI ↑ · Before → After RSI	Target BI ↑ · Before → After RSI
Direct Forecast Ensemble (non-RSI)	—	51.08
A-Evolve (Lin et al., 2026)	62.45 → 70.44	53.69 → 56.02
Adaptive Auto-Harness (Liu et al., 2026c)	65.47 → 70.65	42.55 → 54.28
GEPA (Agrawal et al., 2026)	62.26 → 67.44	44.13 → 44.79
Meta-Harness (Lee et al., 2026)	68.07 → **76.44**	46.33 → 52.65
Continual Harness (Karten et al., 2026)	64.86 → 74.90	46.65 → 48.78
RSI-FORECASTER (Ours)	62.15 → 65.78	50.16 → **65.70**

The fundamental challenge of RSI for forecasting is that the target outcome remains unknown throughout self-improvement. Specifically, a proposed strategy revision can be executed on q_0, but its predictive quality cannot be verified since q_0 is not resolved.

We propose to construct historical proxy tasks with known outcomes to provide feedback before q_0 resolves. Turning these tasks into effective target-specific self-improvement introduces three tightly connected gaps: (1) *RSI Data* concerns how to construct historical proxy tasks that provide useful feedback for improving q_0. (2) *RSI Environment* concerns whether these historical tasks can be replayed with high fidelity. (3) *RSI Method* concerns whether improvements evolved from historical proxy tasks remain effective on the unresolved future target.

We examine how well naive choices of proxy-task data and backtesting environments, together with existing RSI methods, support RSI for forecasting. For reference, we also preview results from our solutions, described in Sec. 4. We fix the backbone LLM to Gemini-3.5-Flash and evaluate on 200 target questions. We report Brier Index (BI ↑). Research-set BI and Target BI measure predictive performance on historical proxy tasks used during self-improvement and on target questions, respectively. We repeat three times and report the average. Experimental details are in App. C.

RSI Data: constructing effective historical proxy tasks. We first quantify how historical-task construction affects RSI. We fix the RSI method to Meta-Harness (Lee et al., 2026) and the execution environment to our proposed environment (Sec. 4.2), varying only the proxy tasks. As shown in Tab. 1, before self-improvement, the initial strategy achieves a Target BI of 46.33. With rule-retrieved historical proxy tasks (marked as *Naive*, detailed in App. F), Meta-Harness improves Target BI only to 47.35. By contrast, with our agentic RSI data solution (see Sec. 4.1), Target BI reaches 52.65. Historical-task construction is foundational to RSI because it shapes the successes and failures RSI learns from. The relevant dimensions vary across domains: geographic location and seasonality can dominate weather forecasts, whereas competition format and tournament stage are important for sports forecasts. These requirements call for flexible, fine-grained task construction tailored to each target. Our proposed agentic approach improves Target BI by 5.30 points over rule-based retrieval.

RSI Environment: ensuring high-fidelity backtesting. The outcomes of these historical proxy tasks are already known and often accessible through multiple ways on the web. Unrestricted access can therefore expose the final answer or post-resolution reports during RSI. On the other hand, restricting execution to web archives can substantially mask the evidence originally available to the forecaster, especially information close to the cutoff, leading to a biased forecasting strategy.

For empirical evidence, we isolate the execution environment by fixing RSI Data to our proposed data construction (Sec. 4.1) and the RSI method to Meta-Harness. As shown in Tab. 1, RSI in an archived backtesting environment (based on the Wayback Machine) produces a Target BI of 47.32; leaky backtesting (open-web but controlled only through an LLM prompt; see App. D) reaches 47.89. Under our proposed environment, Target BI rises to 52.65.

Evidence audits using Claude-Sonnet-5 and Gemini-3.8 reveal that 83% of retrieved search results in prompt-only backtesting contain post-cutoff information, while 73% in archived backtesting predate

the forecast cutoff by more than 30 days. Thus, we introduce our RSI environment solution in Sec. 4.2 to balance leakage and sufficiency.

RSI Method: recursively improving forecasting strategies for the target. Finally, we ask whether existing RSI methods are effective enough for forecasting once the RSI Data and Environment gaps are addressed. We fix the backbone LLM and use our RSI Data and Environment (Secs. 4.1 and 4.2) to compare A-Evolve (Lin et al., 2026), Adaptive Auto-Harness (Liu et al., 2026c), GEPA (Agrawal et al., 2026), Meta-Harness (Lee et al., 2026), and Continual Harness (Karten et al., 2026). We adapt each method to forecasting with minimal changes to its original implementation. Each method proposes, implements, and executes its own initial forecasting strategy. Each RSI method runs for eight iterations. As a non-RSI reference but with multiple runs, the direct forecast ensemble averages the predicted probabilities from eight independent runs without iterative revision.

As shown in Tab. 2, all five existing RSI methods improve both Research-set BI and Target BI relative to their initial performances. Yet these results raise concerns about overfitting to historical tasks: (1) Meta-Harness achieves the highest final Research-set BI (76.44), but its Target BI of 52.65 ranks third among the five existing methods. (2) Only three of the five methods exceed the non-RSI method's 51.08 Target BI, with marginal gains of 1.57–4.94 points. These results highlight the risk of overfitting in RSI (Wang et al., 2026), a particular concern for forecasting, which requires learning strategies that transfer from historical tasks to target questions.

To address this challenge, our RSI-FORECASTER uses predictive and process feedback to guide a hierarchical loop, followed by held-out selection and target adaptation. Under the same forecasting setup, it improves Target BI from 50.16 to 65.70, outperforming the strongest existing RSI method at 56.02 and the direct forecast ensemble by 14.62 points, despite a modest final Research-set BI of 65.78. Our 9.68-point lead over the strongest RSI baseline exceeds five times our Target BI standard deviation across three complete runs (1.9 points).

The above experiments identify three fundamental gaps in RSI for forecasting: **Data**: historical tasks must provide useful guidance; **Environment**: historical execution must yield faithful feedback; **Method**: strategy improvements must transfer to the target. We next introduce our corresponding solutions, which together form RSI-FORECASTER, our proposed RSI system for forecasting.

4 PROPOSED SOLUTIONS: RSI-FORECASTER

Given an unresolved target question q_0, RSI-FORECASTER constructs historical proxy tasks that enable proxy verification, backtests candidate strategies on these tasks in a high-fidelity environment, and recursively improves the strategies to transfer to the unresolved target. Fig. 2 provides an overview of RSI-FORECASTER. We detail our RSI system including RSI Data, Environment, and Method.

4.1 RSI DATA SOLUTION

The data challenge is to construct historical proxy tasks whose known outcomes can evaluate strategy revisions for the unresolved target q_0. A useful proxy must preserve the target-relevant *predictive structure*, such as event-generating mechanism, forecast horizon, observation window, season or regime, and resolution semantics, which are not easily modeled by predefined rules (see Tab. 1).

Our multi-stage data agent combines LLM planning and judgment with scalable code-based retrieval. Given q_0, the LLM produces an executable retrieval plan specifying search queries, structured constraints, source priorities, and embedding- and metadata-based ranking code. The agent retrieves candidate tasks from three complementary sources: our archive of approximately 200,000 historical events curated from prediction markets such as Polymarket; cutoff-constrained open-web search via Google Search; and domain-specific APIs such as Open-Meteo, FRED, and Yahoo Finance. Programmatic filters and retrieval produce a Top-K candidate set. The data agent selects resolved historical tasks for relevance to the target's predictive structure. The LLM compares and selects the Top-M tasks, where $M < K$, and audits the selected set for mechanism coverage, redundant event families, temporal concentration, and source quality. If the audit identifies quality issues, the agent retrieves additional candidates to revise or replace the selected tasks.

