Beyond Forecasting: The Belief-to-Trade Layer in Prediction-Market Agents

Yishu Wang [1] **Yuxuan Wang** [2] **Jiaqi Deng** [3] **Hanyang Tang** [4]

Abstract

Forecasting future events has attracted growing attention as a testbed for general-purpose AI. A natural way to ground this evaluation is let the models trade in the prediction markets. Trading, however, requires more than forecasting. Moreover, recent benchmarks report a substantial gap between calibrated probability scores and the trading results. We propose Raven-Agent, to the best of our knowledge, the first autonomous trading agent for prediction markets. On a controlled replay over an archived decision set, our architecture achieves the only positive return and the only positive risk-adjusted return among all tested policies. We have released our code in https://github.com/Alchemist-X/predict-raven.

1. Introduction

Forecasting future events has attracted growing attention as a testbed for general-purpose AI (Karger et al., 2024; Yang et al., 2025). The task demands retrieval, temporal reasoning, and decision under uncertainty. It is naturally studied as a joint problem of probability estimation and action under that estimate.

To make progress on this problem, one line of exisiting works focuses on probability quality and improves LLM forecasters on resolved events. The Belief-Logit-Forecaster (BLF) of Murphy (2026) uses structured belief states and multi-trial aggregation, currently leading ForecastBench (Karger et al., 2024). Yet such systems improve calibration without acting on their forecasts. On the other hand, prediction market benchmarks score systems by realized trading return as well as by probability (Yang et al., 2025; Cheng et al., 2026; Zhang et al., 2026). They consistently report that strong probability scores do not translate into profitable trades. However, they treat the trading rule as a fixed protocol such as a per market bet or a confidence threshold, not as a designable object. Moreover, position sizing, portfolio exposure, and risk control are either absent or expressed as a prompt instruction in these papers. We refer to these downstream components as the *trading layer*, and we make it the object of study in this paper.

Across both lines, the trading layer is either omitted or hard-coded, never treated as a designable object. A useful trading layer should be explicit and modular, with selection, sizing, and risk control as separate components. It should be deterministic, so that risk constraints remain in effect even when the model is confidently wrong. It should also be composable with any forecaster, so that improvements in calibration and in trade execution can be evaluated independently.

In this paper, we propose Raven-Agent, an autonomous prediction market agent built around an explicit, deterministic, composable trading layer. Within the layer, a ranking module addresses selection by scoring each candidate by a time normalized return rate. The same module addresses sizing with a fractional Kelly bet on the retained positions. An execution and risk module addresses risk control by enforcing deterministic constraints on stake, exposure, stop loss, and drawdown outside the language model prompt. We also include an information module for evidence retrieval and a forecasting module that produces per contract probabilities. We validate Raven-Agent on a controlled replay protocol. The results show our method improves return on stake and risk adjusted return over other heuristic policies.

Our contributions can be summarized in three-folds:

- We propose a replay protocol that reuses archived forecasts to isolate the trading layer from the forecaster.
- We propose Raven-Agent, an autonomous agent that makes the trading layer an explicit module composable with a swappable forecaster.
- On a controlled replay, the trading layer improves return on stake and risk adjusted return over other policies.

[1]Hong Kong University of Science and Technology [2]Peking University [3]The University of Hong Kong [4]Massachusetts Institute of Technology.
Correspondence to: Yishu Wang <issue.00.gui@gmail.com>, Yuxuan Wang <yuxwang@pku.edu.cn>, Jiaqi Deng <jiaqideng@connect.hku.hk>, Hanyang Tang <hytangs@mit.edu>.

ICML 2026 Workshop on AI Forecasting, Seoul, South Korea, 2026.

2. Problem Setup

2.1. Prediction Markets

We consider binary prediction markets in which the selected contract pays one dollar if it resolves YES and zero otherwise. For each market i, we observe the realized outcome of the chosen side, written $z_i \in \{0, 1\}$. The entry price is $q_i \in (0, 1)$, and we let $p_i \in (0, 1)$ denote the model probability that the contract pays. A stake of s_i dollars buys s_i/q_i shares. The realized profit at resolution is

$$\Pi_i = s_i \left(\frac{z_i}{q_i} - 1 \right). \tag{1}$$

For markets unresolved at the evaluation date, we substitute the mark to market price m_i for z_i in Equation (1). Resolved and unresolved positions are reported separately throughout.

