

The Harness Is the Edge: A Disciplined, Self-Improving Language-Model Agent for Weather Prediction Markets

Thomas Wang

thomas@openthomas.com

<https://github.com/PredictionMarketTrader/openthomas> • <https://openthomas.com>

July 15, 2026

Abstract

Every frontier language model given real capital to trade prediction markets autonomously has lost money. We argue this is a *harness* problem, not a forecasting-accuracy one: the profitable signature is selectivity, sized asymmetry, exit discipline, and calibration — properties of the system around the model, not the model itself. We present **OpenThomas**, an open-source agent built to be that harness and pointed at the one category where a harness is measurable: daily station-temperature markets on Kalshi and Polymarket, which settle daily against an objective public record (the U.S. National Weather Service report for a named station). That makes them a *strong-evaluator* domain — dense, objective, and reproducible in leak-free replay. OpenThomas prices each strike from a seven-model numerical-weather-prediction consensus plus its own GraphCast and GenCast forecasts, learns each station’s systematic bias from leak-free hindcasts, lets a language model make a *bounded* adjustment, blends with the market prior, and sizes with fractional Kelly under hard risk limits the model cannot override. It also improves itself: an evolution loop proposes bounded mutations that must clear a kernel-owned gate on leak-free replay, with the evaluator kept off the agent’s edit surface so self-improvement cannot redefine “improvement.” In a 21-day replay on real order books with fees (968 settled markets), a naked model-vs-market rule barely loses (\$0.007/contract, 425 trades) while a market-prior blend turns expectancy positive (+\$0.028) by trading less than half as often (184 trades): selectivity, not the model, is the edge. We report the losing rows too. Contributions: (i) the harness-is-the-edge thesis as a concrete, reproducible system; (ii) a leak-free hindcast/replay methodology for market agents; and (iii) a self-improvement architecture whose safety derives from keeping the judge off the edit surface, evaluated on a domain with a strong judge.

1 Introduction

The obvious way to build a trading agent on top of a large language model (LLM) is to ask the model what will happen and bet accordingly. The empirical record says this does not work. The *Prediction Arena* benchmark [1] gave six frontier models \$10,000 each and let them trade real prediction markets autonomously for 57 days. Every one of them lost money, between -16% and -30.8% . Research *volume* had no correlation with returns; token spend had none either. What separated the least-bad runs from the worst was not a better world model but better *behavior*: being right early, sizing up when confident and down when uncertain, holding winners to settlement rather than exiting early, and — above all — not trading at all when there was no edge. The single profitable run on record ($+10.9\%$) was selective (112 trades), asymmetric (average win \$63.89 vs. average loss \$3.23), and low-drawdown (4.1%).

That profile is a property of the *harness* — the scaffolding that decides when the model is allowed to act, how much, and under what constraints — not of the model’s raw forecasting ability. Our central claim follows:

The LLM is not the edge. The harness is the edge.

This reframing has a sharp methodological consequence. If the edge lives in the harness, then to make progress we need a domain where harness quality is *measurable* — densely, objectively, and without leakage. Most prediction markets fail this test: they settle rarely (one data point every few months), their resolution is ambiguous, and any backtest risks letting the replayed market “know” something the replayed model did not. **Weather markets are the exception.** A daily station-temperature market (“Will the high in Central Park exceed 83°F today?”) resolves against the NWS climatological report for one named station, every day, with no ambiguity. Dozens of ground-truth

data points arrive per day. This is the fastest, cleanest learning loop in prediction markets, and it is what makes both the trading edge and, later, recursive self-improvement, something you can put a number on.