Each selected historical task q_i has a traceable question card recording its question, outcome, resolution rule, source, forecast cutoff t_i, and the time τ_i when its outcome first became available. It is eligible only if $t_i < \tau_i \leq t_0$. Finally, the selected tasks are chronologically split by forecast cutoff

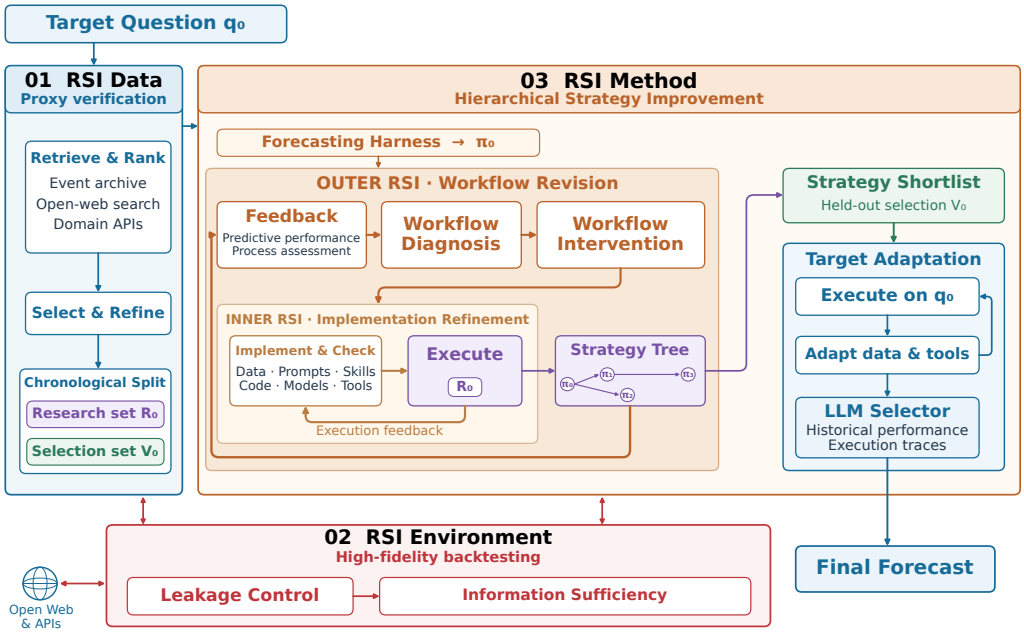


Figure 2: RSI-FORECASTER. Starting from a forecasting harness, the outer loop diagnoses workflow changes and the inner loop realizes them. Scores and process assessments feed a structured strategy lineage. Held-out performance produces a shortlist; target execution informs final selection or synthesis. Historical executions use their respective cutoffs t_i; target execution uses t_0.

into a research set $\mathcal{R}_0$ and a held-out selection set $\mathcal{V}_0$. $\mathcal{R}_0$ supplies iterative feedback, whereas $\mathcal{V}_0$ is for candidate shortlist selections. See an example in App. E.

4.2 RSI ENVIRONMENT SOLUTION

With historical proxy tasks in place, the next challenge is high-fidelity backtesting. Most existing forecasting backtests rely on archived webpages (Zou et al., 2022; Wildman et al., 2025; Chandak et al., 2026). In contrast, we use the open web to reproduce the search, page-reading, and data-access conditions of living forecasting, especially for long-tail domains and low-attention events sparsely covered by archives. However, the open web also exposes outcomes and reports published after the historical cutoff. As shown in Tab. 14, the central challenge is to preserve sufficient forecast-time evidence while preventing future-information leakage.

For each historical task q_i, our environment $\mathcal{E}_{\leq t_i}$ mediates tool access using its forecast cutoff t_i. To **control leakage**, it first screens sources using Google Search date restrictions and publication metadata, then opens retained pages for content-level inspection. This is necessary because search snippets can be incomplete and pre-cutoff pages can later be updated with outcomes. The LLM identifies post-cutoff spans and distinguishes them from admissible forecasts, plans, unrelated events, and dynamically dated page elements. The system removes these spans and rechecks the remaining content. For data APIs, it verifies release times and versions rather than using data timestamps.

Leakage control alone can make a backtest unrealistically information-poor. To preserve **information sufficiency**, the environment retains admissible content from partially contaminated pages instead of discarding entire sources. Because current search rankings often prioritize later resolution reports, it paginates, deduplicates, and screens additional results until the post-filtering evidence budget is met or the retrieval budget is exhausted. Historical API records can recover forecast-time signals no longer visible on updated webpages, subject to the same release-time checks.

The resulting evidence supports historical strategy execution, while the guard and evaluation rules remain fixed throughout RSI. See App. N for environment details.

4.3 RSI METHOD SOLUTION

With historical proxy tasks and a high-fidelity environment in place, the remaining goal is to iteratively improve forecasting strategies on these tasks and transfer the resulting improvements to the unresolved

target. Our loop combines predictive performance and process assessment to guide hierarchical revisions, then uses held-out evaluation and target execution to select and adapt these strategies. Fig. 2 presents the overall framework.

We represent a strategy as $\pi = (G, \theta, d)$, comprising a workflow G, its step implementations θ, and a data-and-modeling recipe d. Our forecasting harness provides shared interfaces for web research, data access, code and model execution, and prediction outputs. It records execution traces, with external information access mediated by the point-in-time environment. Given q_0 and these interfaces, an LLM proposes π_0 without historical feedback, guided by a coarse forecasting scaffold: clarify the task, research evidence, analyze it, and return a forecast. Each strategy, including its revisions, reads task-specific fields from the current question's forecast contract, allowing the same logic to run on historical tasks and the target. See App. G for details.

Predictive and Process Feedback. Forecasting outcomes are uncertain: sound reasoning can precede an unlikely outcome, while flawed reasoning can match the outcome by chance. We therefore evaluate both predictive performance and the forecasting process. Executing strategy π on historical task q_i under its point-in-time environment returns a prediction and trace, $(\hat{y}_i^\pi, z_i^\pi) = \mathrm{Exec}(\pi, q_i; \mathcal{E}_{\le t_i})$. The aggregate research-set score $J_{\mathcal{R}_0}(\pi)$ provides the predictive reward, supplemented by task-level errors to localize gains and failures. A fixed LLM reviewer provides revision guidance as a textual process assessment $\kappa_i^\pi = c_\psi(q_i, z_i^\pi)$, without being given the outcome label. It checks whether the reasoning is sufficiently supported by evidence available at forecast time, examining assumptions, computations, and tool outputs (see the example in App. J.1). This assessment guides evidence-grounded revisions alongside improvements in predictive performance. Scores, traces, and textual assessments across evaluated strategies form the accumulated feedback F_r available at round r.

Hierarchical Strategy Improvement. Improving a multi-step strategy requires deciding what to change and how to implement it. At round r, the outer controller O uses accumulated feedback F_r and the strategy tree $\mathcal{P}_r$ to diagnose which steps may explain gains or failures, select a parent, and propose a workflow intervention: $(\pi_r^{\mathrm{par}}, a_r) = O(q_0, \mathcal{P}_r, F_r)$. The intervention a_r specifies steps to add, remove, reorder, or revise and their intended effects. The inner agent I implements and refines the intervention through prompts, skills, data processing, model fitting, code, or tools, producing $\pi_r' \sim I(\pi_r^{\mathrm{par}}, a_r, F_r)$. It also supplies implementation checks, which the harness runs before historical evaluation to check interface compatibility, execution, and output validity. Evaluation on the same research tasks then checks whether the intended process change occurred and how predictive performance changed. This feedback guides implementation refinement and the next outer diagnosis.

Evaluated revisions are added to the strategy tree with their parent, intervention, executable strategy, tests, traces, and scores. These records support parent–child comparisons and continued search from earlier strategies. The loop runs for eight rounds, with the backbone LLM fixed.

Strategy Selection and Target Adaptation. After strategy search, we evaluate candidates on the held-out set $\mathcal{V}_0$ and form a shortlist $\mathcal{C}_0$. Each shortlisted strategy then executes on q_0 under $\mathcal{E}_{\le t_0}$. The adapter uses execution traces to adjust target-specific inputs and tool bindings while retaining the evaluated forecasting logic, then reruns the strategy as needed. Then, the LLM selector considers each candidate's research and selection performance, target prediction, and execution trace to assess target applicability and execution validity. App. J.2 illustrates how target traces inform this decision. It selects a candidate or combines their forecasts. The final forecast $\hat{y}_0$ is frozen before the target outcome becomes available. See details in App. H.

5 Main Results: Living Evaluation

Dataset and metrics. We evaluate RSI-Forecaster on two popular living forecasting benchmarks, ForecastBench (Karger et al., 2025) and FutureX (Liu et al., 2026a;b)[1], hosted by the Forecasting Research Institute and ByteDance Seed, respectively. ForecastBench questions resolve progressively over time, with horizons extending to several years. FutureX instead targets questions expected to resolve within seven days of submission. ForecastBench evaluates binary probability forecasts with the Brier Index (BI ↑) calculated from Brier score (BS) (Glenn et al., 1950), $\mathrm{BI} = 100(1 - \sqrt{\mathrm{BS}})$, where $\mathrm{BS} = \frac{1}{n}\sum_{i=1}^{n}(p_i - y_i)^2$ for predicted probabilities p_i and binary outcomes y_i. For FutureX

[1]Benchmark websites: ForecastBench and FutureX.

Table 3: ForecastBench living results, July–September. By September 16, 947 questions are resolved. BI uses the September 16 snapshot; Mean Rank averages ranks across updates. RSI-FORECASTER maintains top-three performance in every round as participation grows from about 75 to 100, including frontier labs and professional forecasting companies. — indicates no submission. For both tables, **bold** and underlining mark the best and second-best values, respectively; Blue: external; Black: organizer-hosted.