To compare candidates whose payoff dates differ widely, we need a return rate, not just an edge. The per dollar expected return of a long position is $p_i/q_i - 1$ before fees and slippage. The raw probability edge $e_i = p_i - q_i$ does not by itself reflect that capital is locked until resolution. A small edge that resolves quickly can dominate a larger edge that ties capital for months. With T_i days to resolution, we therefore rank candidates by a time normalized score

$$\rho_i = \left(\frac{p_i}{q_i} - 1 \right) \cdot \frac{30}{\max(T_i, 1)}. \tag{2}$$

This is the per dollar expected return of a candidate trade in units of one month. It is a ranking device, not a claim of exact monthly compounding.

2.2. Trading policies

A trading policy formalizes which action follows from a forecast. We write it as$\pi : (p_i, q_i, T_i, \ell_i, W, \mathcal{S}) \mapsto (a_i, s_i)$, where ℓ_i is liquidity, W is bankroll, and $\mathcal{S}$ is the current portfolio state. The action a_i is either open or skip, and $s_i \geq 0$ is the stake. Two agents with the same forecasts can produce very different outcomes if they differ in their policy. We hold all positions to resolution in the main replay and defer dynamic exit to future work.

3. Methodology

In this section, we instantiate the trading agent Raven-Agent. The full loop is summarized in Algorithm 1, and the four modules are described below.

3.1. Information collection

The information module scans candidate markets and writes a timestamped pulse. We periodically query the Polymarket Gamma API for each candidate, retrieving its price, liquidity, fee schedule, and resolution date. A liquidity filter drops markets below a threshold, and an evidence retriever pulls recent news for the survivors. This follows the spirit of retrieval augmented forecasting (Murphy, 2026). The pulse serves as a contract between data sources and downstream modules. New evidence feeds can therefore be added without changing the trading interface.

Algorithm 1 Raven-Agent forecast to trade loop.

Require: bankroll W and current portfolio state
1: Scan candidate markets and retrieve evidence
2: Estimate p_i for each candidate i from the candidate lists.
3: Compute ρ_i via Equation (2); drop candidates below a threshold and rank the rest by ρ_i.
4: Size each surviving candidate by a quarter Kelly fraction, clipped by liquidity and a per round batch cap.
5: Reject any proposal violating exposure, stop loss, or drawdown constraints.
6: Submit the remaining orders and review the portfolio state, checking if we need adjust any positions.
7: Repeat as new candidate markets and evidence arrive.

3.2. Probability analysis

The forecasting module emits the per contract probability p_i for the side the agent prefers to hold. It is a swappable layer, and we currently support two runtimes. One parses the pulse markdown directly and combines model side estimates with market implied priors. The other dispatches the pulse to an external language model through a command line bridge. Both record, alongside p_i, the chosen side, a confidence level, and a short thesis for later audit. Forecaster centric systems like BLF (Murphy, 2026) improve the probability itself via belief states and multi trial aggregation. Our contribution operates downstream of the forecaster, so any such forecaster can be substituted without changing the trading layer. In the present replay we hold this module fixed and reuse archived probabilities, so any difference in outcome is attributable to the trading layer.

3.3. Ranking and sizing

The ranking module both selects candidates and sets their stakes. For selection, the module ranks candidates by ρ_i from Equation (2), drops those below a threshold, and keeps the top $K = 4$. Ranking by ρ_i rather than by the raw edge $e_i = p_i - q_i$ accounts for how long capital is locked. This favors short resolution markets when edges are similar.

For sizing, the module applies one quarter of the Kelly bet to each retained candidate:

$$s_i = \frac{1}{4} \cdot \max\left(0, \frac{p_i - q_i}{1 - q_i}\right) \cdot W, \tag{3}$$

clipped by available liquidity and a per round batch cap. The Kelly criterion gives the bet size that maximizes long run log wealth under a known win probability (Kelly, 1956). We use one quarter rather than full Kelly because p_i is itself an estimate. Scaling down the Kelly bet is the standard practitioner correction for input noise.

3.4. Execution and risk

The execution and risk module enforces deterministic risk constraints on every order before submission. We implement it as a set of services that check each proposed order against five hard constraints. A per trade notional cap and an aggregate exposure cap limit single stake and total open notional respectively. A per event cap limits capital across correlated contracts of the same event. A position level stop loss closes any position whose mark to market loss exceeds a fixed fraction of its cost basis. A portfolio level drawdown halt freezes new openings when the equity drawdown $(W_{\max} - W)/W_{\max}$ exceeds a threshold. The module also rejects orders against stale pulses or contracts outside the allowlist, and routes surviving orders as fill or kill market orders. Prior work shows that prompt-level risk guidance is unreliable under end-to-end LLM control (Zhang et al., 2026), and that optimizing prediction accuracy can reduce trading return when the objective is misaligned (Jang et al., 2025). Keeping these checks outside the prompt avoids that failure mode and produces auditable rejection reasons that a learned policy cannot easily match.