We present **OpenThomas**, an open-source ($\approx 7,300$ lines of Python, 182 tests) autonomous agent that operationalizes the harness thesis on weather markets. Its design commitments are:

- **Selectivity over activity.** “No trade” is the default action. The agent trades only when $|\hat{p} - \text{price}|$ clears an explicit expected-value threshold after fees and spread.
- **A statistical baseline the model may adjust but never replace.** Every strike is priced from a seven-model NWP consensus, corrected by per-(station, lead) bias and spread learned from leak-free hindcasts, and hard-truncated by what today’s thermometer has already locked in. The LLM is clamped to ± 0.15 around this baseline.
- **Hard risk constraints outside the model.** Fractional Kelly sizing, per-market / per-event / per-category caps, a fee-aware EV threshold, a longshot-zone filter, and a drawdown kill-switch are deterministic code. The model proposes; the risk engine disposes.
- **Calibration as a first-class metric,** tracked per (station, lead) as Brier skill against the market, and used to license sizing only where skill is positive.
- **Self-improvement as a gated loop** in which the evaluator is structurally off the agent’s edit surface (Section 5).

We contribute: (1) the harness-is-the-edge thesis operationalized as a concrete system, with an honest empirical account including the losing runs; (2) a leak-free hindcast/replay methodology for evaluating market agents where the replayed market is never allowed to know what the replayed model does not; and (3) a self-improvement architecture whose safety comes from a kernel/agent plane split that keeps the judge off the edit surface, stress-tested on a domain with a genuinely strong judge. The system, the evaluation harness, and the reference agent’s live track record are all public.

2 Background and related work

LLMs as forecasters and traders. A growing literature evaluates LLMs as probabilistic forecasters. The consistent finding is that *pure* LLM forecasts underperform market consensus, but a *blend* of the model with the market price can beat the market [2]. Prediction Arena [1] extends this from forecasting to *live autonomous trading* and finds uniform losses, isolating behavioral rather than epistemic failure modes. Our work takes the blend result and the behavioral diagnosis as design inputs: we blend with the market prior

by construction, and we push the behavioral factors (selectivity, asymmetry, exit discipline) into deterministic code rather than prompting for them.

Neural weather prediction. Machine-learned global weather models now match or exceed operational physics-based systems: GraphCast [3] produces deterministic 10-day forecasts, and GenCast [4] produces calibrated ensembles via a diffusion sampler. These open-weights models can be run on commodity accelerators. OpenThomas uses them as *additional voters* in a forecast consensus and, crucially, post-processes them per settlement station — the place where public global forecasts are systematically biased and where a small, learnable, local correction becomes a durable edge.

Prediction-market microstructure. The favorite—longshot bias — low-priced contracts are systematically overbet — is well documented [5]; we encode it as a hard longshot-zone filter. Cross-venue listing of the same real-world event (Polymarket vs. Kalshi) creates episodic arbitrage, which our scanner flags deterministically.

Self-improving agents. Recent framing of near-term recursive self-improvement (RSI) locates the leverage not in model weights but in the *harness* — how the agent plans, acts, remembers, and is evaluated — and identifies *evaluator strength* as the first bottleneck [6]. Open-ended methods such as the Darwin Gödel Machine [7] keep an archive of past variants as mutation parents to resist diversity collapse. OpenThomas adopts both ideas and adds a security model: because weather markets supply an unusually strong evaluator, the risk is no longer a weak judge but a *gameable* one, so we make the judge structurally unwritable by the optimizer.

3 Why weather markets are a strong-evaluator domain

Three properties make daily station-temperature markets uniquely suited to measuring harness quality.

Dense, objective settlement. Each market resolves against the NWS Climatological Report (product type CLI) for one named station, once per day. There is a single number — the day’s observed maximum or minimum — and no interpretive latitude. A generalist prediction-market agent might see a few dozen settlements per *year*; a weather desk on ten U.S. cities (highs and lows) sees dozens per *day*.

A learnable, local edge. Markets settle on a *specific* station (NYC = Central Park; Chicago = Midway),

and global forecast models have systematic, station-specific biases there. Our first 90-day hindcast found the Miami consensus running 1.8–2.1°F cold (σ 1.7), Philadelphia +2.3°F, and LAX about 1°F warm. These are biases you can estimate from free public data, per station, offline — a moat that does not depend on privileged information, only on the discipline to measure and apply it.

Leak-free reproducibility. Historical “as-of” forecast data (what a model predicted at a past issue time) and historical order books are both available, so a backtest can present the replayed agent with exactly the information it would have had — and no more. This is what lets us evaluate not just the forecast but the *decision rule*, and later, self-improvement, without the replay silently cheating.