SUBMISSIONS	07-05		08-02		08-30	
	BI ↑	Mean Rank ↓	BI ↑	Mean Rank ↓	BI ↑	Mean Rank ↓
RSI-Forecaster	70.53	**1.62**	67.84	1.84	65.60	**1.83**
NeuroFusion	—	—	—	—	**66.16**	2.33
Torchcast AI	—	—	67.75	3.11	65.86	**1.83**
FutureSearch	**71.05**	2.50	**68.90**	**1.16**	—	—
Google DeepMind	71.04	2.23	—	—	—	—
Ryso AI	—	—	—	—	65.35	5.50
Anthral Labs	69.89	4.88	—	—	64.46	6.17
Yhat Labs	—	—	—	—	64.56	6.33
yarrow	—	—	66.92	5.63	64.16	9.33
LightningRodLabs	69.02	7.35	—	—	—	—
Artificial Judgement	69.38	4.69	67.06	6.95	62.81	12.33
Cassi-AI	—	—	—	—	64.19	9.67
Claude Fable5 Max	—	—	67.03	6.47	—	—
GPT5.6 Sol	—	—	66.43	14.89	64.86	4.17
Seldon	—	—	—	—	63.53	10.50
Vaticinus	67.41	8.42	66.90	6.00	58.38	21.67
Cenva Intelligence	—	—	65.31	15.26	64.51	8.33
Kerem Delikoyun	—	—	66.75	10.16	61.02	14.83
Phinomial Labs	—	—	—	—	62.37	12.67
GPT5.5	66.76	9.15	66.48	12.16	60.25	16.83
Gemini3.5 (**Our Backbone**)	67.23	7.00	66.49	11.47	59.11	21.17
KimiK3 Max	—	—	66.91	9.21	59.62	20.17
Grok4.20	66.64	11.96	66.35	14.05	60.23	16.17
MiniMax-M3	65.90	13.27	65.67	17.21	—	—
Claude Sonnet5	66.52	9.46	66.17	12.53	56.93	25.00
Gemini3.1 Pro	66.74	8.88	65.75	16.95	58.64	23.00
Claude Opus4.7	66.31	13.58	66.07	11.21	56.31	26.00
… *other submissions*						

Table 4: FutureX living results across three rounds. Mean F_1 ↑ and Mean Rank ↓ are averaged across June 17, June 24, and July 1 for the 27 organizations participating in all three rounds. RSI-FORECASTER achieves the highest mean F_1 and best rank. Per-round results are provided in Tab. 10. Our backbone LLM is Gemini-3.5-Flash for these two tables.

06-17 / 06-24 / 07-01		
SUBMISSIONS	Mean F_1 ↑	Mean Rank↓
RSI-Forecaster	**64.97**	**3.00**
React (RUC)	64.32	6.00
Doubao2.0 Pro	63.13	6.00
HAgent	63.12	8.67
GPT5.4 High	62.38	6.00
DeepSeekV4 Pro	61.22	8.00
Branchus	61.01	9.00
DragonAgent	60.94	10.33
Openclaw	60.71	9.00
Apodex-FutureFlow	60.32	11.67
DeepSeekV3.2 Thinking	60.21	12.67
Qwen3.7 Max	59.93	13.00
H2O AI Super Agent	59.82	10.67
Gemini3.1 Pro High	59.56	13.67
GLM5.1	59.54	13.00
JSearch_V1	59.01	13.67
Hermes	58.34	15.33
Grok4.2 Reasoning	58.31	13.33
MAS	57.69	15.67
Deepagents	57.38	17.33
Gemini3.5 (**Our Backbone**)	56.47	16.67
Qwen3.6 Max	55.38	18.33
Doubao2.0 Lite	54.87	18.67
MiniMax-M2.7	52.58	22.33
DeepSeekV4 Flash	49.66	21.00
MiniMax-M3	45.44	25.67
Grok4	22.79	26.33

Level-2 questions, we evaluate predicted answer sets $\widehat{Y}$ against correct sets Y using set-F_1 ↑, $F_1 = 2|\widehat{Y} \cap Y|/(|\widehat{Y}| + |Y|)$, reported as a percentage. **Baselines.** Both benchmarks attract active community participation. Alongside organizer-hosted LLM baselines, their leaderboards include external submissions from frontier labs such as Google DeepMind and Thinking Machines Lab, and specialized forecasting startups such as H2O.ai, Torchcast AI, NeuroFusion, FutureSearch, and Cassi-AI.[2] Together, these benchmarks enable us to assess RSI-FORECASTER against professional private forecasting systems. **Evaluation period and scale.** Our living evaluation spans 3 months and 3 rounds each of ForecastBench and FutureX with more than 1,000 resolved questions till September 16, 2026. This extended evaluation tests whether RSI-FORECASTER remains stable and competitive as participation grows. E.g., ForecastBench submissions increased from about 75 in July to 100 in September. We use Gemini-3.5-Flash as our LLM backbone. More setting details are in App. K.

ForecastBench Living Results. Tab. 3 reports results for the July 5, August 2, and August 30 rounds, with resolution updates tracked through September 16, 2026. BI uses the September 16 snapshot. Mean rank averages ranks across every official update within each round to assess performance

[2]Some forecasting company websites: H2O.ai, Torchcast AI, NeuroFusion, FutureSearch, and Cassi-AI.

stability. We find: (1) **Gains over hosted LLM baselines.** RSI-FORECASTER outperforms every listed organizer-hosted LLM baseline in both BI and mean rank in all three rounds. Several private systems also surpass these baselines. These results suggest that forecasting remains an underexplored capability despite rapid advances in frontier LLMs. (2) **Sustained leading performance against private submissions.** Our mean ranks are 1.62, 1.84, and 1.83, indicating sustained competitiveness across resolution updates. On the September 16 snapshot, our BI scores of 70.53, 67.84, and 65.60 still rank top three. Note that submissions increase from about 75 to 100.

FutureX Living Results. Tab. 4 summarizes the June 17, June 24, and July 1, 2026 rounds, covering 73 questions and 27 systems represented in all three rounds. We report mean F_1 and mean rank across rounds. We find: RSI-FORECASTER achieves the highest mean set-F_1 of 64.97, ahead of the leading competing private system, React (64.32), and the strongest organizer-hosted baseline, Doubao-Seed-2.0-pro (63.13). It also exceeds the Gemini-3.5-Flash baseline, our backbone, by 8.50 points (64.97 versus 56.47). Our mean rank of 3.00 is also the best. See details in Tab. 10.

Moreover, RSI-FORECASTER does not fine-tune its backbone LLM, while some private submissions report training forecasting-specific foundation models.

Comparison with Human and Crowd Forecasts. Using the same ForecastBench results reported above, we compare RSI-FORECASTER against human forecasting baselines at three levels of difficulty: the general public, crowd (prediction market participants), and superforecasters with a record of consistently high accuracy.

(1) With the public and superforecasters. ForecastBench officially reports the performances of superforecasters and members of the general public. To enable comparison across different question sets and rounds, ForecastBench applies difficulty-adjusted Brier Index (App. L). Under this protocol, RSI-FORECASTER achieves an overall BI of 69.6, compared with 68.8 for superforecasters and 62.6 for the public (Fig. 1(c)). The detailed breakdown in App. Tab. 11 shows: (1) On dataset questions, generated from real-world time series, RSI-FORECASTER scores higher than superforecasters (65.9 vs. 63.6); (2) on market questions, superforecasters score slightly higher (75.0 vs. 73.8). (3) RSI-FORECASTER substantially exceeds the general public's performance on both question types (65.9 vs. 59.2 and 73.8 vs. 66.3, respectively).

(2) With crowd predictions. On 213 market questions from the same three rounds, our forecasts outperform the corresponding market crowd predictions (crawled by ForecastBench), with BI scores of 72.42 versus 68.57 (Fig. 4; error bars show 95% confidence intervals). To test whether this gain can be explained simply by providing the crowd forecast to the backbone, we additionally give the same crowd prediction to Gemini-3.5-Flash (Fig. 5); it reaches only 69.12. Besides, RSI-FORECASTER beats crowd predictions on 154 of 213 questions (72.3% win rate), with 8 ties and 51 losses.

Overall, these results demonstrate that RSI-FORECASTER outperforms the public and even crowd predictions, approaching the performance of human superforecasters.

6 ANALYSIS RESULTS

Setup. We use backtesting for ablation and analysis because thorough experiments exceed FORECASTBENCH's 24-hour living submission window. We use the backtesting environment in Sec. 4.2; App. N provides empirical evidence of its fidelity. We randomly sample 200 ForecastBench questions resolved after May 2026, beyond the knowledge cutoffs of all LLMs considered. Note that we cannot fully exclude LLM knowledge exposure to historical proxy tasks, given the rapid evolution of LLMs. We construct 12 historical proxy tasks per target: 8 for research and 4 for held-out selection, given computational resource constraints. Preliminary observations suggest that using more historical tasks improves forecasting performance. Unless stated otherwise, we use Gemini-3.5-Flash, 8 RSI iterations, up to 20 web searches per strategy per question, and a shortlist of 3 strategies for our RSI method. We compute Target BI from the pooled Brier score across questions. See details in App. I.