4. Experiments

4.1. Setup

Environment. We evaluate on a counterfactual replay of archived live decisions from a Polymarket deployment. The archive records 59 open decisions across 65 pulse runs; 44 reach the executable stage after filtering by liquidity, resolution clarity, and event overlap. For each candidate, Raven-Agent estimated the payout probability, compared it with the market price, computed the edge, and proposed a stake; live trades were submitted through Polymarket's CLOB API. The replay reuses the archived forecasts, prices, and execution records from these runs (concrete examples in Section C).

All policies face the same archived markets, timestamps, prices, and probability estimates; the only difference is the rule used to decide which positive-edge forecasts become trades and how much capital they receive. Positions are held to resolution where available, and otherwise marked to market at a fixed evaluation date.

Policies. We compare five policies. **i)** *Forecast-only* trades every positive-edge forecast at a fixed \$10 stake. **ii)** *Edge-proportional* trades the same set but sizes each stake proportionally to the probability edge $e_i = p_i - q_i$, with total stake normalized to match Forecast-only (\$580) so that ROI differences reflect allocation, not capital deployed. **iii)** *Edge-filter* keeps only forecasts whose edge exceeds five percentage points, at a fixed \$10 stake. **iv)** *Raven-Agent (fixed)* applies the full execution and risk layer of Raven-Agent at a fixed \$10 stake per surviving trade. It rejects forecasts when market data is stale, model confidence is low, liquidity is insufficient, or event exposure is excessive. **v)** *Raven-Agent (full)* uses the same execution layer with Raven's deployed quarter-Kelly sizing, so capital scales with edge, time to resolution, liquidity, and portfolio risk.

Metrics. We report three metrics. Probability quality is measured by the Brier score $\mathrm{Brier} = \frac{1}{M}\sum_i (p_i - z_i)^2$ on resolved rows (Brier, 1950). Trading quality is measured by return on stake $\mathrm{ROI} = \sum_i \Pi_i / \sum_i s_i$. Risk adjusted return is measured by a stake weighted Sharpe ratio $\mathrm{Sharpe}_w = \bar{r}_w / \mathrm{std}_w(r_i)$, where r_i is the realized or mark to market trade return and $w_i = s_i$. We abbreviate Sharpe_w as S_w and report the aggregate $\sum_i \Pi_i$ as profit and loss (PnL) in the tables.

4.2. Result Analysis

Raven-Agent (full) is the only policy with positive ROI and the only one with positive S_w (Table 1). On Polymarket, Raven-Agent has also accumulated \$53.92 across 20 predictions in live deployment (Figures 1 and 2).

The Edge-proportional policy shows that sizing alone, without selection, is harmful: concentrating capital on confidently wrong high-edge predictions amplifies losses to -55.5% ROI, far worse than flat sizing (-10.7%). Raven-Agent (full) (+15.9%) avoids this by first removing value-destroying forecasts, suggesting that selection and risk filtering are important complements to informed sizing. Among the fixed-stake policies, each additional trading-layer component improves ROI, and sizing then turns the same selected set from a small loss into a substantial gain.

The Selected Brier decreases along the policy sequence because more aggressive filtering retains better-calibrated rows, not because the forecaster improves (the full-archive Brier is 0.329 for every policy). Bootstrap confidence intervals and leave-one-out sensitivity are reported in Section D; an expanded 132-row replay on a fused archive is in Section E.

5. Conclusion and discussion

5.1. Conclusion

We studied the gap between forecasting and trading on live prediction markets. We presented Raven-Agent, a system

Table 1. Trading layer ablation on the main archive. Edge-proportional total stake is normalized to $580 to match Forecast-only.

Policy	ROI	Selected Brier†	S_w	# Exec.	PnL	Stake
Forecast-only	−10.7%	0.329	−0.20	58	−$62	$580
Edge-proportional	−55.5%	0.329	−0.91	58	−$322	$580
Edge-filter	−9.3%	0.302	−0.16	48	−$45	$480
Raven-Agent (fixed)	−4.7%	**0.258**	−0.09	42	−$20	$420
Raven-Agent (full)	**+15.9%**	**0.258**	**+0.42**	42	+$131	$820

† Brier score computed on each policy's traded subset of resolved rows. All policies use the same archived probabilities; the full-archive Brier is 0.329 for every policy. See §4.2 for discussion.