The same three properties are what the RSI literature asks for and rarely gets: an objective, dense, reproducible evaluator [6].

4 System architecture

OpenThomas is organized as a stack of layers with a strict rule about who is in control at each one (Figure 1, Table 1). Everything that touches money — sizing and exposure — is deterministic code; the language model is confined to a bounded adjustment in the middle of the pipeline.

4.1 Market connectors and paper simulation

A unified `Market/Order/Position` model spans Poly-market (Gamma API for discovery, CLOB for books/orders) and Kalshi (REST + WebSocket, RSA-signed, CFTC-regulated). A *paper* connector wraps either venue’s real market data but simulates fills locally at bid/ask (never mid), with mark-to-market at bid prices (liquidation value, not hope). Paper mode is the default and needs no exchange keys, which makes the whole system runnable — and reproducible — by anyone.

4.2 The weather baseline

For a market on station s , valid day d , kind $k \in \{\text{high}, \text{low}\}$, the baseline forms a consensus temperature $\bar{T}$ across the seven operational global models (GFS, ECMWF-IFS, ICON, GEM, UKMO, Météo-France, JMA) plus the system’s own GraphCast point forecast, with a cross-model spread σ_{model} . It then applies two corrections learned strictly out-of-window:

$$\hat{T} = \bar{T} + b_{s,k,\ell}, \quad \sigma^2 = \sigma_{\text{model}}^2 + \sigma_{s,k,\ell}^2,$$

where ℓ is the forecast lead, $b_{s,k,\ell}$ is the learned station bias and $\sigma_{s,k,\ell}$ its residual spread. GenCast contributes a *flow-dependent* multiplier on σ (an uncertain synoptic setup widens the distribution) rather than a point estimate, because its 12-hour steps undersample the diurnal peak. The per-strike probability is then the Gaussian (or empirical) mass on the correct side of the strike, with one more hard step: *truncation*. If today’s thermometer has already recorded a value that makes one side of the strike impossible (the observed high already exceeds the floor), the probability is clamped accordingly. This is where a naive backtest leaks, and where Section 6 spends most of its care.

4.3 Forecast engine: bounded LLM adjustment

The language model’s role is deliberately small. It receives the baseline probability, the resolution rules, base rates, and a news brief, and returns an adjusted probability — which is then *clamped to baseline* ± 0.15 . The statistics do the heavy lifting; the model is there for what the statistics cannot see (a blown front, a stale consensus, a discontinuity in the question text), not to substitute its own guess for a calibrated ensemble. Because of the clamp, mid-size local reasoning models suffice, which keeps the agent runnable on self-hosted open-weights models with no API bill and full privacy. The engine takes the median of N independent estimates under different framings to reduce single-sample variance.

4.4 Calibration and market-prior blend

Once ≥ 30 markets settle in a category, OpenThomas fits Platt scaling to correct the model’s systematic over/under-confidence, then blends the calibrated forecast with the market price. The blend is not a hedge — it is the empirically supported position that a model-market blend beats either alone [2]. The blend weight is an entry-*selectivity* parameter (how much to trust our view vs. the crowd), and as such is a candidate for self-improvement under kernel bounds (Section 5); the sizing that follows is not.

4.5 Risk engine (deterministic)

The risk engine takes bankroll, risk profile, the (blended) probability and confidence, and market price/liquidity, and returns an approved size or a rejection naming the binding constraint. It applies fractional Kelly (profile-scaled 0.15 $\times$ –0.33 $\times$), per-market concentration caps, per-category and per-event correlation caps (correlated settlements produced Prediction Arena’s largest single-session losses), a fee-aware EV threshold, a longshot-zone filter, a solvency check including fees, and a draw-down kill-switch that halts trading and requires a hu-