RSI Data and Environment Ablations. As shown in Fig. 3: (1) Replacing the data agent with rule-based or embedding-only retrieval lowers Target BI from 65.70 to 54.09 or 57.99. See App. F for retrieval details and examples. These results support our agentic RSI Data pipeline (Sec. 4.1), which combines target-specific retrieval, task-quality checks, and refinement. (2) Replacing our environment with archived (via Wayback Machine) or leaky (see App. D) backtesting yields 55.35 or 51.65. This verifies the advantage of our RSI Environment (Sec. 4.2) to balance information sufficiency and

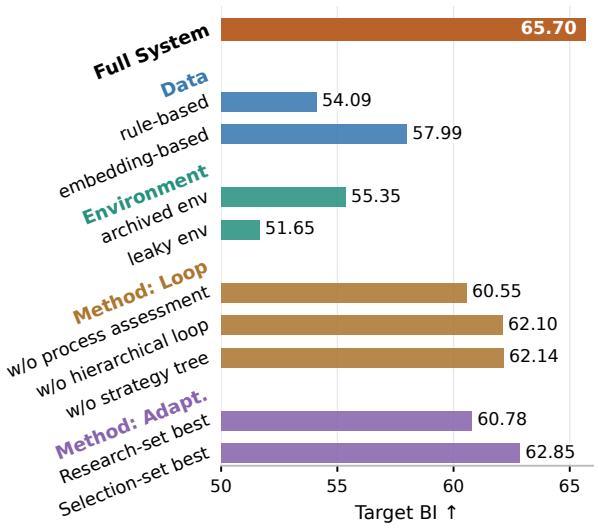


Figure 3: Ablation study of RSI-FORECASTER.

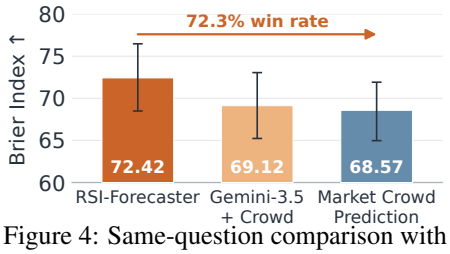


Figure 4: Same-question comparison with market crowd forecasts. See Fig. 1(c) for the cross-round comparison.

Table 5: With different backbone LLMs.

LLM Backbone	Before RSI	After RSI	SciCode Index
Gemini-3.5	50.16	**65.70**	54
Gemini-3.8	61.16	**65.24**	57
Sonnet-5	56.70	**66.27**	54
Gemma-4-26A4B	50.08	**61.04**	40
Qwen3-80A3B	50.76	**59.30**	33

leakage. (3) Replacing either our RSI Data or Environment design substantially reduces performance: all four replacements score below 58, compared with 65.70 for the full system.

RSI Method Ablations. Loop. We hold RSI Data and Environment fixed to assess the contributions of loop components (Sec. 4.3). Removing process assessment (LLM review of evidence and reasoning), the hierarchical loop (separate diagnosis and implementation), or the strategy tree (parent–child revision history) reduces Target BI from 65.70 to 60.55, 62.10, or 62.14, respectively (Fig. 3). Among these ablations, removing process assessment causes the largest drop in Target BI, from 65.70 to 60.55, leaving only predictive performance as feedback. The hierarchical loop and strategy tree further help identify where and how to improve strategies. See App. J.1 for concrete examples. **Selection and adaptation.** Selecting a single strategy by its research-set or held-out performance yields 60.78 or 62.85 Target BI, respectively. Our combined full adaptation procedure reaches 65.70.

Across LLM Backbones. RSI improves all five backbones—Gemini-3.5-Flash, Gemini-3.8-Flash, Claude-Sonnet-5, Gemma-4-26B-A4B, and Qwen3-Next-80B-A3B—by 4.08–15.54 Target BI points over their initial performance (Tab. 5). All LLM components use the selected backbone. To help interpret the backbone differences, we report SciCode (Tian et al., 2024) scores from the third-party evaluator Artificial Analysis, which assess scientific research coding performance. The three backbones with SciCode scores above 50 (54, 57, and 54) start at 50.16–61.16 forecasting Target BI but converge to similar scores of 65.24–66.27 after our RSI. Gemma-4 and Qwen3, with lower coding scores of 40 and 33, also improve in Target BI but only reach 61.04 and 59.30, respectively. As a reference, the 200 recorded LLMs have a median SciCode score of 48 and a maximum of 67. These results support broad applicability: RSI improves all five tested backbones, including those with modest coding scores.

RSI Progress Analysis. As shown in Fig. 1(b): (1) Target BI shows an overall upward trend across iterations, rising from 50.16 to 65.70 after eight rounds. The direct forecast ensemble, which averages eight independent forecasts, reaches only 51.08, highlighting RSI's advantage. (2) Our system achieves strong performance within a few iterations: it reaches 65.58 within four iterations and 61.41 after just two, already surpassing Gemini CLI with Gemini-3.5-Flash (60.76) and Claude Code with Claude-Sonnet-5 (61.05), two coding agents with powerful, complex harnesses. Round-wise scores and coding agent scores are in Tabs. 8 and 9, respectively. See App. J.1 for concrete examples.

7 CONCLUSION

We study recursive self-improvement for forecasting as a path toward autonomous strong forecasting performance. We introduce an end-to-end system RSI-FORECASTER, which systematically addresses the key data, environment, and method gaps. Across three months and over 1,000 resolved questions, RSI-FORECASTER maintains top-three performance amid strong competition.

AI USE STATEMENT

In this work, we used generative AI tools for cleaning and reformatting datasets because we are doing agentic evaluation research.

We have not used generative AI tools for proposing or refining hypotheses, developing conceptual frameworks, designing research methodology or experiments, implementing core methods, or interpreting results.

Generating synthetic datasets, formulating mathematical claims, providing critical ingredients for mathematical proofs, assisting in the writing of proofs, supporting qualitative and thematic data analysis, and translation are not applicable to this work.

Additionally, we used generative AI tools for editing the manuscript to improve readability, suggesting structure and organization, and editing auxiliary software code.

We have thoroughly reviewed all AI-assisted work: all data-processing scripts and baseline utilities were unit-tested and manually debugged, and all polished text was cross-checked line-by-line by the authors to ensure clarity, factual accuracy, and that no technical interpretations were altered. We take full responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, et al. Gepa: Reflective prompt evolution can outperform reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 8479–8565, 2026.

Nikos I Bosse, Peter Mühlbacher, Jack Wildman, Lawrence Phillips, and Dan Schwarz. Automating forecasting question generation and resolution for ai evaluation. *arXiv preprint arXiv:2601.22444*, 2026.

Nikhil Chandak, Shashwat Goel, Ameya Prabhu, Moritz Hardt, and Jonas Geiping. Curating the future: A scalable recipe for training open-ended forecasters. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=SiMYGtHfxT.

Yi Duan, Ying Liu, Zirui Tang, Haodong Chen, Jun Zhou, Yumou Liu, Bangrui Xu, Yukai Wu, Sidi Chen, Yuhan Zhou, et al. The last ai built by humans: Toward genuine recursive self-improvement. *arXiv preprint arXiv:2609.11873*, 2026.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. *arXiv preprint arXiv:2309.16797*, 2023.

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

Danny Halawi, Fred Zhang, Chen Yueh-Han, and Jacob Steinhardt. Approaching human-level forecasting with language models. *Advances in Neural Information Processing Systems*, 37: 50426–50468, 2024.

Yoo-Geun Ham, Jeong-Hwan Kim, and Jing-Jia Luo. Deep learning for multi-year enso forecasts. *Nature*, 573(7775):568–572, 2019.

Prannay Hebbar, Yogendra Manawat, Samuel Verboomen, Alesia Ivanova, Selvam Palanimalai, Kunal Bhatia, and Vignesh Baskaran. Sia: Self improving ai with harness & weight updates. *arXiv preprint arXiv:2605.27276*, 2026.

Håvard Hegre, Marie Allansson, Matthias Basedau, Michael Colaresi, Mihai Croicu, Hanne Fjelde, Frederick Hoyles, Lisa Hultman, Stina Högbladh, Remco Jansen, et al. Views: A political violence early-warning system. *Journal of peace research*, 56(2):155–174, 2019.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025.

Ezra Karger, Houtan Bastani, Chen Yueh-Han, Zachary Jacobs, Danny Halawi, Fred Zhang, and Philip E Tetlock. Forecastbench: A dynamic benchmark of ai forecasting capabilities. In *International Conference on Learning Representations*, volume 2025, pp. 93943–93980, 2025.

Seth Karten, Joel Zhang, Tersoo Upaa Jr, Ruirong Feng, Wenzhe Li, Chengshuai Shi, Chi Jin, and Kiran Vodrahalli. Continual harness: Online adaptation for self-improving foundation agents. *arXiv preprint arXiv:2605.09998*, 2026.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T Joshi, Hanna Moazam, et al. Dspy: Compiling declarative language model calls into self-improving pipelines. *arXiv preprint arXiv:2310.03714*, 2023.