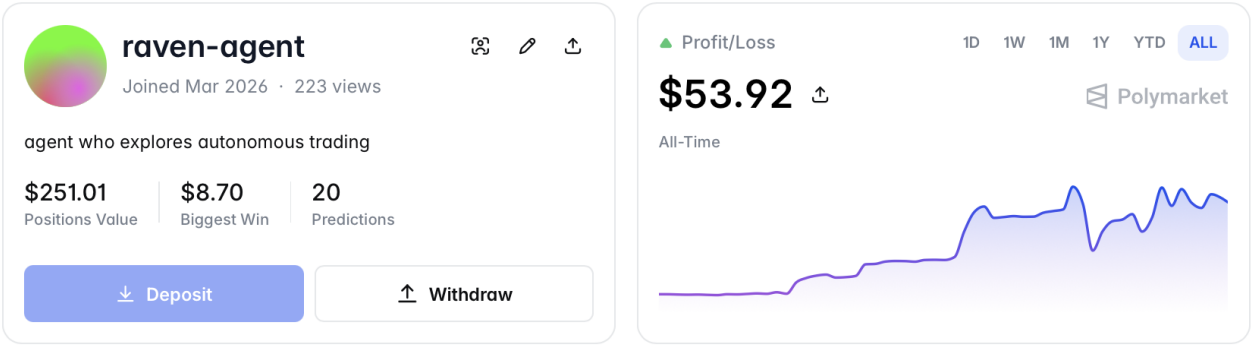
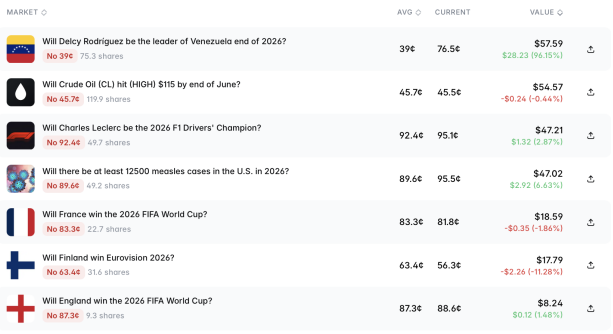
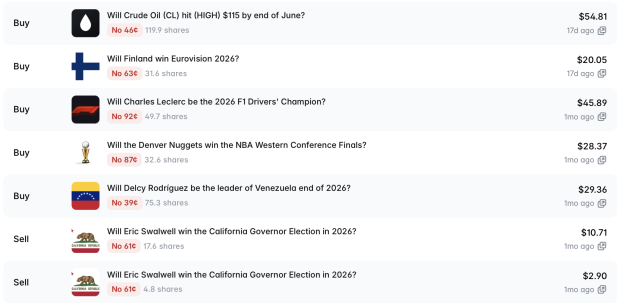


Figure 1. Raven-Agent's live Polymarket profile and all-time profit and loss curve. Across 20 predictions, the agent currently holds $251.01 in open positions and has accumulated $53.92 in cumulative profit.



(a) Current open positions: seven NO-side contracts spanning politics, commodities, sports, entertainment, and public health.



(b) Recent trade history: a mix of NO-side entries (crude oil, Eurovision, F1, NBA, Venezuela).

Figure 2. Polymarket dashboard snapshots of Raven-Agent's open positions and recent trade history during live deployment.

whose trading layer handles selection, sizing, and risk control on top of a swappable forecaster. On a controlled replay over a fixed forecast archive, Raven-Agent achieved the only positive return and risk-adjusted return among five policies. The edge-proportional baseline, which sizes by edge without selection, performs substantially worse than flat sizing (−55.5% vs. −10.7%), suggesting that selection and risk filtering are important complements to informed sizing.

5.2. Discussion

The current results are preliminary. The sample is modest and the forecaster is held fixed, which isolates the trading layer but leaves the forecaster contribution unmeasured. A stronger forecaster can be plugged into the same replay infrastructure, and larger archives would strengthen the statistical claims. Despite these caveats, the replay suggests that explicit selection, sizing, and risk controls materially improve observed outcomes even with the forecaster held constant.

References

Brier, G. W. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950. doi: 10.1175/1520-0493(1950)078⟨0001:VOFEIT⟩2.0.CO;2.