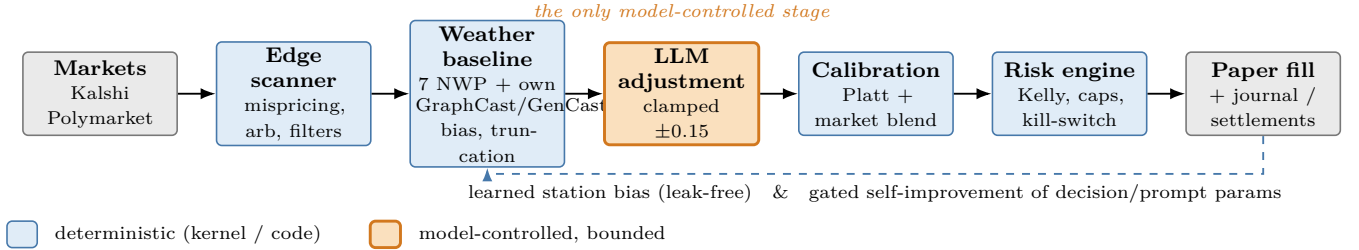


Figure 1: The OpenThomas pipeline. A trade flows left to right; only one stage is model-controlled, and it is clamped to the statistical baseline ± 0.15 and bracketed by deterministic layers on both sides. Settled outcomes feed back to teach the baseline each station’s bias (strictly leak-free) and to gate self-improvement of the decision and forecast-prompt parameters. Sizing and exposure never leave deterministic code.

Layer	What it does	Who’s in control
Edge scanner	Filters markets to plausible mispricings; flags cross-venue arbitrage and longshot/resolution-trap zones	deterministic code
Weather baseline	Seven-model NWP consensus + own GraphCast/GenCast $\rightarrow$ per-strike probability at the settlement station, with per-(station, lead) bias/ σ from leak-free hindcasts and hard truncation by today’s observed extreme	deterministic code
Forecast engine	LLM ensemble (median of N independent estimates), grounded in resolution rules and base rates, <i>clamped to baseline ± 0.15</i>	your choice of model
Calibration	Platt-scales forecasts against your own settled history; blends with the market prior	deterministic code
Risk engine	Fractional Kelly, per-market/event/category caps, fee-aware EV threshold, longshot filter, drawdown kill-switch	deterministic code
Memory	Every forecast and fill journaled to SQLite; post-settlement reflection distills bounded playbook rules	agent, human-auditable

Table 1: The OpenThomas stack. The model *proposes*; the risk engine *disposes*. Only one layer is model-controlled, and it is bounded above and below by deterministic layers.

man to resume. None of these are reachable by the model or by the self-improvement loop; a kernel policy test fails any change that tries to move a safety rail into the optimizer’s reach.

5 Self-improvement with the judge off the edit surface

OpenThomas updates OpenThomas. The operator (a human, or a coding assistant) is moved *up* the stack — setting bounds and reviewing lineage — and is forbidden from hand-tuning trading parameters; that is what the evolution loop is for. The design problem is that self-improvement is safe only if improvement cannot re-define “improvement.” An optimizer that can edit its own judge will find it easier to lower the bar than to clear it.

Kernel/agent planes. The codebase is split into two planes, and the split *is* the security model (Table 2). The **kernel** plane — promotion gate, parameter bounds,

the risk engine and all safety rails, the truth pipeline, the journal schema — is written only by the operator; the evolution loop has no write path to it. The **agent** plane — decision-rule parameters, the forecast prompt template, and (on the roadmap) workflow structure, strategy code, and the evolver itself — is written by the evolution loop, but only *through* the kernel gate. The line is drawn precisely: entry-selectivity knobs (minimum edge, market-prior blend weight) are genome parameters under kernel bounds, while sizing, exposure caps, and the kill-switch can never enter the parameter space.

Three nested loops. A fast *trading* loop (minutes) runs the champion parameters and never waits on anything below it. A *reflection* loop (on settlements) distills bounded playbook rules. An *evolution* loop mines failure evidence from the journal, proposes mutations — LLM-directed by the evidence, plus Gaussian mutations drawn from an *archive* of past variants (Darwin Gödel Machine-style, against diversity collapse [7]) — and submits them to the gate.