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-harness: End-to-end optimization of model harnesses. *arXiv preprint arXiv:2603.28052*, 2026.

Minhua Lin, Hanqing Lu, Zhan Shi, Bing He, Rui Mao, Zhiwei Zhang, Zongyu Wu, Xianfeng Tang, Hui Liu, Zhenwei Dai, et al. Position: Agentic evolution is the path to evolving llms. *arXiv preprint arXiv:2602.00359*, 2026.

Jiashuo Liu, Siyuan Chen, Tianci He, Yali Liao, Yixiao Tian, Jinpeng Wang, Zaiyuan Wang, Lingyue Yin, Mingren Yin, Zhu Zhenwei, et al. Futurex: An advanced live benchmark for llm agents in future prediction. In *International Conference on Learning Representations*, volume 2026, pp. 19522–19563, 2026a.

Jiashuo Liu, Siyuan Chen, Zaiyuan Wang, Zhiyuan Zeng, Jiacheng Guo, Liang Hu, Lingyue Yin, Suozhi Huang, Wenxin Hao, Yang Yang, et al. Futurex-pro: Extending future prediction to high-value vertical domains. *arXiv preprint arXiv:2601.12259*, 2026b.

Zewen Liu, Zhan Shi, Yisi Sang, Bing He, Minhua Lin, Tianxin Wei, Dakuo Wang, Benoit Dumoulin, Wei Jin, and Hanqing Lu. Adaptive auto-harness: Sustained self-improvement for agentic system deployment on open-ended task streams. *arXiv preprint arXiv:2606.01770*, 2026c.

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.

Barbara Mellers, Lyle Ungar, Jonathan Baron, Jaime Ramos, Burcu Gurcay, Katrina Fincher, Sydney E Scott, Don Moore, Pavel Atanasov, Samuel A Swift, et al. Psychological strategies for winning a geopolitical forecasting tournament. *Psychological science*, 25(5):1106–1115, 2014.

Barbara Mellers, Eric Stone, Pavel Atanasov, Nick Rohrbaugh, S Emlen Metz, Lyle Ungar, Michael M Bishop, Michael Horowitz, Ed Merkle, and Philip Tetlock. The psychology of intelligence analysis: drivers of prediction accuracy in world politics. *Journal of experimental psychology: applied*, 21 (1):1, 2015.

Zhe Ren, Yimeng Chen, Dandan Guo, Guowei Rong, Tonghui Li, RB Xiong, Qingfeng Lan, Wenyi Wang, Li Nanbo, Yibo Yang, et al. Self-improvements in modern agentic systems: A survey. *arXiv preprint arXiv:2607.13104*, 2026.

Philipp Schoenegger and Peter S Park. Large language model prediction capabilities: Evidence from a real-world forecasting tournament. *arXiv preprint arXiv:2310.13014*, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Minyang Tian, Luyu Gao, Shizhuo D Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, et al. Scicode: A research coding benchmark curated by scientists. *Advances in Neural Information Processing Systems*, 37:30624–30650, 2024.

Alkeos Tsokos, Santhosh Narayanan, Ioannis Kosmidis, Gianluca Baio, Mihai Cucuringu, Gavin Whitaker, and Franz Király. Modeling outcomes of soccer matches. *Machine Learning*, 108(1): 77–95, 2019.

Yike Wang, Huaisheng Zhu, Zhengyu Hu, Yige Yuan, Zhengyu Chen, Shakti Senthil, Hannaneh Hajishirzi, Yulia Tsvetkov, Pradeep Dasigi, and Teng Xiao. Rethinking the evaluation of harness evolution for agents. In *COLM 2026 The 2nd Workshop on Lifelong Agents: Learning, Aligning, and Evolving*, 2026.

Jack Wildman, Nikos I Bosse, Daniel Hnyk, Peter Mühlbacher, Finn Hambly, Jon Evans, Dan Schwarz, Lawrence Phillips, et al. Bench to the future: A pastcasting benchmark for forecasting agents. *arXiv preprint arXiv:2506.21558*, 2025.

Qi Yan, Raihan Seraj, Jiawei He, Lili Meng, and Tristan Sylvain. Autocast++: Enhancing world event prediction with zero-shot ranking-based context retrieval. In *International Conference on Learning Representations*, volume 2024, pp. 34523–34539, 2024.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations*, volume 2024, pp. 12028–12068, 2024.

Chenchen Ye, Ziniu Hu, Yihe Deng, Zijie Huang, Mingyu Derek Ma, Yanqiao Zhu, and Wei Wang. Mirai: Evaluating llm agents for event forecasting. *arXiv preprint arXiv:2407.01231*, 2024.

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.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. Textgrad: Automatic" differentiation" via text. *arXiv preprint arXiv:2406.07496*, 2024.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, et al. Aflow: Automating agentic workflow generation. In *International Conference on Learning Representations*, volume 2025, pp. 34040–34077, 2025.

Andy Zou, Tristan Xiao, Ryan Jia, Joe Kwon, Mantas Mazeika, Richard Li, Dawn Song, Jacob Steinhardt, Owain Evans, and Dan Hendrycks. Forecasting future world events with neural networks. *Advances in Neural Information Processing Systems*, 35:27293–27305, 2022.

Contents of Appendix

A Related Work **14**

B Limitations and Future Work **14**

C Motivation Experiments: Setup and Details **14**

D Prompt-only backtesting. **14**

E An Example of Historical Task Construction **14**

F Historical Proxy Task Retrieval Baselines and Examples **15**
F.1 Rule-based Retrieval 15
F.2 Embedding-only Retrieval 15

G Forecasting Harness and Initial Forecasting Strategy **15**

H Strategy Improvement and Target Prediction **16**

I Analysis Comparisons and Round-wise Results **18**

J Concrete Examples **19**
J.1 From historical feedback to concrete strategy revisions 19
J.2 Using target traces to distinguish candidates 19

K Living Benchmarks and Reporting Protocols **20**
K.1 Living Configuration 20

L ForecastBench Human Baselines and Difficulty Adjustment **20**

M Mean Rank on a Continuously Updating Benchmark **21**

N RSI Environment Evaluation **22**

A Related Work

Early language-agent methods refine outputs or retain verbal feedback (Madaan et al., 2023; Shinn et al., 2023); later work optimizes prompts and modular LM programs (Yang et al., 2024; Fernando et al., 2023; Yuksekgonul et al., 2024; Khattab et al., 2023), searches executable agents and workflows (Hu et al., 2025; Zhang et al., 2025), or evolves agent harnesses (Agrawal et al., 2026; Lin et al., 2026; Lee et al., 2026; Karten et al., 2026; Liu et al., 2026c). Recent surveys organize RSI by what and how autonomously a system improves, spanning model parameters, harnesses, data, and increasingly the improvement process itself (Duan et al., 2026; Ren et al., 2026).

Forecasting tournaments show that training, collaboration, repeated updating, and aggregation improve human probabilistic judgment (Mellers et al., 2014; 2015). Recent work evaluates LLMs against human forecasters (Karger et al., 2025; Halawi et al., 2024; Schoenegger & Park, 2023), builds forecasting questions paired with dated news (Zou et al., 2022; Yan et al., 2024), and provides tool-using agents with structured event histories (Ye et al., 2024). Other systems automate question generation and resolution (Bosse et al., 2026) or evaluate forecasters against a frozen historical information source (Ye et al., 2026).

RSI-Forecaster keeps the backbone LLM fixed and recursively improves its executable forecasting strategy. Forecasting introduces a distinct challenge: the unresolved target provides no direct evaluation signal. We therefore construct the feedback substrate for RSI itself—target-relevant resolved historical tasks provide proxy evaluation, while a leakage-controlled open-web environment supports faithful point-in-time execution—and then improve strategies to transfer this feedback to the unresolved target. See motivations in Sec. 3 and solutions in Sec. 4.

B Limitations and Future Work

Our RSI-Forecaster keeps the backbone LLM weights fixed and performs self-improvement at the executable-strategy level, enabling compatibility with both open- and closed-weight models. Future work could extend this framework to parameter-level RSI by using the outcome-grounded agent trajectories generated on historical proxy tasks for post-training, thereby combining strategy evolution with model-weight updating (Hebbar et al., 2026).

Our results on two popular living forecasting benchmarks, ForecastBench and FutureX, demonstrate the effectiveness of RSI-Forecaster for binary probability and multiple-choice forecasting. A promising next step is to extend the framework to continuous predictive distributions, such as those over inflation rates or GDP growth. This extension calls for dedicated living evaluation beyond the scope of this work.

C Motivation Experiments: Setup and Details

We sample 200 resolved forecasting questions from ForecastBench, all with resolution dates after May 2026. We conduct the ablations through historical backtesting, which provides more controlled comparisons than living deployment. For our Data Agent, we use $K = 100$ retrieved candidates and $M = 12$ selected historical proxy tasks per target. Within each ablation, we evaluate each question under all conditions on the same day. We report Brier Index (BI), where higher values indicate better probabilistic forecasts.