Cheng, P., Liu, J., and Long, Y. Polybench: Benchmarking LLM forecasting and trading capabilities on live prediction market data. *arXiv preprint arXiv:2604.14199*, 2026. URL https://arxiv.org/abs/2604.14199.

Jang, Y., Kim, J., and Zhang, B.-T. The losing winner: An LLM agent that predicts the market but loses money. *NeurIPS 2025 Workshop on Generative AI in Finance*, 2025. URL https://nips.cc/virtual/2025/loc/san-diego/132549.

Karger, E., Bastani, H., Chen, Y.-H., Jacobs, Z., Halawi, D., Zhang, F., and Tetlock, P. E. Forecastbench: A dynamic benchmark of ai forecasting capabilities. *arXiv preprint arXiv:2409.19839*, 2024. URL https://arxiv.org/abs/2409.19839.

Kelly, J. L. A new interpretation of information rate. *The Bell System Technical Journal*, 35(4):917–926, 1956. doi: 10.1002/j.1538-7305.1956.tb03809.x.

Murphy, K. Agentic forecasting using sequential bayesian updating of linguistic beliefs. *arXiv preprint arXiv:2604.18576*, 2026. URL https://arxiv.org/abs/2604.18576.

Yang, Q., Mahns, S., Li, S., Gu, A., Wu, J., and Xu, H. LLM-as-a-prophet: Understanding predictive intelligence with prophet arena. *arXiv preprint arXiv:2510.17638*, 2025. URL https://arxiv.org/abs/2510.17638.

Zeng, Z. et al. Futurex: An advanced live benchmark for LLM agents in future prediction. *arXiv preprint arXiv:2508.11987*, 2025. URL https://arxiv.org/abs/2508.11987.

Zhang, J., Liu, G., Johansson, O., Yitayew, H., Ohly, K., and Li, G. Prediction arena: Benchmarking AI models on real-world prediction markets. *arXiv preprint arXiv:2604.07355*, 2026. URL https://arxiv.org/abs/2604.07355.

A. Live deployment and code

The Raven-Agent deployment discussed in this paper was run on a live Polymarket account. For transparency, the public Polymarket profile is available at: https://polymarket.com/profile/0x6664e32f79aee42639f73633e40b5a842b07614e. The accompanying code and redacted experiment artifacts are available at: https://github.com/Alchemist-X/predict-raven. The repository contains the agent implementation, replay scripts, and derived tables used in the experiments, with private credentials and non-public runtime paths removed.

B. Related work

Forecasting benchmarks. Evaluation of LLM forecasting has moved from static question answering toward live prediction market settings (Karger et al., 2024; Yang et al., 2025; Zeng et al., 2025). Prophet Arena (Yang et al., 2025) reports Brier score, calibration, and market return as three complementary axes. It shows that frontier language models match a market consensus policy on probability quality but do not break even on return. PolyBench (Cheng et al., 2026) couples market snapshots with order book state. It finds that most evaluated models incur losses under realistic execution, even when their stated probabilities are confident. Prediction Arena (Zhang et al., 2026) reports the same pattern under live trading. Our work builds on this evaluation infrastructure, but shifts the unit of analysis from the forecaster to the trading policy. We hold archived forecasts fixed and vary the surrounding decision and risk modules.

Forecasting agents. A complementary line of work pushes the forecasting layer itself. BLF (Murphy, 2026) uses structured linguistic belief states, multi trial aggregation, and hierarchical calibration to substantially improve Brier scores on Karger et al. (2024). These methods are orthogonal to our work. A stronger forecaster can be inserted into the probability analysis module of Raven-Agent without changing the rest of the system, and the benefits of the two lines compound. We do not optimize the forecaster in this paper. We study what should surround it before its outputs control capital.

Risk aware decision making. End to end risk aware policy optimization is widely studied in financial reinforcement learning. Exploration there is sample intensive and tolerates reversible mistakes. On live prediction markets, every exploratory action consumes real bankroll on a market that resolves within days, which makes large scale policy training infeasible. We therefore implement risk constraints as deterministic services outside the language model prompt. This bounds the worst case loss per decision and produces auditable rejection reasons that learned policies cannot easily match. Recent concurrent work provides direct empirical evidence for this design choice. Zhang et al. (2026) deployed six frontier language models as end-to-end prediction market agents on Kalshi, each trading with a $10,000 bankroll over 57 days. All six lost money, with returns ranging from -16% to -31%. The only risk controls that held were hard-coded constraints (a 15% per-market concentration cap and solvency checks); prompt-level risk guidance was frequently ignored. Jang et al. (2025) show that even with a deterministic execution rule, optimizing a proxy forecasting objective (classification accuracy via RLVR) can improve accuracy while worsening trading return, with their best-accuracy agent achieving -14.8% return. Together, these findings motivate separating the trading layer from the language model and enforcing risk constraints outside the prompt.