Plane	Contents	Written by
kernel	bounds, promotion gate, risk engine & safety rails, truth pipeline, journal schema	operator only
agent	decision-rule params, forecast prompt template; later: workflow, strategy code, the evolver	evolution loop, <i>via the gate</i>

Table 2: The plane split is the security model: the judge is not on the optimizer’s edit surface.

The gate (kernel, frozen). A candidate is a *strategy callable* scored on leak-free replay with four anti-gaming properties: *held-in/held-out* splits (winners are picked on older days; the freshest settled days only get a veto, and the window rolls forward daily so repeated meta-cycles face new data); *totals-not-ratios* scoring with a minimum trade count (a rule that trades twice cannot win on ratio; a rule that trades nothing cannot win on drawdown); *out-of-sample regret rollback* (each cycle first re-scores the active generation against its parent on the fresh window and reverts if the parent now wins); and a *Brier veto* for the forecast operator (a prompt that wins PnL while worsening calibration got lucky, and luck does not promote). Generation 0 is the operator’s config and applies no overrides — the human’s settings win until the loop has promoted evidence-backed changes past them.

One gate, growing write surface. Improvement axes are unified as mutation operators feeding the *same* accept mechanism: playbook rules and calibration refits (shipped), decision-parameter mutation and forecast-prompt mutation (shipped), then workflow structure, then agent-plane code diffs (which add shadow paper-trading and human diff review), then LoRA/GRPO weight updates with a time-discounted reward $R \cdot e^{-\lambda \text{ days-to-close}}$ — a trained adapter is just another candidate at the same gate. The admission rule is strict: a parameter enters the space only when the gate can actually discriminate on it. *Tuning what you cannot measure is drift, not improvement.* This is why code diffs come last — they need the strongest evaluation (replay *plus* shadow trading *plus* human review of the diff) — and why the whole enterprise is anchored to a domain with a strong evaluator in the first place.

6 Leak-free evaluation methodology

The hardest part of evaluating a market agent is not computing PnL; it is not lying to yourself. Two com-

Decision rule (21-day replay)	Trades	Win	PnL/ct
Naked model-vs-market	425	41%	−\$0.007
+ learned station bias	396	39%	−\$0.005
+ market-prior blend (production)	184	45%	+\$0.028

Table 3: Leak-free replay over 968 settled markets (2026-06-24 to 07-14; fees in; timing-safe snapshots — lows 06Z, highs 15Z; station bias learned strictly out-of-window). The market-prior blend more than halves activity (425 $\rightarrow$ 184 trades) and flips expectancy positive: *selectivity*, not the model, is the edge. The naked rule barely clears zero because the late-morning book has already priced the day’s early observations.

mands encode the discipline.

hindcast teaches the baseline each station’s bias from historical as-of forecasts and official settlements, strictly *before* any trading window it will be evaluated on. **replay** then backtests the *decision rule* against real historical order books, drawing prices from snapshot candlesticks that *precede* the temperature extreme being bet on, with station bias/ σ learned only out-of-window. The governing rule, learned the hard way when an early backtest snapshotted the dawn low after it had already been realized: *never let the replayed market know something the replayed model does not.* A single timing leak turned a losing strategy into a fake winner; fixing it turned it back.

This methodology is what makes the self-improvement gate trustworthy: the same `collect_rows` pipeline that scores a candidate generation is the one that enforces no-peeking, and it lives in the kernel where the optimizer cannot touch it.

7 Empirical results

We report an ablation from a 21-day leak-free replay on real Kalshi order books with fees included (Table 3); the window covers 968 settled markets with settlement dates from 2026-06-24 to 07-14. These are small-sample, paper-mode numbers on a baseline *deliberately stripped* of every live-only advantage (same-day model runs, intraday observations, LLM adjustment, and the calibration layer); they are an existence proof about *where the edge comes from*, not a track record. The live agent trades with all four advantages on, and whether they clear the bar is precisely what the public paper run exists to answer.