D Prompt-only backtesting.

The leaky baseline uses the open web and relies solely on a prompt instructing the forecasting LLM to ignore information from after each historical forecast cutoff.

E An Example of Historical Task Construction

For a target asking whether SpaceX will launch another Starship before a specified deadline, the Data Agent searches the historical event archive and the web for resolved questions about earlier launches. It excludes contracts concerning the target launch and groups candidates by launch event, so multiple time-threshold contracts for one flight do not dominate the historical set. For each retained question, the agent checks that its forecast cutoff precedes outcome publication and that the outcome was

Table 6: Shared interfaces for executable forecasting strategies.

Interface	Role in strategy execution
Task binding	Exposes the question's predicate, entity, cutoff, horizon, observation window, threshold, units, and resolution rule.
Evidence access	Supports search, webpage reading, API queries, and file or table access through the point-in-time environment.
Computation	Provides an execution interface for data processing, feature construction, statistical and time-series analysis, model fitting, and inference.
Prediction and traces	Returns the required prediction and records evidence sources, assumptions, computations, tool outputs, and fallback use.

known before the target cutoff. It then checks the set for variation in outcomes, clear resolution rules, and enough eligible tasks, refining the selection as needed. The selected historical questions are then split chronologically by forecast cutoff into research and held-out selection tasks.

F Historical Proxy Task Retrieval Baselines and Examples

We describe rule-based and embedding-only retrieval of historical proxy tasks and illustrate their limitations with selected tasks.

F.1 Rule-based Retrieval

This baseline scores each candidate historical task using five rules: (1) the same source platform as the target; (2) a similar forecast horizon; (3) resolution within the 90 days before the target forecast cutoff; (4) at least one shared word in the question text after removing stopwords and words shorter than three characters; and (5) matching presence or absence of threshold expressions in the question text. Each satisfied rule contributes one point. Candidates are selected by total score, with ties broken randomly.

Failure case: attribute matches without mechanism relevance. For a target asking whether SpaceX would launch Starship again before a specified deadline, this baseline selected "Kilmarnock vs Dundee FC Winner?" Both questions have short forecast horizons and are classified as non-threshold questions by the rules, earning points for these matches. However, football outcomes and space launches depend on different evidence and event-generating mechanisms. This example shows that coarse attribute matching can select historical tasks with little relevance to the target's predictive mechanism.

F.2 Embedding-only Retrieval

This baseline uses Qwen3-Embedding-0.6B for semantic retrieval. Question text is encoded into 1024-dimensional vectors, and cosine similarity measures semantic proximity between target and candidate question text.

Failure case: concentration on one historical event. For the same Starship target, nine of the 12 selected historical tasks concern the same launch, asking whether it would occur before different times of day, such as 6:30, 6:40, or 6:50 PM. These candidates are topically relevant, but multiple time-threshold contracts for one event limit the selected set's event coverage and diversity.

G Forecasting Harness and Initial Forecasting Strategy

The forecasting harness provides reusable interfaces for implementing and executing strategies. It standardizes task inputs, evidence access, computation, prediction outputs, and execution records for process assessment and revision. Tab. 6 summarizes these interfaces.

Task binding and execution. Each question is represented by a forecast contract containing its task-specific fields. Each strategy, including its revisions, reads these fields at execution time, allowing the same forecasting logic to operate on historical questions and the target. The strategy representation

$\pi = (G, \theta, d)$ stores the workflow, step implementations, and data-and-modeling recipe d. The recipe can fit auxiliary predictive models while keeping the backbone LLM fixed. For each historical task q_i, the recipe is instantiated using evidence and fitting data available under $\mathcal{E}_{\leq t_i}$. Evidence records retain their sources and relevant timestamps, while computations retain their inputs and outputs. These observable artifacts, together with explicit analyses and tool outputs, form the execution trace z_i^π used in feedback.

Initial forecasting strategy. The seed proposer receives q_0 and the harness interfaces, before any historical feedback is available. A coarse forecasting scaffold guides it to produce a strategy that clarifies the input task, researches relevant evidence, analyzes it, and returns a forecast in the required format. The proposer conditions the forecasting logic on the target's structure; the resulting strategy reads entities, dates, thresholds, and other task fields from the forecast contract at execution time. This produces π_0, which is tested and evaluated on $\mathcal{R}_0$ to begin the loop.

Implementation checks. The inner agent can use the shared interfaces to write new processing routines, fit predictors, compose tools, and revise prompts or skills. It supplies implementation checks alongside each revision. The harness runs these checks and a short execution to check that the implementation loads, accepts the task input, and returns the required prediction and trace. For binary forecasts, output checks include a finite probability in $[0, 1]$. Check results guide execution repairs; historical evaluation then assesses forecasting quality.

H Strategy Improvement and Target Prediction

The research set $\mathcal{R}_0$ supplies feedback throughout strategy search; the held-out set $\mathcal{V}_0$ supports selection after search. The task sets, scoring rules, and information-access rules remain fixed, while the loop revises the forecasting strategy. Tab. 7 gives the procedure.

Feedback construction. Each research execution produces a forecast $\hat{y}_i^\pi$ and trace z_i^π. Outcome-based scoring provides $J_{\mathcal{R}_0}(\pi)$, supplemented by task-level errors to localize gains and failures. For probability forecasts, the aggregate score is the Brier Index and individual errors are Brier losses. Comparisons use the same tasks for each revision to track aggregate and task-level changes.

The fixed LLM reviewer receives the question and its forecast-time trace, with the realized outcome withheld. Its textual assessment $\kappa_i^\pi = c_\psi(q_i, z_i^\pi)$ checks whether the reasoning is sufficiently supported by evidence available at forecast time. It examines task interpretation, assumptions, computations, and the connection between evidence and forecast. The assessment suggests workflow steps for revision, citing trace evidence where support is insufficient. The outer controller and inner agent receive numerical scores and process assessments through F_r and can access research-set outcomes and execution records. Selection-set outcomes, scores, and execution traces are withheld from both during strategy search.

Diagnosis, implementation, and comparison. The outer controller reads the strategy tree and feedback to choose a parent strategy and an intervention a_r. The intervention specifies the workflow change, its supporting observations, and its expected effect. For example, an intervention may revise how a modeling step chooses seasonal training examples; the inner agent can implement that change through data filtering or weighting.

The inner agent implements the intervention, checks its execution, and refines it using feedback from the research tasks. The resulting feedback shows whether the intended intermediate change occurred and how predictive performance changed relative to the parent. The inner agent uses this feedback to refine the implementation, while the outer controller decides whether to continue, redirect, or replace the workflow intervention.

Strategy tree and budget. At round r, each node in $\mathcal{P}_r$ stores its parent, intervention, executable strategy, implementation checks, predictions, scores, process assessments, and execution traces. Retaining alternative branches supports parent–child comparisons and continued search from earlier strategies. After constructing the initial forecasting strategy, the outer loop runs for eight rounds, each involving workflow diagnosis and inner implementation refinement. Budget B covers the overall procedure. The backbone LLM and controller instructions remain fixed.

Table 7: The historical feedback loop and its application to the target. Process assessments use the question and execution trace; predictive scores use resolved outcomes.

	Strategy improvement and target prediction
	Input: q_0, $\mathcal{R}_0$, $\mathcal{V}_0$, per-task environments, forecasting harness, and budget B.
1	Propose π_0 from q_0 and the harness interfaces.
2	Check and repair π_0; evaluate it on $\mathcal{R}_0$ and initialize the strategy tree $\mathcal{P}_0$ and feedback F_0.
3	**Repeat for eight outer rounds:**
4	Outer RSI reads the current tree and feedback, and selects a parent and intervention.
5	**Repeat within the intervention's implementation budget:**
6	Inner RSI implements the revision, runs implementation checks, and repairs execution.
7	Evaluate the revised strategy on $\mathcal{R}_0$; record scores, traces, and process assessments in the tree; use this feedback to refine its implementation.
8	Use the updated records for the next outer diagnosis.
9	Freeze the strategy versions accumulated across rounds, evaluate and rank them on $\mathcal{V}_0$, and form $\mathcal{C}_0$.
10	Bind and execute each shortlisted strategy on q_0; adapt data and tool use as needed, retaining the resulting forecast and trace.
11	Select or combine forecasts using historical scores and target execution records; return $\hat{y}_0$.

Held-out selection. After search terminates, strategy versions accumulated across rounds are frozen and evaluated on $\mathcal{V}_0$, then ranked by $J_{\mathcal{V}_0}(\pi)$ to form a shortlist $\mathcal{C}_0$. Selection scores guide shortlisting and the final target decision, not strategy revision. Each shortlisted strategy retains both its research and selection performance.

Target adaptation and final selection. Each shortlisted strategy binds its task fields to q_0 and executes under $\mathcal{E}_{\leq t_0}$. The adapter inspects the resulting evidence coverage, computations, tool returns, and fallbacks. It can adjust a source binding, repair an input schema, or activate the strategy's fallback, then re-execute within the allocated budget. Adaptation aligns data and tool use with the target's conditions while retaining the evaluated forecasting logic.