Comparison with Prediction Arena. Prediction Arena (Zhang et al., 2026) and Raven-Agent differ in a key architectural choice. Prediction Arena gives each language model end-to-end control: the model receives market data and portfolio state, reasons freely with tool access, and directly issues trade orders with no external sizing or risk module. Raven-Agent separates the forecasting module from the trading layer, so selection, sizing, and risk enforcement are deterministic components that the language model cannot override. The two setups are therefore not directly comparable on the same leaderboard: Prediction Arena evaluates the LLM as a complete trading agent, while our replay protocol evaluates the trading layer with the forecaster held fixed. A controlled comparison would require running Prediction Arena agents on our archived decision set, substituting their end-to-end decisions for our trading layer while keeping the same market snapshots and price data. We leave this to future work.

C. Archive coverage

The replay archive records the intermediate outputs produced by Raven-Agent during deployment. It starts from raw market scans, then applies filtering, candidate persistence, recommendation generation, and final trading decisions. The table below

reports how many artifacts are available at each stage of this pipeline. These counts are intended to make the replay scope clear: many markets are scanned, a smaller set is kept as candidates, and only a subset becomes executable trading decisions.

Replay procedure. In each archived run, the agent first queried Polymarket markets and filtered them by liquidity, resolution clarity, usable prices, and event overlap. The candidate set included markets such as crude-oil threshold contracts, political resignation markets, sports futures, and other high-liquidity event markets. For example, one archived run scanned 1,544 Polymarket markets, retained 92 after filtering, and selected 12 candidates for deeper analysis. The final recommendations in that run focused on the No side of crude oil reaching $100 by the end of March, Bitcoin dipping to $65,000 in March, and Netanyahu leaving office by June 30.

For each retained candidate, Raven-Agent estimated the probability that the selected contract would pay out. It compared this probability with the Polymarket price, computed the edge, ranked opportunities, and proposed a stake. When the live system decided to trade, it submitted real orders through Polymarket's CLOB API rather than simulating execution inside the model. The replay uses the archived forecasts, prices, order metadata, and execution records from these live runs.

Table 2. Archive coverage used by the replay experiments. Counts are grouped by the replay pipeline stage.

Stage	Quantity	Count
Raw scan	Pulse JSON files	65
Raw scan	Fetched market observations	264,859
Filtering	Filtered market observations	25,231
Candidate set	Persisted candidate rows	844
Candidate set	Unique persisted candidate markets	222
Recommendation	Recommendation files	45
Decision	Decision rows	250
Decision	Open-decision rows	59
Decision	Executable-plan rows	44
Execution	Matched observed fills	18
Expansion	Earlier selected Full Raven rows	90
Expansion	Combined priced rows	132

D. Robustness analysis

Bootstrap confidence intervals. We resample trades with replacement 10,000 times and report 95% percentile intervals for ROI and S_w (Table 3). Raven-Agent (full) is the only policy whose ROI confidence interval lies entirely above zero. Edge-proportional's interval lies entirely below zero, confirming that the negative result is robust and not an artifact of a few extreme trades.

Table 3. Bootstrap 95% confidence intervals (10,000 resamples).

Policy	ROI 95% CI	S_w 95% CI
Forecast-only	[−24.7%, +3.2%]	[−0.46, +0.06]
Edge-proportional	[−71.8%, −29.2%]	[−1.42, −0.44]
Edge-filter	[−25.1%, +6.8%]	[−0.46, +0.12]
Raven (fixed)	[−20.6%, +11.5%]	[−0.41, +0.22]
Raven (full)	[+3.8%, +30.0%]	[+0.11, +0.76]

Leave-one-out sensitivity. We compute ROI after dropping each trade in turn (Table 4). Raven-Agent (full) ROI stays positive across all deletions, ranging from +12.7% to +19.0%, indicating that no single trade drives the result.

E. Expanded replay on a fused archive

To probe robustness on a larger sample, we extend the replay to an earlier Raven-Agent deployment that produced 90 additional selected recommendations. The combined 132-row set yields a positive ROI of +5.9% under fixed $10 stakes (Table 5), confirming the main-archive direction on a larger sample. Bootstrap 95% CI for the fused set is [−4.5%, +16.1%] for ROI and [−0.07, +0.27] for S_w. The earlier archive contains only selected trades (the rejected diagnostic cannot be

Table 4. Leave-one-out ROI sensitivity.