Two findings are worth drawing out. First, the naked model-vs-market rule *loses* (−\$0.007/contract over 425 trades): by late morning the order book has already digested the day’s early observations, so the raw model

has almost nothing to add — exactly the “the model is not the edge” prediction, borne out on real books. Second, the edge is recovered not by a sharper forecast but by *selectivity*: blending the model with the market prior more than halves activity (425 $\rightarrow$ 184 trades) and flips expectancy to +\$0.028/contract at a 45% win rate, i.e. the surviving trades pay asymmetrically. Learned station bias adds a smaller, consistent nudge ($-\$0.007 \rightarrow -\0.005); the 90-day hindcast that supplies it recovers stable, station-specific biases (Miami +1.8–2.1°F, Philadelphia +2.3, LAX -1) that persist out-of-sample.

We report the losing rows next to the positive one, and stress the caveats: 184 trades over three weeks in paper mode is an existence proof, not a track record, and a single window can flatter a rule. The honest claim is narrow and, we think, more useful than a headline return: on weather markets the structural edges are *real but small*, discipline — selectivity and blending with the crowd, not a smarter model — is what realizes them, and this is the market category where that statement can be checked.

8 Discussion and limitations

Small samples, paper mode. The replay is 21 days and simulated at real bid/ask; it is an ablation, not a track record. The reference agent’s live paper run is the ongoing, public test, reported with wins and losses alike.

Capacity is the real ceiling. Daily-temperature markets are thin. A survey of seven Kalshi cities found on the order of a few hundred thousand dollars of daily notional in aggregate, which bounds extraction at low-single-digit millions per year even under optimistic assumptions. This is a research and methodology contribution about *measurable* edge, not a claim of scalable returns.

Replay fidelity. Replay has no news brief, no intraday observations, no forecaster-discussion text, and no calibration layer; it measures the template’s skill at adjusting the statistical baseline, which is the dominant live pathway for weather markets, but it is not the whole live agent. A residual leak risk — an LLM “remembering” recent weather from pre-training — is negligible with a local model whose cutoff precedes the window, but must be rechecked whenever the forecaster is swapped.

Generalization. The strong-evaluator property is what makes weather the right first domain; it is also what most other markets lack. Extending the harness (and the self-improvement gate) to weaker-evaluator

markets is future work, and the journal already archives each live forecast’s inputs so that a future trace-replay evaluator can begin to close that gap.

9 Reproducibility, ethics, and broader impact

The system, the evaluation harness, and the reference agent’s live feed are public; paper mode requires no exchange keys and reproduces the replay numbers from public data. Live trading is off by default and requires two independent explicit opt-ins; the agent has no deposit, withdrawal, or transfer capability and can only trade within an allocated bankroll. We ship hard position limits and a drawdown kill-switch, and we state plainly that no software makes trading safe: roughly 70% of Polymarket addresses have lost money, and every layer of this paper is about losing less, not winning easily. On the self-improvement side, the safety argument is architectural rather than behavioral: the judge is off the optimizer’s edit surface by construction, safety rails are outside the parameter space by a kernel policy test, and every future promotion is an auditable commit authored by the loop itself.

10 Conclusion

Autonomous LLM traders lose money not because they forecast badly but because they lack discipline. OpenThomas puts the discipline in deterministic code, uses a language model only for a bounded adjustment it is well-suited to, and points the whole system at weather markets — the one category where the resulting edge, and the system’s ongoing self-improvement, can be measured densely, objectively, and without leakage. The edge we find is real, small, and preserved by discipline. We report it in public, losses included, and we invite the community to check it.

References

- [1] Prediction Arena: evaluating frontier language models as autonomous prediction-market traders. arXiv:2604.07355, 2026.
- [2] Calibrated LLM forecasts and market-prior blending. arXiv:2511.07678, 2025.
- [3] R. Lam et al. GraphCast: Learning skillful medium-range global weather forecasting. *Science*, 2023.
- [4] I. Price et al. GenCast: Probabilistic weather forecasting with machine learning (diffusion ensembles). *Nature*, 2025.
- [5] J. Whelan et al. The favorite–longshot bias in prediction markets. SSRN 5502658, 2026.

- [6] L. Weng. Harness Engineering for Self-Improvement. <https://lilianweng.github.io/posts/2026-07-04-harness/>, 2026.
- [7] The Darwin Gödel Machine: open-ended self-improvement with a variant archive. 2025.