The final LLM selector receives each candidate strategy, its $J_{\mathcal{R}_0}(\pi)$ and $J_{\mathcal{V}_0}(\pi)$, and its target forecast and execution trace. Using fixed instructions, it considers historical performance, target applicability, and execution validity, including evidence coverage, tool returns, and fallback use. It selects a candidate or combines their forecasts in the required output format. Probability combinations use nonnegative weights summing to one. The final forecast $\hat{y}_0$ is recorded before y_0 becomes available.

The framework also supports other prediction formats. For multiple-choice tasks such as FutureX, predictive feedback uses set-F_1. The selector uses candidates' historical scores, answer sets, and execution traces to produce the final answer set.

I Analysis Comparisons and Round-wise Results

Comparison definitions. In Fig. 3, data and environment variants replace the indicated component; "w/o" denotes a component removal relative to the full system. Data and environment replacements retain our RSI method. For both the motivation and analysis experiments, environment variants affect historical-task execution only; target execution uses the same environment across variants. Removing process assessment retains predictive feedback; "w/o hierarchical loop" combines workflow diagnosis and implementation in a single-level loop; removing the strategy tree removes structured revision history. The direct forecast ensemble combines eight independent forecasts without iterative revision.

"Best on research set" and "Best on selection set" choose by $J_{\mathcal{R}_0}$ and $J_{\mathcal{V}_0}$, respectively. The LLM selector additionally uses target execution records, either from all searched strategies or from the three-strategy shortlist. The final decisions use historical performance and target applicability and execution validity, as described in Sec. 4.3.

Table 8: Numerical values for Fig. 1(b).

Round	Research BI ↑	Target BI ↑
0	62.15	50.16
1	61.56	59.54
2	63.04	61.41
3	63.44	62.59
4	64.63	65.58
5	65.13	65.11
6	65.02	65.55
7	64.55	65.70
8	65.78	65.70

Table 9: Reference scores shown as horizontal lines in Fig. 1(b).

Reference	Target BI ↑
Gemini CLI (3.5flash)	60.76
Claude Code (Sonnet5)	61.05
Best RSI Baseline	56.02
Direct Forecast Ensemble	51.08

J CONCRETE EXAMPLES

We illustrate individual stages of strategy revision and candidate selection using development examples of a House disciplinary-action forecast. The target asks whether the US House of Representatives would censure, expel, or reprimand a member before April 1, 2026, with a forecast cutoff of March 29. The examples illustrate how feedback changes forecasting instructions and how target execution records inform selection.

J.1 FROM HISTORICAL FEEDBACK TO CONCRETE STRATEGY REVISIONS

Historical proxy task examples. For this target, we show one historical proxy task from the research set and one from the selection set. The research-set example asks whether a resolution to censure Rashida Tlaib would pass the House; the selection-set example asks whether Al Green would be censured before April. Both involve assessing whether legislative disciplinary action can pass before a deadline.

Initial strategy and historical feedback. The initial strategy follows a general sequence: clarify the question, establish an outside view, gather evidence, compute, and synthesize a forecast. On a historical question about whether a resolution to censure Rashida Tlaib would pass, it reports a probability of 0.45. The process reviewer identifies a specific inconsistency: executed code yields 0.6635, but the subsequent narrative changes the assumptions and reports 0.45 without recalculation. The reviewer flags this mismatch without receiving the outcome. The outer controller combines this assessment with the historical Yes outcome and recommends explicit parameter declarations and consistency between calculations and reported forecasts.

Strategy revisions. Search explores two revisions. One directly addresses the diagnosis by requiring parameter declarations and consistent calculations. The other expands the event analysis to cover the main route, a substitute route if the first fails, and the chance of further action over the remaining legislative calendar. For example, a blocked censure motion does not eliminate the possibility of a successor motion. The revised strategy explicitly examines such alternative routes before estimating the overall probability. This second revision changes the prediction on the historical task from 0.45 to 0.58. Subsequent revisions refine its parameter and calendar instructions. The selected strategy requires explicit sources or assumptions for parameters and recalculation when those assumptions change.

Target execution. We next examine the revised strategy's target execution. At the March 29 forecast cutoff, three days remain until April 1. The available crowd reference is 0.27 from March 19, when thirteen days remained. Both the initial and selected strategies interpret the target as requiring further disciplinary action. The initial strategy reports 0.22 without completing its proposed time-adjustment calculation. Following the selected strategy's instruction to model remaining-time risk, this execution uses a constant daily hazard approximation:

$$p_{\text{remaining}} = 1 - (1 - 0.27)^{3/13} \approx 0.070.$$

Across five executions of each fixed strategy on this target, mean Brier loss is 0.0338 for the initial strategy and 0.0187 for the selected strategy; the resolved outcome is No. The revised instructions turn a qualitative time adjustment into an explicit calculation during target execution.

A financial forecasting example. For a Bitcoin price-threshold question, revisions instead prioritize intraday crossing conditions, price and volatility inputs, and barrier-crossing analysis.

J.2 USING TARGET TRACES TO DISTINGUISH CANDIDATES

The LLM selector receives candidate strategies from the shortlist and compares their target execution traces. One candidate treats events preceding the supplied task window as satisfying the target condition. The selected candidate applies the window and distinguishes proposed disciplinary action from completed action. The selector favors this candidate based on how it uses the evidence.

K LIVING BENCHMARKS AND REPORTING PROTOCOLS

ForecastBench. ForecastBench[3] continuously evaluates probability forecasts on unresolved binary questions and scores them as outcomes become available. Its tournament accepts externally submitted systems, including agents with tools, ensembles, and forecasting-specific training. For probabilities p_i and outcomes $y_i \in \{0, 1\}$, the Brier score and Brier Index are

$$\mathrm{BS} = \frac{1}{n}\sum_{i=1}^{n}(p_i - y_i)^2, \qquad \mathrm{BI} = 100(1 - \sqrt{\mathrm{BS}}).$$

The underlying Brier score is a proper score for probabilistic forecasts (Glenn et al., 1950). Higher BI is better; constant 0.5 forecasts have a BI of 50. We began with the market subset in the July 5 round and expanded our coverage to all questions in the August 2 and August 30 rounds. For July 5, all reported entries are compared on the same market questions.

Tab. 3 reports BI on the September 16, 2026 snapshot and mean rank across all official resolution updates within each round through that date. The reported 947 resolved questions count the latest resolution sets as of September 16, 2026. Question composition differs across rounds, so within-round comparisons are primary. App. M details mean rank.

FutureX. FutureX[4] gathers unresolved real-world events, records predictions before their outcomes, and evaluates them after resolution. We report its Level-2 multiple-choice task, where a prediction is a set of selected options. For predicted and correct sets $\widehat{Y}$ and Y, respectively,

$$F_1 = \frac{2|\widehat{Y} \cap Y|}{|\widehat{Y}| + |Y|}.$$

We retain the benchmark's empty-set handling and multiply F_1 by 100 for reporting. This evaluates answer-set overlap, whereas Brier scoring evaluates probability forecasts; the two benchmarks' scores are not directly comparable.

Tab. 4 covers the June 17, June 24, and July 1, 2026 rounds and the 27 participants present in all three snapshots. The three rounds contain 23, 31, and 19 Level-2 questions, respectively (73 in total).

The main table reports mean F_1 and mean rank, the mean of each family's three per-round ranks within the same 27-family panel (ties share a rank, with subsequent ranks skipped).

Rows remain ordered by mean F_1; mean rank is distinct from rank by mean F_1. Tab. 10 reports all per-round scores and the mean-score ranking.

K.1 LIVING CONFIGURATION

For each target, the Data Agent selects $M = 12$ historical proxy tasks from $K = 100$ retrieved candidates: eight for research and four for held-out selection. Unless stated otherwise, we use Gemini-3.5-Flash, eight RSI iterations, up to 20 web searches per strategy per question, and a shortlist of three strategies.

L FORECASTBENCH HUMAN BASELINES AND DIFFICULTY ADJUSTMENT

Human baselines. All human results in Tab. 11(a) are provided by ForecastBench. ForecastBench recruited 39 superforecasters and 500 public forecasters to predict a 200-question human set, and takes each group's per-question median as its forecast; humans did not receive market prices. Because submitted systems and humans answer questions from different rounds, ForecastBench compares them with the difficulty-adjusted Brier score and reports confidence intervals on its leaderboard. Tab. 11(a) follows this protocol directly.

[3] https://www.forecastbench.org/

[4] https://futurex.live/

Table 10: FutureX Level-2 living results: set-F_1 ↑ across three weekly rounds for 27 organizations. **Bold** and underlining mark the best and second-best values in each column, respectively. Blue: external submissions, including our **RSI-FORECASTER**; black: organizer-hosted baselines.