Policy	ROI range	Std
Forecast-only	[−13.6%, −9.1%]	1.0%
Edge-proportional	[−58.0%, −51.6%]	1.5%
Edge-filter	[−12.8%, −7.4%]	1.2%
Raven (fixed)	[−8.5%, −2.4%]	1.3%
Raven (full)	[+12.7%, +19.0%]	1.0%

extended), and the archived sizing column is omitted because the earlier deployment used a much larger bankroll (∼$100k vs. ∼$1k), making absolute stake and PnL incomparable.

Table 5. Fused Raven-Agent selected-trade replay (fixed $10 stake).

Dataset	Priced	Stake	PnL	ROI	S_w	Brier
Main archive	42	$420	−$20	−4.7%	−0.09	0.258
Earlier archive	90	$900	+$98	+10.9%	+0.17	0.188
Combined	132	$1,320	+$78	+5.9%	+0.10	0.204

F. Example archived agent traces

Raven-Agent stores intermediate artifacts during deployment, not only final trades. These artifacts are useful for auditing the belief-to-trade pipeline because they record which markets were scanned, which candidates were filtered, what probability estimates were produced, and why a selected market was considered actionable. Figure 3 shows an example daily market pulse after translation and condensation. The trace begins with a broad market scan, narrows the pool through filtering and candidate selection, and records the rationale for the final Top 3 recommendations. Figure 4 shows the corresponding probability reasoning trace for one selected market. It preserves the market-implied probability, the agent probability, evidence adjustments, settlement-rule checks, confidence notes, and the resulting trading proposal.

These figures are not additional quantitative results. They are included to make the archived decision process concrete and to show that the replay is based on structured records left by the deployed agent rather than on reconstructed explanations after the fact. Private credentials and non-public runtime paths are omitted.

RAVEN-AGENT ARCHIVE ARTIFACT · FIGURE 1

Polymarket Daily Market Pulse

Translated archive excerpt showing market screening, candidate filtering, and final recommendation rationale from one Raven-Agent deployment run.

Generated	**2026-03-20 07:00:42 UTC**
Fetched	**2026-03-20 07:00:48 UTC**
Market data	**Polymarket Gamma API**
Order book	**Polymarket CLOB API**

> The clearest opportunities in this round concentrated on the No side of markets where dramatic events appeared overestimated: crude oil reaching an extreme month-end price, Bitcoin making a sharp short-term decline, and Netanyahu leaving office within a fixed window. External news evidence in the research context was limited, so the report gives explicit recommendations while lowering overall confidence and recording evidence gaps directly in the conclusion.

Candidate Pool and Screening Approach

The candidates came from markets already pre-screened and fetched in the research-context JSON. The base scope contained **1,544 fetched markets**, **92 markets after filtering**, and **12 selected candidates**. Among candidates with additional rules, comments, and order-book samples, the priority research directions included:

1. Whether Crude Oil (CL) would hit $100 / $105 / $110 / $120 by the end of March.
2. Whether Bitcoin would dip to $65,000 / $60,000 / $55,000 in March, or reach $80,000.
3. Whether Netanyahu would leave office by April 30 or June 30.
4. Long-tail NCAA championship outcomes, including Duke and Iowa State.

Screening Principles

- Prefer candidates with a complete event-level URL, basic settlement rules, comment samples, and order-book samples.
- Prefer markets with sufficient liquidity, active 24h volume, and prices that visibly exhibit longshot-bias-style features.
- Deduplicate highly overlapping threshold markets within the same event chain, keeping the level with the clearest auditable edge and explanation.
- Allow markets with incomplete rules, weak external evidence, or low-information comments to remain only if confidence is reduced and the missing-data issue is written explicitly.

Representative Candidates Not in the Top 3

- **Bitcoin to $55,000:** the No side still looked intuitively favorable, but the market already priced No at 93.7%, leaving limited additional edge.
- **Crude oil to $120:** similar longshot-No logic, but the odds were more extreme and the trade was more tail-like than the $100 threshold.
- **Crude oil to $105 / $110:** highly correlated with the $100 market; the $100 level had better liquidity, volume, and pricing-bias visibility.
- **Netanyahu out by April 30:** similar No logic to the June 30 market, but the shorter window was more efficiently priced and had no stronger evidence.
- **NCAA long-tail markets:** longshot features were present, but the evidence chain was weaker than in the political and macro markets.

Why the Final Top 3 Entered the Report

Market	Reason for selection	Archived side
Crude Oil $100	The market directly mapped a geopolitical-conflict narrative into the claim that CME settlement would reach $100 before month end. Raven-Agent treated this as narrative amplification.	No
Bitcoin to $65,000	The short-term drop threshold was nontrivial, and the context lacked a new negative catalyst. The No side fit a base-rate reversion under missing evidence.	No
Netanyahu out by June 30	The price was already low, but settlement required a clear resignation, removal, or factual departure. Comments mostly added noise and rule misunderstanding.	No

Translated and condensed from the March 20, 2026 Raven-Agent daily market pulse. Archived system trace, not investment advice.

Figure 3. Example Raven-Agent market screening trace. The artifact records the daily scan, filtering rules, representative rejected candidates, and final selected markets from one archived market pulse.

RAVEN-AGENT ARCHIVE ARTIFACT · FIGURE 2

Probability Reasoning Trace: Crude Oil $100

Translated archive excerpt for one selected market: probability comparison, evidence adjustments, settlement rule, reasoning logic, and position proposal.

Market	**Will Crude Oil (CL) hit $100?**
Snapshot	**2026-03-20 07:00:48 UTC**
Judgment	**2026-03-20 07:00:42 UTC**
Link	**polymarket.com/event/will-crude-oil-cl-hit-by-end-of-march**

SELECTED DIRECTION	MARKET-IMPLIED NO	RAVEN NO ESTIMATE	CONFIDENCE
Buy No	**34.6%**	**82.0%**	**Medium**

Probability Assessment

	Market Pricing	AI Estimate	Edge
Yes	65.4%	18.0%	-47.4% edge on Yes
No	34.6%	82.0%	+47.4% edge on No

Evidence Chain

#	Evidence summary	Source	Direction	Strength	Adj.	Yes prob.
0	Use the market price as a prior, but only as a starting point; allow reversal for longshot and narrative premium.	Polymarket price	-	-	-	65.4%
1	Settlement uses the CME active-month official settlement price, not intraday highs and not the headline meaning of "oil price surge."	CME settlement rules	Down Yes	Strong	-20%	45.4%
2	Annotations only show geopolitical conflict pushing oil above "70+", still far from the $100 threshold.	Polymarket event page	Down Yes	Strong	-17%	28.4%
3	The remaining window before 2026-03-31 is short and would require a very large move; the context gives no hard evidence for that jump.	Research JSON	Down Yes	Medium	-10%	18.4%
4	Comments are mostly jokes and emotion, with no substantive new information.	Comment sample	Down Yes	Weak	-0.4%	18.0%
5	**Final estimate.**	-	-	-	-	**18.0%**

Four-Dimension Analysis

Dimension	Score	Assessment
Information flow	4/5	Crude oil and geopolitics are information-dense, but the market can over-map conflict escalation into a $100 month-end settlement.
Settlement clarity	5/5	The rule is clear: CME active-month official settlement, excluding intraday prices and later revisions.
Liquidity	3/5	Candidate-level liquidity and 24h volume are high, but order-book best bid/ask disagrees with the market snapshot.
Volatility	4/5	Oil is volatile and narrative-driven, making mispricing possible, but short-term headline risk remains high.

Settlement Rule, Reasoning Logic, and Position Suggestion

Basic condition. If, before the last trading day of March 2026, the CME official settlement price for active-month crude oil futures reaches or exceeds $100 on any trading day, the market resolves Yes; otherwise it resolves No.

Reasoning logic. The core error is a mismatch between narrative and settlement. The context supports geopolitical upside risk, but not a move that pushes the CME active-month official settlement to $100 within a very short window. Since settlement does not use intraday highs and the annotations only mention "70+", Raven-Agent treats the 65.4% Yes price as strongly overestimated and selects No.

Position suggestion. Buy No at limit 0.346. Full Kelly: 72.5% of bankroll. Quarter Kelly: 18.1%, or $1,812 on a $10,000 bankroll. Suggested position: 18.1%. Order type: limit order. Estimated monthly return: 950.5% over an 11-day holding period.

Translated and condensed from the selected-market section of the March 20, 2026 Raven-Agent pulse. The archive preserves the reasoning trace before replay analysis.

Figure 4. Example Raven-Agent probability reasoning trace for a selected crude oil market. The artifact records the market-implied probability, Raven-Agent probability estimate, evidence adjustments, settlement-rule checks, and proposed action.