SUBMISSIONS	06-17	06-24	07-01	Mean F_1
RSI-FORECASTER	59.20	58.55	77.15	**64.97**
React (RUC)	63.30	54.35	75.30	64.32
Doubao2.0 Pro	53.45	57.83	78.11	63.13
HAgent	**63.79**	50.72	74.86	63.12
GPT5.4 High	56.90	54.35	75.90	62.38
DeepSeekV4 Pro	53.45	54.78	75.44	61.22
Branchus	56.90	50.72	75.41	61.01
DragonAgent	57.47	50.00	75.35	60.94
openclaw	46.55	**58.70**	76.88	60.71
Apodex-FutureFlow	53.45	**58.70**	68.80	60.32
DeepSeekV3.2 Thinking	51.72	54.35	74.55	60.21
Qwen3.7 Max	51.72	54.35	73.73	59.93
H2O AI Super Agent	53.45	50.43	75.57	59.82
Gemini3.1 Pro High	50.00	53.48	75.20	59.56
GLM5.1	58.62	46.96	73.04	59.54
JSearch_V1	56.90	44.93	75.19	59.01
hermes	58.62	44.78	71.61	58.34
Grok4.2 Reasoning	46.55	49.86	**78.51**	58.31
MAS	56.90	46.38	69.79	57.69
deepagents	50.00	50.43	71.72	57.38
Gemini3.5 (**Our Backbone**)	56.90	45.65	66.85	56.47
Qwen3.6 Max	51.72	39.71	74.70	55.38
Doubao2.0 Lite	41.38	53.04	70.20	54.87
MiniMax-M2.7	46.55	45.65	65.54	52.58
DeepSeekV4 Flash	53.45	34.78	60.75	49.66
MiniMax-M3	36.21	33.33	66.77	45.44
Grok4	20.69	34.78	12.90	22.79

Difficulty-adjusted Brier score. ForecastBench decomposes the Brier score of forecaster i on question j as

$$b_{ij} = \alpha_i + \gamma_j + \varepsilon_{ij},$$

where α_i is forecaster ability and γ_j is question difficulty. Given estimates $\hat{\gamma}_j$, the adjusted score is $b_{ij} - \hat{\gamma}_j$, shifted so that the constant 0.5 forecast keeps a score of 0.25. For dataset questions, $\hat{\gamma}_j$ is estimated by least squares from all participating models; for market questions, $\hat{\gamma}_j$ is the Brier score of the market price itself, so a forecaster ranks above the market only by outperforming it on average across the scored questions. Overall averages the adjusted Brier scores of the two question types with equal weight before converting to BI; because the conversion is nonlinear, Overall differs from the mean of the two BI columns.

Question-level comparison with Market Crowd Prediction. Tab. 11(b) compares forecasts on 213 resolved market questions across three ForecastBench rounds, using platform-provided crowd probabilities supplied by ForecastBench. A win means a lower question-level Brier score than the market crowd prediction. RSI-FORECASTER wins on 154 questions, ties on 8, and loses on 51; the win rate includes ties in its denominator: $154/213 = 72.3\%$. Tab. 12 reports the source breakdown; RSI-FORECASTER wins more than two thirds of questions on every source.

M MEAN RANK ON A CONTINUOUSLY UPDATING BENCHMARK

ForecastBench is a living benchmark. Different from FutureX with all questions expected to be resolved in seven days, questions keep resolving after the submission deadline, up to several years. Every official resolution update re-scores all entries against a larger answer set. A leaderboard position is therefore not a fixed outcome but a trajectory, and any single snapshot is one draw from it whose value depends on the date the manuscript happened to be compiled. Reporting a single snapshot would make the reported standing an artefact of that date rather than a property of the

```
You are an expert superforecaster, familiar with the work of Tetlock and
others.  Make a prediction of the probability that the question will be
resolved as true.  You MUST give a probability estimate between 0 and 1
UNDER ALL CIRCUMSTANCES. If for some reason you can't answer, pick the
base rate, but return a number between 0 and 1.

Question:
{question}

Question Background:
{background}

Market value on:
{value}

Output your answer (a number between 0 and 1) with an asterisk at the
beginning and end of the decimal.
Do not output anything else.
Answer:  {{ Insert answer here }}
```

Figure 5: Prompt for Gemini-3.5-Flash with the crowd prediction as input.

Table 11: Detailed statistics for the human comparisons in Fig. 1(c) and Fig. 4. Human comparisons follow ForecastBench's official difficulty-adjusted results; crowd comparisons use paired forecasts from our three participating rounds. Parentheses give 95% confidence intervals; N counts resolved forecasts, not human participants. Overall BI converts the equal-weight mean of the two adjusted Brier scores, rather than averaging the two BI values.

(a) Human forecasters

	Dataset		Market		Overall
Forecaster	BI ↑ (95% CI)	N	BI ↑ (95% CI)	N	BI ↑
RSI-FORECASTER	65.9 (64.3, 67.7)	241	73.8 (72.0, 76.0)	97	69.6
Human Superforecaster	63.6 (62.4, 65.0)	521	75.0 (72.0, 78.7)	57	68.8
Human Public	59.2 (57.6, 60.7)	521	66.3 (61.1, 72.1)	57	62.6

(b) Market Crowd Prediction

Paired market comparison ($N = 213$)	BI (95% CI)
RSI-FORECASTER	72.42 (68.49, 76.47)
Gemini-3.5 + Crowd	69.12 (65.23, 73.05)
Market Crowd Prediction	68.57 (64.97, 71.91)

system. The September 16 snapshot reports performance on the outcomes available by that date, while mean rank summarizes standings across official resolution updates within each round.

N RSI ENVIRONMENT EVALUATION

Our RSI environment controls leakage while preserving sufficient forecast-time evidence for high-fidelity backtesting (Sec. 4.2). We evaluate leakage detection, webpage processing, and the reproduction of living forecasting performance.

API responses are discarded if their versions cannot be verified as available by the forecast cutoff.

Evaluation data. We construct the evaluation set from the June 3 FutureX Level-2 tasks. Search snippets and webpage content saved before each question's forecast cutoff serve as clean examples. After resolution, we retrieve content that explicitly reveals the corresponding outcomes as leaking examples. The snippet evaluation contains 153 leaking and 534 clean examples; the webpage evaluation contains 26 leaking and 150 clean pages.

Leakage detection in search snippets. We compare deterministic rules, an LLM content judge, and our combination of both. Recall (↑) is the fraction of leaking snippets detected; the false positive

Table 12: Question-level wins against Market Crowd Prediction, by source. Win rates include ties in the denominator.

	MANIFOLD	METACULUS	POLYMARKET	KALSHI	ALL
Questions	42	59	96	16	213
Wins	33	40	69	12	154
Win rate	78.6%	67.8%	71.9%	75.0%	72.3%

Table 13: Detection of outcome-revealing snippets. Higher recall and lower FPR are better.

Guard	Recall ↑	FPR ↓
Deterministic rules	54.2%	0.0%
LLM content judge	69.3%	8.4%
Rules + content judge (ours)	**95.4%**	8.4%

Table 14: FutureX Level-2 performance under different backtesting environments. Differences are backtesting scores minus the living score; smaller absolute differences indicate closer reproduction of living performance.

Environment	Mean set-F_1	Difference
Living forecasts	56.2	—
No leakage control	83.4	+27.2
Archived backtesting	45.0	−11.2
Our backtesting environment	**54.1**	**−2.1**

rate (FPR, ↓) is the fraction of clean snippets incorrectly filtered out. As shown in Tab. 13, our combination reaches 95.4% recall, compared with 54.2% for rules alone and 69.3% for the LLM judge, while matching the LLM judge's 8.4% FPR.

Leakage removal from webpages. Our method inspects page content and removes leaking spans to retain the remaining admissible content. We compare this approach with whole-page filtering and check whether annotated leakage remains in the processed text.

The leakage removal rate (↑) is the fraction of leaking pages with no annotated leakage remaining after processing. The clean-page discard rate (↓) is the fraction of clean pages discarded entirely.

Our method removes leakage from 25/26 pages (96.2%), compared with 22/26 (84.6%) for whole-page filtering. Of the 150 clean pages, the two approaches discard 3 (2.0%) and 8 (5.3%) entire pages, respectively. Our method achieves a higher leakage removal rate and a lower clean-page discard rate.

Reproducing living forecasting performance. To assess how closely backtesting reproduces living performance, we replay 15 forecasting strategies randomly proposed by an LLM on the June 3 FutureX Level-2 tasks, with three repeats per strategy. We compare their mean set-F_1 with the corresponding living scores. A smaller gap indicates closer reproduction of mean living performance.

As shown in Tab. 14, the mean living score is 56.2. Backtesting without leakage control scores 83.4, while archived backtesting using the Wayback Machine scores 45.0. Our environment scores 54.1, only 2.1 points below the living mean, with the smallest mean-score gap among the compared environments. This result supports our environment's balance between leakage control and information sufficiency. Environment ablations also show higher Target BI with our environment (Tab. 1; Fig. 3).